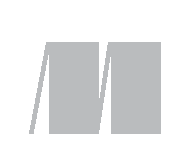
 [Translated from "?" to Vietnamese - www.onlinedoctranslator.com](https://www.onlinedoctranslator.com/en/?utm_source=onlinedoctranslator&utm_medium=docx&utm_campaign=attribution)



ẤN BẢN THỨ HAI

Madhusudhan Konda

Lời nói đầu củaShay Banon

## NGƯỜI ĐÀN ÔNG

**Elastic Stack: Beats, Logstash, Elasticsearch và Kibana**



Beats (người vận chuyển dữ liệu)

**Beats giúp đưa dữ liệu vào Elasticsearch.**

**Kibana là một trang web đa diện**

**ứng dụng (UI).**

Khách hàng

Tập tin

**Elasticsearch, một công cụ tìm kiếm và phân tích, là trung tâm của Elastic Stack.**

Nhịp tim

**2 VÀphi công**

Nhịp điệu

Chức năngbeat

Kiểm toán nhịp

Bộ xử lý dữ liệu

**Logstash là một công cụ xử lý dữ liệu ETL (trích xuất, chuyển đổi, tải).**

Dịch vụ tìm kiếm

**Ứng dụng do người dùng thiết kế**

Nội bộứng dụng người dùng

Ứng dụng hướng tới người dùng bên ngoài

***Lời khen ngợi cho phiên bản đầu tiên***

Elasticsearch in Action có phạm vi bao quát sâu rộng về cách sử dụng Elasticsearch để đạt được tiềm năng tối đa.

—Paul Vẫn

*Elasticsearch là một chủ đề phức tạp và cuốn sách này là nguồn tài liệu tốt nhất. Tôi thực sự khuyên bạn nên đọc!*

—Daniel Beck, công ty luật GmbH

*Một cuốn sách tuyệt vời cần có khi bạn bắt đầu làm việc với Elasticsearch.*

—Tanguy Leroux, kỹ sư phần mềm, Elastic

*Cuốn sách Elasticsearch hay nhất. Tôi không cần phải tìm kiếm thêm nữa.*

—Koray mạnh mẽ

*Cuốn sách này sẽ trở thành cẩm nang không thể thiếu của bạn để giải quyết những thách thức của bán cấu trúcdữ liệu.*

—Artur Nowak, Giám đốc công nghệ, Evidence Prime

*Hướng dẫn dễ hiểu dành cho người mới bắt đầu về hệ thống tìm kiếm hiện đại quy mô lớn.*

—Sen Xu, Kỹ sư phần mềm cấp cao, Twitter

*Đã đưa tôi từ trạng thái bối rối đến tự tin chỉ trong một tuần.*

—Alan McCann, CTO, Givsum.com

**tôi**

*Elasticsearch trong hành động*

*Phiên bản thứ hai*

*Elasticsearch trong hành động*

**STHÁNG 12VÀPHẦN MỞ ĐẦU**

##### MADHUSUDHAN KONDA

Lời nói đầu của SHAY BẨN DANH

##### NGƯỜI ĐÀN ÔNG

SHÃNGTÔIQUỐC GIA

Để biết thông tin trực tuyến và đặt mua sách này và các sách khác của Manning, vui lòng truy cập[www.manning.com.](http://www.manning.com/)Nhà xuất bản cung cấp giảm giá cho cuốn sách này khi đặt hàng với số lượng lớn. Để biết thêm thông tin, vui lòng liên hệ

Phòng Bán Hàng Đặc Biệt Manning Publications Co. 20 Baldwin Road

Hòm thư 761

Đảo Shelter, NY 11964 Email:[đơn hàng@manning.com](mailto:orders@manning.com)

©2023 của Manning Publications Co. Bảo lưu mọi quyền.

Không được sao chép, lưu trữ trong hệ thống truy xuất hoặc truyền tải bất kỳ phần nào của ấn phẩm này dưới bất kỳ hình thức nào hoặc bằng phương tiện điện tử, cơ học, photocopy hoặc cách khác mà không có sự cho phép trước bằng văn bản của nhà xuất bản.

Nhiều tên gọi do nhà sản xuất và người bán sử dụng để phân biệt sản phẩm của họ được coi là nhãn hiệu. Khi những tên gọi đó xuất hiện trong sách và Manning Publications biết về khiếu nại về nhãn hiệu, thì những tên gọi đó được in bằng chữ in hoa đầu hoặc toàn bộ chữ in hoa.

Nhận thức được tầm quan trọng của việc bảo tồn những gì đã viết, chính sách của Manning là in những cuốn sách chúng tôi xuất bản trên giấy không axit và chúng tôi nỗ lực hết sức vì mục đích đó.

Nhận thức được trách nhiệm bảo tồn tài nguyên của hành tinh, sách Manning được in trên loại giấy được tái chế và xử lý ít nhất 15 phần trăm mà không sử dụng clo nguyên tố.

Tác giả và nhà xuất bản đã nỗ lực hết sức để đảm bảo rằng thông tin trong cuốn sách này là chính xác tại thời điểm xuất bản. Tác giả và nhà xuất bản không chịu trách nhiệm và từ chối bất kỳ trách nhiệm nào đối với bất kỳ bên nào về bất kỳ mất mát, thiệt hại hoặc gián đoạn nào do lỗi hoặc thiếu sót, cho dù lỗi hoặc thiếu sót đó là do sơ suất, tai nạn hoặc bất kỳ nguyên nhân nào khác, hoặc do bất kỳ cách sử dụng thông tin nào ở đây.

Xuất bản ManningĐồng phát triển biên tập: Ian Hough 20 BaldwinĐường bộPhát triển kỹ thuật biên tập viên: Al Krinker

Hộp thư bưu điện761Đánh giá biên tập viên: Đảo trú ẩn Aleksandar Dragosavljević, NY11964Sản xuất biên tập: Keri Hales

Biên tập viên: Tiffany Taylor Biên tập viên: Jason Everett

Biên tập viên kỹ thuật: Simon Hewitt Người sắp chữ và thiết kế bìa: Marija Tudor

Mã số 9781617299858

Được in tại Hoa Kỳ

*Để tưởng nhớ đến cha tôi.*

*Chúng tôi nhớ bạn!*

*Tình yêu, Konda*

# nội dung tóm tắt

1. ■ [Tổng quan1](#_bookmark6)
2. ■ [Nhậnbắt đầu20](#_bookmark83)
3. ■ [Kiến trúc60](#_bookmark275)
4. ■ [Bản đồ100](#_bookmark429)
5. ■ [Làm việc vớitài liệu149](#_bookmark661)
6. ■ [Lập chỉ mụchoạt động192](#_bookmark884)
7. ■ [Chữphân tích235](#_bookmark1062)
8. ■ [Giới thiệutìm kiếm280](#_bookmark1278)
9. ■ [Cấp độ học kỳtìm kiếm313](#_bookmark1453)
10. ■ [Toàn văntìm kiếm334](#_bookmark1546)
11. ■ [Hợp chấttruy vấn361](#_bookmark1685)
12. ■ [Trình độ caotìm kiếm397](#_bookmark1822)
13. ■ [Tổng hợp434](#_bookmark1970)
14. ■ [Quản lý467](#_bookmark2142)
15. ■ [Hiệu suất vàxử lý sự cố497](#_bookmark2289)

**viii**

# nội dung

[*Lời nói đầu xviii*](#_bookmark0)[*lời nói đầu xix*](#_bookmark1)[*lời cảm ơn xxi*](#_bookmark2)[*về cuốn sách này xxiv*](#_bookmark3)

[*về tác giả xxix*](#_bookmark4)

[*về hình minh họa bìa xxxi*](#_bookmark5)

[***Tổng quan1***](#_bookmark6)

[*1*](#_bookmark6)

* 1. [Điều gì tạo nên một tìm kiếm tốtđộng cơ?2](#_bookmark8)
  2. [Tìm kiếm là cái mớibình thường3](#_bookmark11)

[*Có cấu trúc so với không có cấu trúc (toàn văn)ngày3*](#_bookmark13) ■ [*Tìm kiếm được hỗ trợ*](#_bookmark18)[*bởi mộtcơ sở dữ liệu 5*](#_bookmark18)■ [*Cơ sở dữ liệu so với công cụ tìm kiếm 6*](#_bookmark21)

* 1. [Tìm kiếm hiện đạiđộng cơ7](#_bookmark26)

[*Chức năng8*](#_bookmark28) ■ [*Công cụ tìm kiếm phổ biến 8*](#_bookmark30)

* 1. [Tìm kiếm đàn hồiTổng quan9](#_bookmark40)

[*Các lĩnh vực cốt lõi 10*](#_bookmark43)■ [*Đàn hồingăn xếp11*](#_bookmark50) ■ [*Sử dụng Elasticsearchcác trường hợp 12*](#_bookmark56)[*Elasticsearch không phù hợpsử dụng 13*](#_bookmark67)■ [*Những quan niệm sai lầm 15*](#_bookmark72)

* 1. [Áp dụng phổ biến16](#_bookmark75)
  2. [AI tạo sinh và hiện đạitìm kiếm16](#_bookmark78)

[***Nhậnbắt đầu 20***](#_bookmark83)

[*2*](#_bookmark83)

* 1. [Chuẩn bị Elasticsearch vớidữ liệu21](#_bookmark85)

[*Một hiệu sách trực tuyến 21*](#_bookmark87)■ [*Lập chỉ mục tài liệu 23*](#_bookmark95)■ [*Lập chỉ mục*](#_bookmark103)[*tài liệu đầu tiên của chúng tôi 26*](#_bookmark103)■ [*Đang lập chỉ mục thêm tài liệu 30*](#_bookmark116)

**ix**

**x MỤC LỤC**

* 1. [Lấy lạidữ liệu30](#_bookmark118)

[*Đếm tài liệu 31*](#_bookmark121)■ [*Lấy lại tài liệu 31*](#_bookmark125)

* 1. [Toàn văntìm kiếm36](#_bookmark146)

[*Truy vấn khớp: Sách của tác giả 36*](#_bookmark148)■ [*Phù hợp với truy vấn với*](#_bookmark153)[*Toán tử AND 37*](#_bookmark153)■ [*Lập chỉ mục tài liệu bằng cách sử dụng \_bulk API 39*](#_bookmark160)[*Tìm kiếmtrên nhiều lĩnh vực 41*](#_bookmark167)■ [*Tăng cường kết quả 42*](#_bookmark176)

[*Tìm kiếmcụm từ 42*](#_bookmark178)■ [*Các cụm từ bị thiếutừ 44*](#_bookmark187)[*Xử lý lỗi chính tả 44*](#_bookmark191)

* 1. [Cấp độ học kỳtruy vấn45](#_bookmark198)

[*Thuật ngữtruy vấn 46*](#_bookmark204)■ [*Truy vấn phạm vi 47*](#_bookmark207)

* 1. [Hợp chất truy vấn47](#_bookmark213)

[*Boolean (bool)truy vấn 48*](#_bookmark216)■ [*Điều khoản bắt buộc 49*](#_bookmark219)■ [*Điều không nên làm*](#_bookmark227)[*điều khoản 50*](#_bookmark227) ■ [*Mệnh đề nên 51*](#_bookmark233)■ [*Điều khoản lọc 51*](#_bookmark237)

* 1. [Tổng hợp52](#_bookmark243)

[*Số liệu 53*](#_bookmark254)■ [*Xôtổng hợp 56*](#_bookmark264)

[***Kiến trúc60***](#_bookmark275)

[*3*](#_bookmark275)

* 1. [Một cấp độ caoTổng quan61](#_bookmark276)

[*Dữ liệuin62*](#_bookmark278) ■ [*Xử lýdữ liệu 63*](#_bookmark282)■ [*Dữ liệu ra 66*](#_bookmark296)

* 1. [Tòa nhàkhối67](#_bookmark300)

[*Tài liệu 67*](#_bookmark302)■ [*Chỉ số 71*](#_bookmark317)■ [*Luồng dữ liệu 72*](#_bookmark323)■ [*Mảnh vỡ*](#_bookmark328)[*và bản sao 74*](#_bookmark328)■ [*Các nút vàcác cụm 81*](#_bookmark344)

* 1. [đảo ngượcchỉ số86](#_bookmark378)
  2. [Sự liên quan89](#_bookmark383)

[*Điểm liên quan 89*](#_bookmark385)■ [*Thuật toán liên quan (tương tự) 90*](#_bookmark391)

* 1. [Lộ trìnhthuật toán96](#_bookmark413)
  2. [Tỷ lệ97](#_bookmark419)

[*Mở rộng quy mô (tỷ lệ theo chiều dọc) 97*](#_bookmark421)■ [*Thu nhỏ (ngang*](#_bookmark425)[*tỷ lệ)98*](#_bookmark425)

[***Bản đồ100***](#_bookmark429)

[*4*](#_bookmark429)

* 1. [Tổng quan vềlập bản đồ101](#_bookmark431)

[*Bản đồđịnh nghĩa102*](#_bookmark433) ■ [*Lập chỉ mục một tài liệu cho*](#_bookmark438)[*lần đầu tiên 103*](#_bookmark438)

* 1. [Năng động lập bản đồ104](#_bookmark450)

[*Cơ chế suy luận các loại 105*](#_bookmark454)■ [*Những hạn chế của*](#_bookmark460)[*năng độnglập bản đồ107*](#_bookmark460)

**MỤC LỤC xi**

* 1. [Bản đồ rõ ràng 109](#_bookmark473)

[*Bản đồ sử dụng API lập chỉ mục 110*](#_bookmark477)■[*Đang cập nhật lược đồ*](#_bookmark484)[*sử dụng API lập bản đồ 112*](#_bookmark484)■[*Sửa đổi các trường hiện có là*](#_bookmark495)[*không được phép 113*](#_bookmark495)■[*Loại ép buộc 114*](#_bookmark501)

* 1. [Kiểu dữ liệu 115](#_bookmark505)
  2. [Kiểu dữ liệu cốt lõi 118](#_bookmark530)

[*Dữ liệu văn bảnkiểu 118*](#_bookmark534)■ [*Dữ liệu từ khóacác loại 122*](#_bookmark549)

[*Kiểu dữ liệu ngày 124*](#_bookmark559)■ [*Dữ liệu sốcác loại 127*](#_bookmark562)

[*Kiểu dữ liệu boolean 127*](#_bookmark565)■ [*Dữ liệu phạm vicác loại 128*](#_bookmark569)

[*Dữ liệu địa chỉ IP (ip)loại130*](#_bookmark586)

* 1. [Dữ liệu nâng caoloại131](#_bookmark595)

[*Dữ liệu geo\_pointkiểu 131*](#_bookmark597)■ [*Dữ liệu đối tượngkiểu 134*](#_bookmark604)

[*Kiểu dữ liệu lồng nhau 138*](#_bookmark624)■ [*Dữ liệu phẳngkiểu 140*](#_bookmark631)

[*Kiểu dữ liệu nối 142*](#_bookmark634)■ [*Dữ liệu search\_as\_you\_typekiểu 144*](#_bookmark644)

* 1. [Các trường có nhiều dữ liệuloại146](#_bookmark652)

[***Làm việc vớitài liệu 149***](#_bookmark661)

[*5*](#_bookmark661)

* 1. [Lập chỉ mục tài liệu150](#_bookmark664)

[*Tài liệuAPI151*](#_bookmark666) ■ [*Cơ học củalập chỉ mục158*](#_bookmark699)

[*Tùy chỉnh làm mớiquá trình 160*](#_bookmark702)

* 1. [Lấy lạitài liệu161](#_bookmark713)

[*Sử dụng tài liệu đơnGiao diện lập trình ứng dụng (API) 162*](#_bookmark715)■ [*Lấy lại nhiều*](#_bookmark725)[*tài liệu 163*](#_bookmark725)■ [*Các idtruy vấn165*](#_bookmark735)

* 1. [Thao tácphản hồi166](#_bookmark741)

[*Xóa siêu dữ liệu khỏi phản hồi 166*](#_bookmark743)■ [*Đàn áp*](#_bookmark748)[*tài liệu nguồn 167*](#_bookmark748)■ [*Bao gồm và loại trừtrường167*](#_bookmark751)

* 1. [Đang cập nhật tài liệu170](#_bookmark759)

[*Cập nhật tài liệucơ học 170*](#_bookmark761)■ [*\_Cập nhậtAPI171*](#_bookmark767)

[*Cập nhật theo kịch bản 173*](#_bookmark778)■ [*Thay thếtài liệu 177*](#_bookmark809)

[*Upserts 178*](#_bookmark815)■ [*Cập nhật nhưupserts 179*](#_bookmark819)■ [*Đang cập nhậtvới một*](#_bookmark825)[*truy vấn180*](#_bookmark825)

* 1. [Xóa bỏtài liệu181](#_bookmark831)

[*Xóa với ID 181*](#_bookmark833)■ [*Xóa theo truy vấn*](#_bookmark840)[*(\_xóa\_bởi\_truy\_cấp) 182*](#_bookmark840)■ [*Xóa bằng mộttruy vấn phạm vi 183*](#_bookmark846)

[*Xóa tất cảtài liệu 184*](#_bookmark849)

* 1. [Làm việc với các tài liệu trongsố lượng lớn185](#_bookmark855)

[*Định dạng của \_bulk API 185*](#_bookmark860)■ [*Lập chỉ mục hàng loạttài liệu186*](#_bookmark864)

[*Các thực thể độc lập và nhiều hành động 188*](#_bookmark867)■ [*Yêu cầu hàng loạt*](#_bookmark875)[*sử dụngcURL189*](#_bookmark875)

**xii MỤC LỤC**

* 1. [Lập chỉ mục lại tài liệu190](#_bookmark880)

[***Lập chỉ mụchoạt động 192***](#_bookmark884)

[*6*](#_bookmark884)

* 1. [Lập chỉ mục hoạt động193](#_bookmark886)
  2. [Tạo chỉ số193](#_bookmark888)

[*Tạo chỉ mục ngầm định (tự độngsáng tạo) 194*](#_bookmark893)[*Tạo chỉ mụcrõ ràng 198*](#_bookmark908)■ [*Chỉ mục với tùy chỉnh*](#_bookmark910)[*cài đặt 198*](#_bookmark910)■ [*Chỉ số vớiánh xạ 201*](#_bookmark919)■ [*Chỉ mục với*](#_bookmark925)[*bí danh 202*](#_bookmark925)

* 1. [Đọcchỉ số207](#_bookmark937)

[*Đọc chỉ mục công khai 207*](#_bookmark939)■ [*Đọc chỉ mục ẩn 208*](#_bookmark945)

* 1. [Xóa bỏchỉ số209](#_bookmark949)
  2. [Đóng và mởchỉ số210](#_bookmark961)

[*Chỉ số đóng cửa 210*](#_bookmark963)■ [*Mở chỉ số 211*](#_bookmark968)

* 1. [Mục lụcmẫu212](#_bookmark974)

[*Tạo mẫu có thể cấu hình (chỉ mục) 213*](#_bookmark979)■ [*Tạo*](#_bookmark987)[*mẫu thành phần 215*](#_bookmark987)

* 1. [Giám sát và quản lýchỉ số216](#_bookmark994)

[*Mục lụcthống kê217*](#_bookmark996) ■ [*Nhiều chỉ số và số liệu thống kê 218*](#_bookmark1007)

* 1. [Trình độ caohoạt động220](#_bookmark1011)

[*Chia tách mộtchỉ số220*](#_bookmark1012) ■ [*Thu hẹp mộtchỉ số222*](#_bookmark1018) ■ [*Lăn*](#_bookmark1025)[*trên một bí danh chỉ mục 224*](#_bookmark1025)

* 1. [Quản lý vòng đời chỉ mục(ILM)227](#_bookmark1037)

[*Mục lụcvòng đời 228*](#_bookmark1040)■ [*Quản lý vòng đời chỉ mục*](#_bookmark1043)[*thủ công229*](#_bookmark1043) ■ [*Vòng đời với rollover 231*](#_bookmark1051)

[***Chữphân tích235***](#_bookmark1062)

[*7*](#_bookmark1062)

* 1. [Tổng quan236](#_bookmark1069)

[*Truy vấn không có cấu trúcdữ liệu 237*](#_bookmark1071)■ [*Máy phân tích đến*](#_bookmark1073)[*cứu hộ 238*](#_bookmark1073)

* 1. [Máy phân tíchmô-đun238](#_bookmark1076)

[*Mã hóa238*](#_bookmark1079) ■ [*Chuẩn hóa239*](#_bookmark1084) ■ [*Giải phẫu*](#_bookmark1087)[*của một máy phân tích 239*](#_bookmark1087)■ [*Máy phân tích thử nghiệm 241*](#_bookmark1093)

* 1. [Tích hợp sẵnmáy phân tích243](#_bookmark1110)

[*Tiêu chuẩnmáy phân tích 244*](#_bookmark1116)■ [*Máy phân tích đơn giản 251*](#_bookmark1137)[*Khoảng trắngmáy phân tích 251*](#_bookmark1140)■ [*Bộ phân tích từ khóa 252*](#_bookmark1144)[*Dấu vân taymáy phân tích 253*](#_bookmark1148)■ [*Mẫu hìnhmáy phân tích 254*](#_bookmark1155)[*Ngôn ngữmáy phân tích 256*](#_bookmark1164)

**MỤC LỤC xiii**

* 1. [Phong tụcmáy phân tích258](#_bookmark1174)

[*Trình độ caotùy chỉnh 260*](#_bookmark1182)

* 1. [Chỉ địnhmáy phân tích261](#_bookmark1188)

[*Máy phân tích cholập chỉ mục261*](#_bookmark1190) ■ [*Máy phân tích cho*](#_bookmark1197)[*tìm kiếm263*](#_bookmark1197)

* 1. [Tính cáchbộ lọc265](#_bookmark1209)

[*Bộ lọc HTML strip (hmtl\_strip) 266*](#_bookmark1214)■ [*Bản đồ*](#_bookmark1221)[*tính cáchlọc 267*](#_bookmark1221)■ [*Ánh xạ qua một tập tin 268*](#_bookmark1226)■ [*Các*](#_bookmark1229)[*ký tự pattern\_replacelọc 269*](#_bookmark1229)

* 1. [Mã thông báo270](#_bookmark1233)

[*Tiêu chuẩnbộ mã hóa 270*](#_bookmark1235)■ [*ngram và edge\_ngram*](#_bookmark1243)[*bộ mã hóa 271*](#_bookmark1243)■ [*Khácbộ phân tích 274*](#_bookmark1251)

* 1. [Mã thông báobộ lọc274](#_bookmark1259)

[*Bộ lọc phiếu bầu 275*](#_bookmark1264)■ [*Bộ lọc ván lợp 275*](#_bookmark1266)■ [*Các*](#_bookmark1269)[*bộ lọc đồng nghĩa 276*](#_bookmark1269)

[***Giới thiệutìm kiếm 280***](#_bookmark1278)

[*8*](#_bookmark1278)

* 1. [Tổng quan281](#_bookmark1280)
  2. [Làm thế nàotìm kiếm công việc?282](#_bookmark1289)
  3. [Mẫu phimdữ liệu283](#_bookmark1295)
  4. [Tìm kiếmnguyên tắc cơ bản285](#_bookmark1299)

[*Điểm cuối \_search 285*](#_bookmark1302)■ [*Truy vấn so với bộ lọc*](#_bookmark1305)[*ngữ cảnh286*](#_bookmark1305)

* 1. [Giải phẫu của một yêu cầu và mộtphản hồi288](#_bookmark1315)

[*Yêu cầu tìm kiếm 288*](#_bookmark1316)■ [*Tìm kiếmphản hồi 290*](#_bookmark1324)

* 1. [Yêu cầu URItìm kiếm291](#_bookmark1337)

[*Tìm kiếm phim theo tiêu đề 292*](#_bookmark1341)■ [*Tìm kiếm một cụ thể*](#_bookmark1346)[*phim293*](#_bookmark1346) ■ [*Các tham số bổ sung 293*](#_bookmark1352)■ [*Hỗ trợ*](#_bookmark1358)[*Yêu cầu URI với QueryDSL294*](#_bookmark1358)

* 1. [Truy vấnDSL294](#_bookmark1362)

[*Vật mẫutruy vấn 295*](#_bookmark1366)■ [*Truy vấn DSL cho cURL 295*](#_bookmark1369)■ [*Truy vấn*](#_bookmark1373)[*DSL cho tổng hợp 296*](#_bookmark1373)■ [*Lá và hợp chất*](#_bookmark1375)

[*truy vấn296*](#_bookmark1375)

* 1. [Tìm kiếmtính năng297](#_bookmark1381)

[*Phân trang298*](#_bookmark1383) ■ [*Làm nổi bật299*](#_bookmark1395) ■ [*Giải thích sự liên quan*](#_bookmark1400)[*điểm 301*](#_bookmark1400)■ [*Sắp xếp 304*](#_bookmark1419)■ [*Thao táckết quả307*](#_bookmark1427)

[*Tìm kiếm trên các chỉ mụcvà luồng dữ liệu 310*](#_bookmark1446)

**xiv MỤC LỤC**

[***Cấp độ học kỳtìm kiếm 313***](#_bookmark1453)

[*9*](#_bookmark1453)

* 1. [Tổng quan về cấp độ học kỳtìm kiếm314](#_bookmark1455)

[*Các truy vấn cấp độ thuật ngữ không phải làđã phân tích 314*](#_bookmark1457)■ [*Truy vấn cấp độ thuật ngữ*](#_bookmark1459)[*Ví dụ 314*](#_bookmark1459)

* 1. [Thuật ngữtruy vấn316](#_bookmark1467)

[*Thuật ngữ truy vấn trên văn bảncánh đồng 316*](#_bookmark1470)■ [*Thuật ngữ ví dụtruy vấn 317*](#_bookmark1476)[*Truy vấn cấp độ thuật ngữ rút gọn 318*](#_bookmark1479)

* 1. [Các điều khoảntruy vấn318](#_bookmark1482)

[*Ví dụ thuật ngữtruy vấn 319*](#_bookmark1488)■ [*Các thuật ngữ tra cứu*](#_bookmark1493)[*truy vấn320*](#_bookmark1493)

* 1. [Các idtruy vấn321](#_bookmark1497)
  2. [Sự tồn tạitruy vấn322](#_bookmark1501)
  3. [Phạm vitruy vấn323](#_bookmark1505)
  4. [Ký tự đại diệntruy vấn325](#_bookmark1516)
  5. [Cáctiền tố query327](#_bookmark1524)

[*Truy vấn rút gọn 328*](#_bookmark1527)■ [*Tăng tốc tiền tố*](#_bookmark1530)[*truy vấn 329*](#_bookmark1530)

* 1. [Mờtruy vấn330](#_bookmark1537)

[***Toàn văntìm kiếm 334***](#_bookmark1546)

[*10*](#_bookmark1546)

* 1. [Tổng quan335](#_bookmark1549)

[*Độ chính xác335*](#_bookmark1551) ■ [*Thu hồi336*](#_bookmark1556)

* 1. [Vật mẫudữ liệu338](#_bookmark1562)
  2. [Trận đấu\_tất cảtruy vấn338](#_bookmark1565)

[*Xây dựng match\_alltruy vấn338*](#_bookmark1567) ■ [*Dạng rút gọn của a*](#_bookmark1572)[*khớp\_tất\_cảtruy vấn 339*](#_bookmark1572)

* 1. [Trận đấu không cótruy vấn339](#_bookmark1575)
  2. [Trận đấutruy vấn340](#_bookmark1579)

[*Định dạng của một trận đấutruy vấn 340*](#_bookmark1581)■ [*Tìm kiếm bằng cách sử dụng một trận đấu*](#_bookmark1585)[*truy vấn341*](#_bookmark1585) ■ [*Phân tích truy vấn phù hợp 342*](#_bookmark1587)■ [*Tìm kiếm*](#_bookmark1591)[*nhiềutừ 342*](#_bookmark1591)■ [*Phù hợp với ít nhất một vài từ 343*](#_bookmark1596)[*Sửa lỗi đánh máy bằng cách sử dụng tính năng mờtừ khóa 344*](#_bookmark1602)

* 1. [Cụm từ match\_phrasetruy vấn345](#_bookmark1608)
  2. [Tiền tố match\_phrase\_truy vấn346](#_bookmark1616)
  3. [Đa\_trận\_đấutruy vấn347](#_bookmark1624)

[*Các lĩnh vực tốt nhất 348*](#_bookmark1626)■ [*dis\_max là gìtruy vấn350*](#_bookmark1637) ■ [*Máy cắt dây buộc 350*](#_bookmark1640)[*Tăng cường các lĩnh vực cá nhân 351*](#_bookmark1645)

[MỤC LỤC xv](#_TOC_250000)

* 1. [Chuỗi truy vấntruy vấn353](#_bookmark1653)

[Các trường trong một query\_stringtruy vấn 354](#_bookmark1658) ■ [Mặc địnhcác nhà điều hành 355](#_bookmark1663)

[Truy vấn query\_string vớimột cụm từ 356](#_bookmark1668)

* 1. [Mờtruy vấn357](#_bookmark1675)
  2. [Chuỗi đơn giảntruy vấn357](#_bookmark1678)
  3. [Chuỗi truy vấn đơn giảntruy vấn358](#_bookmark1681)

[Hợp chấttruy vấn 361](#_bookmark1685)

[*11*](#_bookmark1685)

* 1. [Sản phẩm mẫudữ liệu362](#_bookmark1688)

[Các sản phẩmsơ đồ 362](#_bookmark1690) ■ [Lập chỉ mụcsản phẩm363](#_bookmark1696)

* 1. [Hợp chất truy vấn364](#_bookmark1700)
  2. [Boolean (bool)truy vấn365](#_bookmark1710)

[Truy vấn boolkết cấu 366](#_bookmark1712) ■ [Phảikhoản 367](#_bookmark1715)

[Tăng cường điều khoản bắt buộc 368](#_bookmark1722) ■ [Điều không nên làmđiều khoản 370](#_bookmark1727)

[Tăng cường điều khoản must\_not 371](#_bookmark1734) ■ [Nênkhoản 373](#_bookmark1739)

[Điều khoản lọc 378](#_bookmark1751) ■ [Kết hợp tất cảcáiđiều khoản381](#_bookmark1757)

[Đã đặt têntruy vấn 382](#_bookmark1761)

* 1. [Không thay đổiđiểm384](#_bookmark1767)
  2. [Sự tăng cườngtruy vấn386](#_bookmark1778)
  3. [Sự phân ly tối đa(dis\_max) truy vấn388](#_bookmark1785)
  4. [Điểm số hàmtruy vấn389](#_bookmark1792)

[Điểm ngẫu nhiênchức năng 391](#_bookmark1796) ■ [Điểm số kịch bản](#_bookmark1800) [chức năng392](#_bookmark1800)  ■ [Hệ số giá trị trườngchức năng 393](#_bookmark1803)

[Kết hợp chức năngđiểm số 394](#_bookmark1807)

[Trình độ caotìm kiếm 397](#_bookmark1822)

[*12*](#_bookmark1822)

* 1. [Giới thiệu vị trítìm kiếm398](#_bookmark1829)

[Hộp giới hạntruy vấn 398](#_bookmark1831) ■ [Khoảng cách địa lýtruy vấn 399](#_bookmark1835)

[Hình dạng địa lýtruy vấn 400](#_bookmark1837)

* 1. [Dữ liệu không gian địa lýloại401](#_bookmark1841)

[Dữ liệu geo\_pointkiểu 401](#_bookmark1844) ■ [Hình dạng địa lý](#_bookmark1851) [dữ liệukiểu 402](#_bookmark1851)

* 1. [Không gian địa lýtruy vấn404](#_bookmark1862)
  2. [Hộp giới hạn địa lýtruy vấn405](#_bookmark1865)
  3. [Khoảng cách địa lýtruy vấn409](#_bookmark1879)
  4. [Hình dạng địa lýtruy vấn410](#_bookmark1883)
  5. [Hình dạngtruy vấn412](#_bookmark1893)

**thứ 16 MỤC LỤC**

* 1. [Khoảng cáchtruy vấn414](#_bookmark1903)

[*Dữ liệu mẫu 415*](#_bookmark1905)■ [*Khoảng cách đầu tiêntruy vấn 416*](#_bookmark1910)■ [*Khoảng cách gần*](#_bookmark1916)[*truy vấn417*](#_bookmark1916) ■ [*Khoảng cách trongtruy vấn418*](#_bookmark1922) ■ [*Khoảng cách*](#_bookmark1927)

[*truy vấn420*](#_bookmark1927)

* 1. [Chuyêntruy vấn421](#_bookmark1936)

[*Tính năng khoảng cáchtruy vấn 421*](#_bookmark1938)■ [*Đã ghimtruy vấn426*](#_bookmark1947) [*Càng nhiều càng tốttruy vấn 427*](#_bookmark1950)■ [*Sự thấm lọctruy vấn 428*](#_bookmark1956)

[***Tổng hợp 434***](#_bookmark1970)

[*13*](#_bookmark1970)

* 1. [Tổng quan435](#_bookmark1976)

[*Điểm cuối và cú pháp 435*](#_bookmark1978)■ [*Kết hợp tìm kiếm và*](#_bookmark1989)[*tổng hợp436*](#_bookmark1989)■ [*Nhiều và lồng nhautổng hợp 436*](#_bookmark1992)[*Bỏ quakết quả437*](#_bookmark1996)

* 1. [Hệ méttổng hợp437](#_bookmark2004)

[*Dữ liệu mẫu 438*](#_bookmark2009)■ [*Giá trị\_đếmmét 438*](#_bookmark2011)■ [*Cáctrung bình*](#_bookmark2019)[*mét440*](#_bookmark2019) ■ [*Tổng sốmét441*](#_bookmark2023) ■ [*Min và max*](#_bookmark2025)[*số liệu441*](#_bookmark2025) ■ [*Số liệu thống kê 442*](#_bookmark2031)■ [*Thống kê mở rộng*](#_bookmark2035)[*mét443*](#_bookmark2035) ■ [*Tính số lượngmét444*](#_bookmark2041)

* 1. [Xôtổng hợp445](#_bookmark2046)

[*Biểu đồ Histogram 446*](#_bookmark2049)■ [*Tổng hợp cấp độ con 450*](#_bookmark2065)■ [*Phong tục*](#_bookmark2070)[*tổng hợp phạm vi 451*](#_bookmark2070)■ [*Các điều khoảntổng hợp 453*](#_bookmark2078)

[*Các thuật ngữ đa dạngtổng hợp 454*](#_bookmark2082)

* 1. [Cha mẹ và anh chị em ruộttổng hợp455](#_bookmark2088)

[*Cha mẹtổng hợp 456*](#_bookmark2089)■ [*Tổng hợp anh chị em 457*](#_bookmark2092)

* 1. [Đường ống tập hợp458](#_bookmark2095)

[*Các loại tổng hợp đường ống 459*](#_bookmark2097)■ [*Vật mẫudữ liệu 459*](#_bookmark2099)[*Cú pháp cho tổng hợp đường ống 459*](#_bookmark2102)■ [*Đường ống có sẵn*](#_bookmark2108)[*tổng hợp461*](#_bookmark2108)■ [*Phần tử cha tổng tích lũy*](#_bookmark2132)[*tổng hợp 462*](#_bookmark2132)■ [*max\_bucket và min\_bucket*](#_bookmark2136)[*tổng hợp đường ống anh chị em 464*](#_bookmark2136)

[***Quản lý467***](#_bookmark2142)

[*14*](#_bookmark2142)

* 1. [Mở rộng quy môcụm468](#_bookmark2144)

[*Thêm các nút vào cụm 469*](#_bookmark2146)■ [*Cụmsức khỏe 470*](#_bookmark2152)[*Tăng đọcthông lượng 473*](#_bookmark2158)

* 1. [Nút giao tiếp474](#_bookmark2165)
  2. [Mảnh vỡkích thước475](#_bookmark2172)

[*Thiết lập mộtchỉ số475*](#_bookmark2174) ■ [*Thiết lập nhiều*](#_bookmark2177)[*chỉ số 476*](#_bookmark2177)

**thứ 17**

* 1. [Ảnh chụp nhanh477](#_bookmark2183)

[*Nhậnbắt đầu 478*](#_bookmark2185)■ [*Đăng ký kho lưu trữ ảnh chụp nhanh 479*](#_bookmark2189)[*Tạo ảnh chụp nhanh 481*](#_bookmark2200)■ [*Khôi phục ảnh chụp nhanh 482*](#_bookmark2207)[*Xóa bỏảnh chụp nhanh 483*](#_bookmark2211)■ [*Tự động hóa ảnh chụp nhanh 483*](#_bookmark2214)

* 1. [Trình độ caocấu hình488](#_bookmark2229)

[*Tệp cấu hình chính 488*](#_bookmark2233)■ [*Tùy chọn ghi nhật ký 488*](#_bookmark2236)[*Tùy chọn máy ảo Java 490*](#_bookmark2248)

* 1. [Cụmthạc sĩ491](#_bookmark2260)

[*Nút chính 491*](#_bookmark2263)■ [*Bậc thầybầu cử 491*](#_bookmark2266)■ [*Trạng thái cụm 492*](#_bookmark2272)[*Đủ số lượng 493*](#_bookmark2277) ■ [*Bộ não chia đôivấn đề 493*](#_bookmark2279)■ [*Tận tụy*](#_bookmark2283)[*nút chính 494*](#_bookmark2283)

[***Hiệu suất vàxử lý sự cố 497***](#_bookmark2289)

[*15*](#_bookmark2289)

* 1. [Tìm kiếm và tốc độvấn đề498](#_bookmark2291)

[*Phần cứng hiện đại 498*](#_bookmark2293)■ [*Tài liệumô hình hóa 499*](#_bookmark2300)[*Chọn loại từ khóa thay vì loại văn bản 501*](#_bookmark2308)

* 1. [Tốc độ chỉ sốvấn đề503](#_bookmark2312)

[*Hệ thống tạo racác định danh 503*](#_bookmark2314)■ [*Yêu cầu hàng loạt 503*](#_bookmark2316)[*Điều chỉnh tốc độ làm mới 503*](#_bookmark2319)

* 1. [Không ổn địnhcụm506](#_bookmark2327)

[*Cụm không phải làXANH LÁ506*](#_bookmark2329) ■ [*Chưa được chỉ địnhmảnh vỡ 507*](#_bookmark2332)[*Ngưỡng sử dụng đĩa 509*](#_bookmark2337)

* 1. [Mạch điện máy cắt511](#_bookmark2345)
  2. [Cuối cùngtừ512](#_bookmark2354)

[*phụ lụcACài đặt515*](#_bookmark2356) [*phụ lụcB Tiêu thụđường ống 523*](#_bookmark2381)[*phụ lụcKhách hàng530*](#_bookmark2409)

[*chỉ số541*](#_bookmark2464)

# lời nói đầu

Tôi là một người hâm mộ lớn của sách Manning In Action trong nhiều năm qua và chúng là một phần quan trọng trong sự nghiệp chuyên môn của tôi. Tôi thích cách các cuốn sách tập trung vào lời khuyên thực tế, hữu ích, thực hành về các công nghệ khác nhau mà tôi sử dụng, cả cho công việc và trong mã nguồn mở.

Cũng với tinh thần đó mà tôi đã ngồi xuống để viết Elasticsearch nhiều năm trước. Tôi bắt đầu tìm kiếm khi cố gắng xây dựng một ứng dụng công thức nấu ăn cho vợ tôi khi cô ấy đang học để trở thành đầu bếp. Tôi đã mã nguồn mở mã tôi viết để triển khai nó, do đó đã thực hiện bước đầu tiên của tôi vào thế giới mã nguồn mở. Và một vài năm sau, tôi ngồi xuống để viết Elasticsearch, cố gắng tạo ra một công cụ tìm kiếm hữu ích, thiết thực, dễ sử dụng—tìm kiếm trong hành động, nếu bạn muốn.

Quay trở lại năm 2023, tôi rất vui mừng khi thấy Elasticsearch đã thu hút được sự chú ý và có phần bổ sung riêng cho loạt bài In Action mà bạn hiện đang cầm trên tay hoặc xem dưới dạng kỹ thuật số. Tôi chắc rằng bạn sẽ thích đọc cuốn sách và tìm hiểu về Elasticsearch. Madhu đam mê cả tìm kiếm và Elasticsearch, và điều đó thể hiện ở chiều sâu và chiều rộng của cuốn sách này, giọng điệu nhiệt tình và các ví dụ thực tế.

Tôi hy vọng rằng sau khi tìm hiểu về Elasticsearch, bạn sẽ áp dụng những gì đã đọc và "thực hành". Suy cho cùng, tìm kiếm hiện diện ở khắp mọi nơi, trong mọi việc chúng ta làm, đó là lý do tại sao tôi đã yêu thích nó từ nhiều năm trước.

-SHAY BẨN DANH,NGƯỜI SÁNG LẬPVÀTÌM KIẾM LASTIC

**thứ 8**

# lời nói đầu

Vào cuối những năm 1990, tôi đã đến IIT Kharagpur ở Ấn Độ, nơi tôi đang theo đuổi bằng thạc sĩ và đồng thời đắm mình vào thế giới hấp dẫn của ngôn ngữ Java (chính xác là phiên bản 1!). Đó là một hành trình đơn độc, mặc dù tôi rất thích sử dụng ngôn ngữ hướng đối tượng lần đầu tiên, mỗi lần biên dịch mã lỗi đều báo hiệu một cuộc đấu tay đôi giữa tôi và thiết bị đầu cuối. Vào thời đó, không có IDE nào lạ mắt; chúng tôi nhập vào trình soạn thảo vi rồi biên dịch. (Có một IDE thô sơ có tên là Java WorkShop từ Sun Microsystems, nhưng nó không phải là không có nếp nhăn và lỗi.) Và không giống như thời đại hiện tại, tràn ngập các tài nguyên như Google, Stack Overflow và GitHub, không có con đường kỹ thuật số nào để thăm dò các giải pháp hoặc khám phá xem các lập trình viên khác có gặp phải những thách thức tương tự hay không. Sau đó, người bạn thân nhất của tôi là Amar, người đang học tại IISc, Bangalore, đã giới thiệu cho tôi một thực thể mới trong lĩnh vực tìm kiếm: Google.

Đối với một sinh viên trẻ ngây thơ, khái niệm tiếp cận một cộng đồng ảo

Những lập trình viên vật lộn với những vấn đề tương tự là một điều mới lạ. Vì vậy, tôi quyết định khám phá con đường mới này. Tôi nhớ mình đã bị hấp dẫn bởi sự đơn giản trong thiết kế của Google, nền trắng không gây khó chịu được tô điểm bằng các màu cơ bản, khi tôi điều hướng các trang để tìm kiếm câu trả lời. Phiên bản đầu tiên của công cụ tìm kiếm này còn lâu mới là công cụ tinh vi mà chúng ta biết ngày nay, nhưng nó là một ngọn hải đăng đáng khích lệ trong những thời điểm đầy thách thức đó. (Bất kỳ ai đã trải nghiệm Java 1.x đều có thể hiểu được những thử thách liên quan.) Khoảnh khắc đó đánh dấu một bước ngoặt quan trọng trong hành trình của tôi.

Trong khi khả năng tìm kiếm của Google là một cuộc cách mạng đối với nhiều người trong chúng ta, thì việc áp dụng nó trong các thiết lập của khách hàng và các tổ chức lại chậm hơn một chút. Các dự án mà tôi tham gia vào thời điểm đó có cơ sở dữ liệu đóng vai trò là xương sống cho các ứng dụng tìm kiếm. Mặc dù có chức năng, nhưng các thiết lập này vẫn còn kém xa so với những gì chúng ta hiện công nhận là khả năng của các công cụ tìm kiếm hiện đại. Các cơ chế tìm kiếm thì vụng về và khó sử dụng,

**xix**

**xx LỜI NÓI ĐẦU**

thiếu hiệu suất và đặt ra những thách thức về bảo trì. Tuy nhiên, sở thích của tôi đối với kiến ​​trúc hợp lý không bao giờ dao động, vẫn là ngọn hải đăng chỉ đường trong hành trình tìm kiếm các giải pháp hiệu quả và hiệu suất.

Hành trình của tôi vào thế giới Elasticsearch bắt đầu vào năm 2015, cùng thời điểm dữ liệu lớn trở nên phổ biến. Tôi đã làm việc trên Enterprise Java Beans (EJB), sử dụng các công cụ máy chủ ứng dụng như JBoss và WebLogic ưa chuộng kiến ​​trúc monolith, và tôi ngay lập tức bị thu hút bởi kiến ​​trúc đơn giản của Elasticsearch—đặc biệt là hỗ trợ không phụ thuộc vào ngôn ngữ lập trình, chức năng sẵn dùng, hiệu suất và tài liệu hướng dẫn tuyệt vời. Phải mất một vài năm tôi mới nắm bắt được toàn bộ tiềm năng của Elasticsearch và áp dụng toàn bộ.

Mặc dù Elasticsearch có tính đơn giản và khả năng mạnh mẽ, tôi nhận ra rằng đường cong học tập rất dốc và các cơ chế nội bộ phức tạp của nó đảm bảo việc điều hướng có hướng dẫn qua vô số tính năng và những thách thức vốn có. Việc hiểu được vô số tính năng và mê cung tài liệu đòi hỏi phải có sự hướng dẫn tận tình. Nhận thức này đã khiến tôi tiếp cận Andy Waldron tại Manning.

Việc sáng tác một cuốn sách là một nỗ lực thường đòi hỏi phải thức khuya, hy sinh những ngày cuối tuần và dành ngày nghỉ bên bàn viết. Đó là một nhiệm vụ to lớn mà không bao giờ có thể thành hiện thực nếu không có sự ủng hộ không ngừng nghỉ của gia đình. Và trong trường hợp này, đó là một hành trình kéo dài hai năm được đánh dấu bằng sự tập trung và quyết tâm không ngừng nghỉ để đưa cuốn sách này đến tay bạn.

Mặc dù tôi thường dè dặt về những thành tựu của mình, nhưng tôi không thể không thừa nhận công sức bỏ ra cho dự án cụ thể này. Nó xứng đáng được tự khen ngợi. Trong suốt hai năm qua, không một ngày nào bị che mờ bởi sự tự ti, và đó là một chiến công đáng để ăn mừng.

Bây giờ, khi bạn đọc Elasticsearch in Action, Phiên bản thứ hai, tôi cảm thấy vô cùng biết ơn và vui mừng. Tôi đánh giá cao việc bạn mua hàng và rất vui mừng khi bạn bắt đầu hành trình này vào thế giới Elasticsearch.

Khi dữ liệu lớn và điện toán đám mây phát triển với tốc độ chưa từng có trong thập kỷ qua, thì chiều sâu và chiều rộng của các khả năng của Elasticsearch cũng vậy. Tính liên quan của nó tiếp tục mở rộng, phản ánh nhu cầu và cơ hội đang phát triển của các bối cảnh công nghệ năng động này. Các tính năng tiên tiến của nó, bao gồm các trình phân tích đa ngôn ngữ, lưu trữ không gian địa lý và chuỗi thời gian, phát hiện bất thường bằng thuật toán học máy, phân tích đồ thị, tìm kiếm tự động hoàn thành và mờ, phân tích nguyên nhân gốc rễ bằng kho lưu trữ nhật ký, hình ảnh hóa dữ liệu phong phú, tổng hợp phức tạp và nhiều tính năng khác, khiến Elasticsearch trở thành công cụ không thể thiếu đối với hầu hết các tổ chức và doanh nghiệp.

Tôi thực lòng hy vọng bạn thích đọc cuốn sách này như tôi thích viết nó!

# lời cảm ơn

Một cuốn sách không bao giờ sinh ra từ hư không! Đó là hiện thân của sự tận tụy, kết quả của vô số giờ dành cho việc lập kế hoạch tỉ mỉ, sự tận tụy kiên định và nỗ lực không mệt mỏi. Quá trình này là một nỗ lực chung bao gồm sự hỗ trợ không ngừng và đầy yêu thương của gia đình tôi, tinh thần đồng đội và chuyên môn của đội ngũ nhân viên Manning, sự động viên của bạn bè và đồng nghiệp, và tất nhiên, sự quan tâm và hỗ trợ liên tục của những độc giả như bạn. Tất cả chúng tôi đã gắn bó với nhau trong hai năm, tận tụy và quyết tâm, tập trung và nhiệt huyết, với một mục tiêu: tạo ra Elasticsearch in Action. Và chúng tôi đã làm được!

Tôi phải gửi lời cảm ơn chân thành và sâu sắc nhất đến vợ tôi, Jeannette D'Souza, người đã là chỗ dựa vững chắc trong suốt hành trình này. Sự kiên nhẫn, thấu hiểu và động viên vô tận của cô ấy là ngọn hải đăng không lay chuyển trong những đêm dài và những khoảnh khắc khó khăn. Cô ấy không chỉ là người bạn đồng hành mà còn là trụ cột của sức mạnh và khả năng phục hồi, biến quá trình gian khổ này thành hành trình của tình yêu và sự quyết tâm. Vì điều này, và vì mọi thứ khác mà cô ấy có, tôi vô cùng biết ơn.

Tương tự như vậy, tôi xin gửi lời biết ơn chân thành đến con trai tôi, Josh, người có tình yêu thương và sự ủng hộ vô điều kiện, không nghi ngờ đã là nguồn sức mạnh trong những thời điểm đầy thử thách này. Sự hiểu biết của con (khi tôi bỏ lỡ mùa hè và ngày lễ) và sự động viên đã đưa cuốn sách này ra đời.

Lòng biết ơn sâu sắc của tôi đối với mẹ tôi, người đã sống cùng tôi trong nửa năm trong suốt quá trình này. Sự hiện diện chu đáo của bà thể hiện qua những cử chỉ yêu thương, từ việc phục vụ cà phê sáng sớm đến việc chuẩn bị bữa sáng thịnh soạn, các món mặn hấp dẫn và đồ ăn nhẹ thú vị. Sự hỗ trợ của bà thực sự là nguồn an ủi và sức mạnh.

Lòng biết ơn chân thành của tôi xin gửi đến hai người anh trai, cháu gái và gia đình trực hệ của tôi. Sự hiểu biết và hỗ trợ của họ đã tiếp thêm sức mạnh cho tôi ngay cả khi nghĩa vụ viết lách của tôi dẫn đến việc giao tiếp ít thường xuyên hơn. Sự động viên liên tục của họ

**xxi**

**xxii LỜI CẢM ƠN**

đóng vai trò như nhiên liệu thúc đẩy những nỗ lực của tôi, giúp động cơ sáng tạo của tôi hoạt động trơn tru.

Tôi phải bày tỏ lòng biết ơn sâu sắc nhất của mình đến Venkat Subramaniam. Vị thế đáng kính của ông với tư cách là một tác giả chỉ có thể bị vượt qua bởi sự sẵn lòng giúp đỡ tôi và những người khác của ông. Ông đã truyền đạt trí tuệ của mình cho các chương trước của tôi, cung cấp lời khuyên vô giá thực sự giúp kết nối với người đọc. Bất kể lịch trình của ông dày đặc như thế nào, Venkat luôn tìm được thời gian để hỗ trợ tôi. Sự hỗ trợ của ông là công cụ, và vì điều này, tôi không thể cảm ơn ông đủ.

Nếu không có sự hướng dẫn vô giá và sự hỗ trợ không ngừng nghỉ của Andy Waldron, biên tập viên của tôi tại Manning, cuốn sách này sẽ không bao giờ có thể từ khái niệm trở thành hiện thực. Chuyên môn và sự động viên của anh ấy đã đóng vai trò quan trọng trong hành trình đưa cuốn sách này đến với ánh sáng của ấn phẩm.

Xin gửi lời tri ân đặc biệt đến Ian Hough, biên tập viên phát triển của tôi, người có sự kiên nhẫn và sự nhạy bén trong biên tập không gì sánh bằng. Khả năng giám sát chiến lược của anh ấy trong việc lập kế hoạch cho từng chương, đảm bảo rằng tôi vẫn đi đúng hướng, là công cụ quan trọng trong việc định hình cuốn sách này. Ngay cả khi thời hạn bị trễ, sự hiểu biết của anh ấy vẫn đáng khen ngợi. Những đóng góp của anh ấy thực sự vô giá.

Biên tập viên phát triển bản thảo của tôi, Frances Buran, đã làm nên điều kỳ diệu trong mọi phần của cuốn sách. Cô ấy không ngừng cải thiện chất lượng viết, đảm bảo rằng ngôn ngữ không phải tiếng mẹ đẻ của tôi nghe hay trên trang.

Xin chân thành và đặc biệt cảm ơn biên tập viên Tiffany Taylor, người có chuyên môn về ngôn ngữ vô cùng quý giá trong việc đảm bảo tính chính xác và mạch lạc về mặt ngữ pháp trong suốt cuốn sách. Những nỗ lực tỉ mỉ của cô đã nâng cao đáng kể tính rõ ràng và dễ đọc của tác phẩm này. Cô ấy đã phát hiện ra những lỗi ngớ ngẩn và đáng xấu hổ của tôi ngay lập tức. Tiffany đã hoàn thành việc biên tập cuối cùng một cách hiệu quả và hiệu suất.

Tôi xin chân thành cảm ơn Melena Selic, Marina Matesic, Rebecca Rinehart, Aria Ducic và Susan Honeywell tại Manning đã làm cho cuốn sách này trở thành một sản phẩm chất lượng cao như vậy. Nếu không có sự giúp đỡ và hướng dẫn không ngừng của các bạn, cuốn sách này sẽ không thể ra đời. Tôi cũng xin chân thành cảm ơn đội ngũ sản xuất tại Manning đã làm việc chăm chỉ để tạo ra cuốn sách này.

Tôi xin cảm ơn Al Krinker, biên tập viên phát triển kỹ thuật của tôi, người đã xem xét từng chương với góc nhìn mới mẻ và góc nhìn mới mẻ. Al đã nhảy vào từng chương gần như ngay khi nó đến tay anh ấy, vì vậy tôi có thể xem xét phản hồi của anh ấy trong khi các chương vẫn còn mới trong tâm trí tôi.

Tôi cũng muốn cảm ơn Simon Hewitt và Simone Cafiero vì phản hồi kỹ thuật kịp thời và vì đã kiểm tra và thử nghiệm mã. Ý kiến ​​đóng góp của họ vô cùng quý giá.

Tôi cũng xin gửi lời cảm ơn đến một số người bạn và đồng nghiệp của tôi: Herodotos Kouk-kides, Semi, Jason Dynes và George Theofanous. Còn một số người khác mà tôi không thể nêu tên, nhưng họ đã đóng vai trò quan trọng trong hành trình của tôi! Cảm ơn các bạn từ tận đáy lòng!

Nếu có một nhóm mà tôi cảm thấy biết ơn nhất, thì chắc chắn đó là những người đánh giá và độc giả của tôi. Họ đã đóng vai trò quan trọng trong việc cải thiện chất lượng của cuốn sách này. Tôi xin gửi lời cảm ơn chân thành nhất đến những người đánh giá sau: Adam Wan, Alan Moffet, Alessandro

**LỜI CẢM ƠN xxiii**

Campeis, Andrei Mihai, Andrew Sacco, Bruno Sonnino, Dainius Jocas, Dan Kacenjar, Edward Ribeiro, Fernando Bernardino, Frans Oilinki, George Onofrei, Giampiero Granatella, Giovanni Costagliola, Hugo Figueiredo, Jaume Lopez, Jim Amrhein, Kent Spillner, Manuel R. Ciosici, Milorad Imbra, Dale S. Francis, Muneeb Shaikh, Paul Grebenc, Raymond Cheung, Richard Vaughan, Sai Gummaluri, Sayeef Rahim, Sergio Fernandez Gonzalez, Simone Cafiero, Simone Sguazza, Srihari Sridharan, Sumit K Singh, Vittorio Marino và William Jamir Silva. Những phản hồi vô giá của họ đã góp phần rất lớn vào việc định hình cuốn sách này thành nguồn tài liệu như ngày nay. Những hiểu biết và quan điểm của họ là hoàn toàn không thể thiếu.

# về cuốn sách này

#### Ai nên đọc cuốn sách này

Cuốn sách này sẽ là nguồn tài nguyên vô giá cho bất kỳ ai muốn hiểu sâu hơn về Elasticsearch và các ứng dụng thực tế của nó. Cụ thể, những điều sau đây sẽ có lợi khi đọc Elasticsearch in Action:

* Các nhà phát triển, kiến ​​trúc sư, nhà phân tích, nhà quản lý hoặc chủ sở hữu sản phẩm là người mới bắt đầu trong lĩnh vực Elasticsearch và muốn hiểu cách thức hoạt động cơ bản của nó
* Các nhà khoa học dữ liệu muốn triển khai Elasticsearch trong đường ống dữ liệu của họ để phân tích và xử lý dữ liệu theo thời gian thực
* Quản trị viên hệ thống duy trì cơ sở dữ liệu lớn và muốn sử dụng Elasticsearch để nâng cao hiệu quả truy xuất dữ liệu và hiệu suất hệ thống tổng thể
* Các chuyên gia tư vấn CNTT hoặc cố vấn kỹ thuật cần hiểu Elasticsearch để họ có thể giới thiệu nó trong các dự án của khách hàng và đưa ra các quyết định chiến lược về CNTT
* Các chủ doanh nghiệp am hiểu công nghệ muốn hiểu cách Elasticsearch có thể nâng cao hiệu quả hoạt động của họ hoặc cung cấp giá trị bổ sung cho khách hàng của họ
* Sinh viên và nhà nghiên cứu học thuật trong lĩnh vực khoa học máy tính, khoa học dữ liệu hoặc các lĩnh vực liên quan đang nghiên cứu công nghệ dữ liệu lớn và quan tâm đến việc tìm hiểu về công nghệ tìm kiếm
* Các cá nhân xử lý các tập dữ liệu lớn và muốn sử dụng Elasticsearch để nâng cao khả năng tìm kiếm của họ, bao gồm tìm kiếm toàn văn, tìm kiếm mờ, tìm kiếm cấp thuật ngữ và các tính năng tìm kiếm phức tạp khác
* Các kiến ​​trúc sư, nhà phát triển hoặc nhà phân tích Elasticsearch đầy tham vọng muốn thiết kế và phát triển các dịch vụ siêu nhỏ giao tiếp với các cụm Elasticsearch

**xxiv**

**VỀ ĐIỀU NÀYSÁCH xv**

#### Cuốn sách này được tổ chức như thế nào: Một lộ trình

Mặc dù cuốn sách này không được chia thành nhiều phần, các chương đều theo trình tự tuyến tính rõ ràng, bắt đầu bằng phần giới thiệu về Elasticsearch theo góc độ tính năng và kiến ​​trúc:

* Chương 1 bắt đầu hành trình khám phá thế giới tìm kiếm, lần theo các bước từ hệ thống cơ sở dữ liệu thô sơ đến các công cụ tìm kiếm tinh vi đang là chuẩn mực hiện nay. Chúng tôi tập trung vào Elasticsearch, một công cụ tìm kiếm hiện đại mạnh mẽ, đa năng đã định nghĩa lại khả năng của các chức năng tìm kiếm, đưa các tính năng riêng biệt, ứng dụng thực tế và sự áp dụng rộng rãi của nó lên hàng đầu.

Chúng tôi cũng hướng đến tiềm năng biến đổi của các công cụ trí tuệ nhân tạo nói chung. Chúng tôi xem xét các khả năng thú vị của các công nghệ như ChatGPT, khám phá cách chúng có thể định hình lại không gian tìm kiếm và định nghĩa lại tương tác của chúng ta với thông tin trong tương lai.

* Chương 2 đi sâu vào Elasticsearch, cung cấp kinh nghiệm thực hành với việc lập chỉ mục và truy xuất tài liệu bằng API tài liệu. Chúng tôi cũng thực hiện các truy vấn tìm kiếm dễ thương bằng API tìm kiếm. Chương này hướng dẫn bạn qua các tiêu chí tìm kiếm thiết yếu, từ khớp mẫu đến tìm kiếm cụm từ, sửa lỗi chính tả, kết quả phạm vi, tìm kiếm nhiều trường, v.v. Một cái nhìn thoáng qua về các truy vấn nâng cao làm phong phú thêm trải nghiệm học tập. Chương kết thúc bằng cách xem xét sắp xếp dữ liệu, phân trang kết quả, tô sáng và các chức năng ấn tượng khác giúp nâng cao khả năng tìm kiếm của người dùng.
* Chương 3 giải mã kiến ​​trúc của Elasticsearch, hướng dẫn bạn qua các thành phần nền tảng và các quy trình phức tạp cho phép tìm kiếm và lập chỉ mục. Khám phá này bao gồm các khái niệm cơ bản cung cấp năng lượng cho công cụ tìm kiếm, bao gồm chỉ mục đảo ngược, tính liên quan và phân tích văn bản. Chúng tôi cũng khám phá cụm và bản chất phân tán của máy chủ Elasticsearch.
* Chương 4 khám phá các lược đồ ánh xạ, kiểu dữ liệu và API ánh xạ, cung cấp tổng quan chi tiết về cách xử lý dữ liệu trong Elasticsearch. Chương này xem xét cách các lược đồ ánh xạ nâng cao độ chính xác và hiệu quả tìm kiếm, kiểm tra kỹ lưỡng ánh xạ động và rõ ràng. Khám phá mở rộng đến các kiểu dữ liệu cốt lõi, bao gồmchữ,từ khóa,ngày, Vàsố nguyên. Chương kết thúc vớicác kiểu dữ liệu nâng cao nhưđiểm địa lý,hình dạng địa lý,sự vật,tham gia,phẳngvà nhiều hơn nữa.
* Chương 5 cung cấp thảo luận toàn diện về API đơn và đa tài liệu và các hoạt động liên quan của chúng. Chương này cung cấp hiểu biết thực tế về lập chỉ mục, truy xuất, cập nhật và xóa tài liệu bằng các API này. Chương này cũng khám phá tính năng lập chỉ mục lại.
* Chương 6 tập trung vào các hoạt động lập chỉ mục bằng cách sử dụng API lập chỉ mục. Chương này cũng hướng dẫn bạn qua các cấu hình cơ bản của chỉ mục, bao gồm các thiết lập, ánh xạ và bí danh. Khám phá này cung cấp sự hiểu biết về

**xxvi VỀ CUỐN SÁCH NÀY**

tạo các chỉ mục tùy chỉnh phù hợp với các tình huống sản xuất. Chương này cũng xem xét cách làm việc với các mẫu chỉ mục, thảo luận về cơ chế của các mẫu chỉ mục và có thể cấu hình. Phần cuối cùng sẽ tìm hiểu về quản lý vòng đời chỉ mục.

* Chương 7 đưa chúng ta vào phân tích văn bản, xem xét cách toàn bộ văn bản được mã hóa và chuẩn hóa bằng cách sử dụng các mô-đun phân tích của Elasticsearch. Chúng ta xem xét cơ chế phân tích văn bản, khám phá các trình phân tích tích hợp nhưtiêu chuẩn,đơn giản,từ khóavà trình phân tích ngôn ngữ. Chương này sẽ cung cấp cho bạn kiến ​​thức để tạo trình phân tích tùy chỉnh.

Các chương từ 8 đến 13 dành riêng để tìm kiếm:

* Chương 8 đặt nền tảng để hiểu các nguyên tắc cơ bản của tìm kiếm, giải thích cơ chế xử lý yêu cầu tìm kiếm và tạo phản hồi. Chúng tôi giới thiệu hai loại tìm kiếm chính: yêu cầu URL và DSL truy vấn. Chúng tôi cũng xem xét các tính năng cắt ngang như tô sáng, sắp xếp, phân trang và các tính năng khác, cung cấp phần giới thiệu toàn diện về chức năng tìm kiếm của Elasticsearch.
* Chương 9 khám phá lĩnh vực truy vấn cấp thuật ngữ, được thiết kế riêng cho dữ liệu có cấu trúc. Chương này cung cấp một cuộc kiểm tra chi tiết về nhiều loại truy vấn cấp thuật ngữ, bao gồm phạm vi, tiền tố, ký tự đại diện và truy vấn mờ.
* Chương 10 xem xét các truy vấn toàn văn được thiết kế riêng để tìm kiếm dữ liệu phi cấu trúc. Chương này xem xét cơ chế sử dụng API tìm kiếm toàn văn, sử dụng nhiều loại truy vấn bao gồm truy vấn họ khớp, chuỗi truy vấn, truy vấn mờ và truy vấn chuỗi đơn giản, cùng nhiều loại khác.
* Chương 11 điều hướng thế giới phức tạp của các truy vấn hợp chất, làm nổi bật truy vấn Boolean như một công cụ đa năng để tạo các truy vấn tìm kiếm nâng cao. Chúng tôi xem xétkhi sử dụng các mệnh đề điều kiện nhưphải,phải\_không,nên, Vàlọcđể cấu trúc các truy vấn lá thành các truy vấn phức tạp hơn, hợp chất. Chương này kết thúc bằng một cuộc kiểm tra chi tiết về việc tăng cường vàđiểm số hằng sốtruy vấn.
* Chương 12 giới thiệu các truy vấn chuyên biệt, bao gồmkhoảng cách\_tính năng,máy lọc nước,nhiều hơn như thế này, Vàghim. Chương này xem xét những lợi ích độc đáo của từng loại truy vấn, chẳng hạn nhưkhoảng cách\_tính năngkhả năng của truy vấn để ưu tiên kết quả gần hơn với một vị trí nhất định vànhiều hơn như thế nàychức năng của truy vấn là tìm kiếm các tài liệu có giao diện tương tự.máy lọc nướctruy vấn, thông báo cho người dùng về các kết quả mới có sẵn, cũng được kiểm tra chi tiết.
* Chương 13 cung cấp một bài kiểm tra chi tiết về tổng hợp. Chúng tôi khám phá tổng hợp số liệu, tạo ra các số liệu thống kê như tổng, trung bình, tối thiểu, tối đa, lượt truy cập cao nhất và chế độ. Chương này cũng nêu bật việc sử dụng tổng hợp nhóm trong việc thu thập dữ liệu tổng hợp thành các tập hợp nhóm. Chúng tôi cũng xem xét tổng hợp đường ống, cung cấp phân tích thống kê nâng cao như đạo hàm và trung bình động.

**VỀ ĐIỀU NÀYSÁCH xxvii**

Hai chương cuối khép lại cuốn sách bằng cách tập trung vào quản lý và hiệu suất:

* + Chương 14 xem xét khía cạnh quản trị của việc sản xuất Elasticsearch. Điều này bao gồm hiểu cách mở rộng cụm dưới nhiều tải khác nhau, giao tiếp giữa các nút và kích thước phân đoạn. Chương này cũng khám phá khái niệm quan trọng về snapshotting, cung cấp các ví dụ thực tế về việc chụp nhanh và truy xuất dữ liệu từ đó khi cần. Các cấu hình nâng cao và khái niệm cụm chủ cũng được xem xét kỹ lưỡng.
  + Chương 15 đi sâu vào việc khắc phục sự cố cụm Elasticsearch hoạt động kém hoặc có vấn đề. Chương này xem xét các nguyên nhân phổ biến, chẳng hạn như tắc nghẽn tìm kiếm và tốc độ, cụm không ổn định và không lành mạnh, và bộ ngắt mạch, cùng nhiều nguyên nhân khác. Chương này sẽ trang bị cho bạn kiến ​​thức để chẩn đoán và giải quyết các vấn đề về hiệu suất, đảm bảo cụm Elasticsearch của bạn chạy trơn tru và hiệu quả.

Cuốn sách cũng có ba phần phụ lục:

* + Phụ lục A là hướng dẫn thực tế về cách cài đặt Elasticsearch và Kibana trong môi trường cục bộ của bạn.
  + Phụ lục B xem xét các đường ống thu thập dữ liệu, một thành phần quan trọng của quá trình xử lý dữ liệu trước trong Elasticsearch, cũng như cách cấu hình và sử dụng chúng trong nhiều tình huống khác nhau.
  + Phụ lục C trình bày khả năng tương tác của Elasticsearch với nhiều ngôn ngữ lập trình khác nhau thông qua các máy khách như Java, Python, JavaScript, C# và các ngôn ngữ khác, đồng thời đưa ra các ví dụ và phương pháp hay nhất.

#### Về mã

Một trong những mục tiêu chính của cuốn sách này là cung cấp trải nghiệm thực hành liền mạch bằng cách đưa vào mã dễ thực thi. Sau nhiều lần lặp lại, một quyết định đã được đưa ra là lưu trữ tất cả các truy vấn, được viết và thực thi trên Kibana, trong một tệp văn bản trên GitHub. Các truy vấn này được ghi lại dưới dạng mã JSON dựa trên Query DSL. Mục đích là cung cấp một quy trình đơn giản, trong đó bạn có thể sao chép tệp văn bản từ GitHub và dán vào ứng dụng Kibana Dev Tools của mình để thực thi ngay lập tức.

Để hỗ trợ thêm cho hành trình học tập của bạn, các tệp dữ liệu mẫu và, khi cần thiết, các bản đồ cho các chỉ mục này được cung cấp trong một thư mục tập dữ liệu chuyên dụng. Cách tiếp cận này đảm bảo trải nghiệm thực tế, thân thiện với người học, cho phép bạn tương tác trực tiếp với tài liệu và áp dụng kiến ​​thức mới học được.

Mã nguồn có sẵn trên GitHub ([https://github.com/madhusudhan](https://github.com/madhusudhankonda/elasticsearch-in-action) [konda/elasticsearch-trong-hành-động](https://github.com/madhusudhankonda/elasticsearch-in-action)) và trang web của cuốn sách ([https://www.manning.com/](https://www.manning.com/books/elasticsearch-in-action-second-edition) [sách/elasticsearch-in-action-second-edition](https://www.manning.com/books/elasticsearch-in-action-second-edition)). Các thư mục như sau:

* + [*kibana\_scripts*](https://github.com/madhusudhankonda/elasticsearch-in-action/tree/b8fbe9068dfde8dc0912cb4330e51e33745a0da8/kibana_scripts)—Truy vấn các tập lệnh DSL cho từng chương.
  + [*bộ dữ liệu*](https://github.com/madhusudhankonda/elasticsearch-in-action/tree/b8fbe9068dfde8dc0912cb4330e51e33745a0da8/datasets)—Các bản đồ và bộ dữ liệu mẫu cần thiết cho các chương sách.
  + [*mã số*](https://github.com/madhusudhankonda/elasticsearch-in-action/tree/b8fbe9068dfde8dc0912cb4330e51e33745a0da8/code)—Mã Java và Python.

**xxviii VỀ CUỐN SÁCH NÀY**

* [*người đóng tàu*](https://github.com/madhusudhankonda/elasticsearch-in-action/tree/b8fbe9068dfde8dc0912cb4330e51e33745a0da8/docker)—Docker tập tin để chạy các dịch vụ trên môi trường cục bộ. Ví dụ,[elasticsearch-docker-8-6-2.yml](https://github.com/madhusudhankonda/elasticsearch-in-action/blob/main/docker/elasticsearch-docker-8-6-2.yml) lưu trữ hai dịch vụ: Elasticsearch và Kibana. Vì vậykhi bạn thực hiệndocker-compose lênlệnh này sẽ khởi động cả Elasticsearch và Kibana trong vùng chứa Docker.
* [*phụ lục*](https://github.com/madhusudhankonda/elasticsearch-in-action/tree/0b3333014b072e9e64d995c842515cb4eea14133/appendices)—Với tốc độ phát triển của Elasticsearch, cuốn sách này sẽ cần được cập nhật. Các tính năng mới sẽ được cung cấp dưới dạng phụ lục của cuốn sách trên GitHub. Tôi sẽ thêm và sửa đổi nội dung khi có bản phát hành mới.

Các bản phát hành Elasticsearch diễn ra khá thường xuyên—chúng tôi đang ở phiên bản 7.x khi tôi bắt đầu viết cuốn sách này, và phiên bản là 8.7 khi cuốn sách đang chuẩn bị được in. Và tôi mong đợi một vài bản phát hành lớn và nhỏ hơn nữa từ Elastic vào thời điểm bạn đọc bài viết này! Sẽ là một nhiệm vụ rất lớn để cập nhật cơ sở mã bất cứ khi nào có bản phát hành mới. Tôi sẽ cố gắng cập nhật mã, nhưng tôi cũng sẽ rất vui khi những người đóng góp duy trì cơ sở mã. Vì vậy, vui lòng liên hệ với tôi nếu bạn muốn trở thành người đóng góp cho dự án này.

Cuốn sách này chứa nhiều ví dụ về mã nguồn, cả trong danh sách được đánh số và theo dòng văn bản thông thường. Trong cả hai trường hợp, mã nguồn được định dạng theo chiều rộng cố địnhphông chữ như thế này để phân biệt nó với văn bản thông thường.

Trong nhiều trường hợp, mã nguồn gốc đã được định dạng lại; chúng tôi đã thêm ngắt dòng và chỉnh sửa thụt lề để phù hợp với không gian trang có sẵn trong sách. Trong một số trường hợp hiếm hoi, ngay cả điều này cũng không đủ và danh sách bao gồm các dấu tiếp tục dòng (➥). Ngoài ra, các bình luận trong mã nguồn thường bị xóa khỏi danh sách khi mã được mô tả trong văn bản. Chú thích mã đi kèm với nhiều danh sách, làm nổi bật các khái niệm quan trọng.

#### Diễn đàn thảo luận liveBook

Mua Elasticsearch in Action, Phiên bản thứ hai bao gồm quyền truy cập miễn phí vào liveBook, nền tảng đọc trực tuyến của Manning. Sử dụng các tính năng thảo luận độc quyền của liveBook, bạn có thể đính kèm bình luận vào sách trên toàn cầu hoặc vào các phần hoặc đoạn văn cụ thể. Thật dễ dàng để tự ghi chú, hỏi và trả lời các câu hỏi kỹ thuật và nhận trợ giúp từ tác giả và những người dùng khác. Để truy cập diễn đàn, hãy truy cập[https://livebook](https://livebook.manning.com/book/elasticsearch-in-action-second-edition)

[.manning.com/book/elasticsearch-in-action-second-edition](https://livebook.manning.com/book/elasticsearch-in-action-second-edition). Bạn cũng có thể tìm hiểu thêm về diễn đàn Manning và các quy tắc ứng xử tại[https://livebook.manning.com/](https://livebook.manning.com/discussion) [cuộc thảo luận](https://livebook.manning.com/discussion).

Cam kết của Manning đối với độc giả của chúng tôi là cung cấp một địa điểm nơi có thể diễn ra cuộc đối thoại có ý nghĩa giữa từng độc giả và giữa độc giả và tác giả. Đây không phải là cam kết về bất kỳ mức độ tham gia cụ thể nào từ phía tác giả, người mà sự đóng góp của họ cho diễn đàn vẫn là tự nguyện (và không được trả công). Chúng tôi đề xuất bạn thử hỏi tác giả một số câu hỏi đầy thử thách để tránh mất hứng thú! Diễn đàn và kho lưu trữ các cuộc thảo luận trước đây sẽ có thể truy cập được từ trang web của nhà xuất bản miễn là cuốn sách vẫn được in.

# về tác giả

**MADHUSUDHAN KONDA**là một chuyên gia công nghệ dày dạn kinh nghiệm với cam kết không ngừng nghỉ trong việc đơn giản hóa những điều phức tạp, định hướng bức tranh toàn cảnh và dấn thân vào những ranh giới mới của ngôn ngữ lập trình và các khuôn khổ tiên tiến. Niềm đam mê công nghệ của anh không chỉ là một nghề nghiệp mà còn là hành trình khám phá và học hỏi suốt đời. Anh phát triển mạnh mẽ trong nghệ thuật chuyển đổi các vấn đề phức tạp thành các giải pháp đơn giản hơn, dễ quản lý hơn, cung cấp một kim chỉ nam rõ ràng trong bối cảnh công nghệ luôn thay đổi.

Trong suốt 25 năm sự nghiệp của mình, Madhu đã đảm nhiệm nhiều vai trò, bao gồm kiến ​​trúc sư giải pháp, kỹ sư trưởng/kỹ sư chính và nhiều vai trò khác. Tuy nhiên, vai trò của ông luôn được nhấn mạnh bởi mong muốn mãnh liệt chia sẻ kiến ​​thức và bồi dưỡng cho các đồng nghiệp hiểu biết về ngôn ngữ lập trình, khuôn khổ và công nghệ mới nổi.

Chuyên môn của Madhu đóng vai trò quan trọng trong việc thiết kế và cung cấp các giải pháp chất lượng cao cho nhiều khách hàng, từ các ngân hàng như Credit Suisse, UBS, Mizuho, ​​Deutsche Bank và Halifax cho đến các công ty hàng đầu về năng lượng và hàng không như British Petroleum và British Airways, cùng nhiều công ty khác.

Năng lực của ông không chỉ dừng lại ở việc lãnh đạo và triển khai các dự án phần mềm từ khi bắt đầu đến khi hoàn thành và thiết kế các giải pháp cho các vấn đề kinh doanh phức tạp. Madhu là một nhà chiến lược và người có tầm nhìn xa, được biết đến với sự thành thạo trong việc xây dựng lộ trình chiến lược, kiến ​​trúc tiết kiệm chi phí và thiết kế sản phẩm. Phong cách lãnh đạo của ông kết hợp giữa cố vấn với tư duy lãnh đạo, luôn vượt qua ranh giới và truyền cảm hứng cho các nhóm để đạt được tiềm năng của họ. Ông tự hào trong việc giảng dạy và đào tạo các chuyên gia từ người mới bắt đầu đến người cao cấp cũng như cố vấn và hướng dẫn những người mới vào nghề.

**xxix**

**xxx VỀ TÁC GIẢ**

Ngoài sự nghiệp ấn tượng của mình, Madhu còn là một tác giả nổi tiếng. Các cuốn sách và khóa học video của ông về Java, Spring và hệ sinh thái Hibernate đã được đón nhận nồng nhiệt, càng làm nổi bật cam kết của ông trong việc thúc đẩy văn hóa học tập và khám phá trong thế giới công nghệ. Ông là một blogger nhiệt tình, luôn cố gắng viết những bài viết sâu sắc vượt ra ngoài công nghệ, đào sâu vào lĩnh vực quan trọng của các kỹ năng mềm của kỹ sư.

Trong quá trình theo đuổi sự rõ ràng và súc tích, Madhu kiên trì nỗ lực chắt lọc các khái niệm công nghệ phức tạp thành nội dung dễ hiểu. Triết lý của ông tập trung vào việc đơn giản hóa các ý tưởng phức tạp đến mức mà ngay cả một đứa trẻ 10 tuổi cũng có thể hiểu được, qua đó giúp công nghệ tiên tiến dễ tiếp cận và dễ hiểu với tất cả mọi người.

# về hình minh họa bìa

Hình ảnh trên trang bìa của Elasticsearch in Action, Phiên bản thứ hai là “Một người đàn ông đến từ Croatia”, được lấy từ album trang phục truyền thống của Croatia từ giữa thế kỷ XIX của Nikola Arsenovic.

Vào thời đó, thật dễ dàng để nhận ra nơi mọi người sống và nghề nghiệp hoặc địa vị xã hội của họ chỉ bằng cách ăn mặc. Manning tôn vinh sự sáng tạo và chủ động của ngành kinh doanh máy tính bằng bìa sách dựa trên sự đa dạng phong phú của nền văn hóa khu vực cách đây nhiều thế kỷ, được tái hiện lại bằng hình ảnh từ các bộ sưu tập như thế này.

**xxxi**

**xxxii VỀ HÌNH ẢNH BÌA**

*Tổng quan*

***Chương này bao gồm***

* Thiết lập bối cảnh cho các công cụ tìm kiếm hiện đại
* Giới thiệu Elasticsearch
* Hiểu các lĩnh vực cốt lõi, trường hợp sử dụng và các tính năng nổi bật của Elasticsearch
* Elastic Stack: Beats, Logstash, Elasticsearch và Kibana

Sự bùng nổ của dữ liệu trong những năm gần đây đã dẫn đến một chuẩn mực mới về các tiêu chuẩn mong đợi của chức năng tìm kiếm và phân tích. Khi các tổ chức tích lũy dữ liệu, khả năng tìm ra “kim trong đống cỏ khô” là một điều cần thiết tối quan trọng. Ngoài tìm kiếm, khả năng thu nhỏ và tổng hợp dữ liệu bằng chức năng phân tích đã trở thành một yêu cầu bắt buộc đối với các tổ chức. Thập kỷ qua đã chứng kiến ​​sự áp dụng theo cấp số nhân của các công cụ tìm kiếm và phân tích hiện đại. Một trong những công cụ tìm kiếm hiện đại như vậy là Elasticsearch.

Elasticsearch là một công cụ tìm kiếm và phân tích phân tán mã nguồn mở mạnh mẽ và phổ biến. Nó được xây dựng trên thư viện Apache Lucene và có thể thực hiện tìm kiếm và phân tích gần như thời gian thực trên dữ liệu có cấu trúc và không có cấu trúc. Nó được thiết kế để xử lý hiệu quả lượng dữ liệu lớn.

**1**

**2 CPHẦN1*Tổng quan***

Elasticsearch đã đi một chặng đường dài trong việc cho phép các tổ chức sử dụng các tính năng mạnh mẽ của mình trong không gian tìm kiếm và phân tích. Ngoài các trường hợp sử dụng tìm kiếm và phân tích, nó còn được sử dụng cho phân tích nhật ký ứng dụng và cơ sở hạ tầng, bảo mật doanh nghiệp và phát hiện mối đe dọa, hiệu suất và giám sát ứng dụng, kho dữ liệu phân tán, v.v.

Trong chương này, chúng tôi sẽ xem xét không gian tìm kiếm nói chung và lướt qua quá trình phát triển của tìm kiếm, từ tìm kiếm dựa trên cơ sở dữ liệu truyền thống đến các công cụ tìm kiếm hiện đại hiện nay và nhiều tính năng tiện lợi của chúng. Trong quá trình này, chúng tôi giới thiệu Elastic-search, công cụ tìm kiếm mã nguồn mở cực nhanh và xem xét các tính năng, trường hợp sử dụng và sự chấp nhận của khách hàng.

Chúng tôi cũng xem nhanh cách các công cụ trí tuệ nhân tạo (AI) tạo ra đang bắt đầu làm sáng tỏ và phá vỡ không gian tìm kiếm. Với sự ra đời của ChatGPT, một cuộc đua để nắm bắt AI và trở thành người dẫn đầu trong không gian tìm kiếm đã bắt đầu. Chúng tôi dành một phần để giới thiệu những người chơi hiện tại và khám phá tương lai của các công cụ tìm kiếm do AI dẫn đầu.

#### Thế nào là một công cụ tìm kiếm tốt?

Hãy dành chút thời gian để xem xét điều gì làm cho một công cụ tìm kiếm trở nên "tốt" xét về mặt trải nghiệm hàng ngày. Để giúp hình dung điều này, hãy xem xét một trải nghiệm của tôi với một công cụ tìm kiếm tệ.

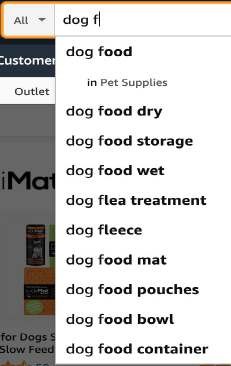
Gần đây gia đình tôi đã nhận nuôi một chú chó con, Milly (đó là cô ấy, trong ảnh!). Vì chúng tôi là chủ sở hữu lần đầu của một chú chó con, chúng tôi bắt đầu tìm kiếm trực tuyến về thức ăn cho chó. Tôi đã duyệt trang web của cửa hàng tạp hóa ưa thích của mình, nhưng thật đáng thất vọng, kết quả tìm kiếm không phải là thứ tôi đang tìm kiếm. Danh sách kết quả bao gồm "túi đựng phân" (điều cuối cùng bạn mong đợi khi tìm kiếm thức ăn cho chó) và các sản phẩm không liên quan khác. Trang web cũng không có bộ lọc, menu thả xuống hoặc bộ chọn phạm vi giá; đó chỉ là một trang đơn giản hiển thị kết quả tìm kiếm, với chức năng phân trang vụng về được bật.

Không hài lòng với tìm kiếm của người bán hàng tạp hóa hiện tại (sự tò mò đã chiếm lấy tôi—tôi háo hức muốn tìm hiểu cách các công cụ tìm kiếm khác được triển khai), tôi đã tìm kiếm ở những nơi khác

cửa hàng tạp hóa. Một trang web giới thiệu dây nịt cho thú cưng, trong khi những trang web khác thì không có gì đặc sắc, bao gồm cả một tìm kiếm cho tôi thấy hộp đựng đồ ăn trưa của em bé!

Tôi không chỉ nhận được kết quả tìm kiếm kém mà các công cụ tìm kiếm đằng sau những siêu thị này cũng không đưa ra gợi ý khi tôi nhập; họ cũng không sửa thông tin tôi nhập khi tôi viết sai từ dog food thành dig fod (chúng tôi được Google chiều chuộng khi nói đến các gợi ý và sửa lỗi—chúng tôi mong đợi mọi công cụ tìm kiếm đều có các tính năng tương tự như Google). Hầu hết không thèm gợi ý các lựa chọn thay thế hoặc các mặt hàng tương tự. Một số kết quả không liên quan (tức là không liên quan dựa trên mức độ liên quan—mặc dù điều này có thể được tha thứ, vì không phải tất cả các công cụ tìm kiếm đều được mong đợi sẽ đưa ra kết quả có liên quan). Một người bán tạp hóa đã trả về 2.400 kết quả cho một tìm kiếm đơn giản!

* 1. ***Tìm kiếm là cái mớiBình thường* 3**

Không hài lòng với kết quả từ nhiều siêu thị khác nhau, tôi đã đến Amazon, một cửa hàng trực tuyến phổ biến, nơi tôi tìm thấy một công cụ tìm kiếm tốt. Ngay khi tôi bắt đầu nhập dog, một menu thả xuống cho tôi thấy các gợi ý (xem hình bên cạnh đoạn văn này). Theo mặc định, tìm kiếm ban đầu trên Amazon sẽ trả về các kết quả có liên quan (khá gần với những gì chúng ta đang tìm kiếm). Sử dụng tùy chọn Sắp xếp theo Nổi bật, chúng ta có thể thay đổi thứ tự sắp xếp (giá thấp đến cao hoặc ngược lại, v.v.) nếu cần. Sau khi thực hiện tìm kiếm ban đầu, chúng ta cũng có thể đi sâu vào các danh mục khác bằng cách chọn bộ phận, đánh giá trung bình của khách hàng, phạm vi giá, v.v. Tôi thậm chí còn thử viết sai chính tả: dig food. Amazon hỏi tôi có phải ý tôi là thức ăn cho chó không. Thông minh, phải không?

Trong thế giới kỹ thuật số hiện nay, tìm kiếm đang là tâm điểm chú ý. Các tổ chức đang áp dụng tìm kiếm mà không cần suy nghĩ thêm vì họ hiểu giá trị kinh doanh mà công cụ tìm kiếm mang lại và các trường hợp sử dụng đa dạng mà nó có thể giải quyết. Trong phần tiếp theo, chúng ta sẽ khám phá sự tăng trưởng theo cấp số nhân của công cụ tìm kiếm và cách công nghệ đã cho phép tạo ra các giải pháp tìm kiếm tiên tiến.

#### Tìm kiếm là chuẩn mực mới

Với sự tăng trưởng theo cấp số nhân của dữ liệu (terabyte đến petabyte đến exabyte), nhu cầu về một công cụ cho phép tìm kiếm thành công trong các tình huống như mò kim đáy bể là điều bắt buộc. Những gì từng được coi là tìm kiếm đơn giản giờ đây là chức năng cần thiết cho hầu hết các bộ công cụ sinh tồn của các tổ chức. Các tổ chức được kỳ vọng sẽ cung cấp chức năng tìm kiếm theo mặc định để khách hàng có thể nhập vào thanh tìm kiếm hoặc điều hướng tìm kiếm chi tiết để tìm thấy những gì họ cần trong nháy mắt.

Ngày càng khó tìm thấy các trang web và ứng dụng không có thanh tìm kiếm khiêm tốn với một kính lúp nhỏ. Cung cấp tìm kiếm đầy đủ là một lợi thế cạnh tranh.

Ngày nay, các công cụ tìm kiếm hiện đại luôn hướng đến tốc độ và tính liên quan cũng như cung cấp chức năng nâng cao được đóng gói trong một bộ tính năng kinh doanh và kỹ thuật phong phú. Elasticsearch là một trong những công cụ tìm kiếm hiện đại như vậy, bao gồm tìm kiếm và phân tích với tốc độ và hiệu suất là cốt lõi. Khi làm việc với các công cụ tìm kiếm như Elasticsearch, bạn sẽ bắt gặp dữ liệu và các biến thể tìm kiếm: dữ liệu có cấu trúc và không có cấu trúc cùng các tìm kiếm tương ứng của chúng. Điều quan trọng là phải quen thuộc với các loại dữ liệu này để hiểu được bối cảnh tìm kiếm. Chúng ta sẽ xem xét chúng một cách ngắn gọn trong phần tiếp theo.

* + 1. Dữ liệu có cấu trúc so với dữ liệu không có cấu trúc (toàn văn)

Dữ liệu chủ yếu có hai loại: có cấu trúc và không có cấu trúc. Sự khác biệt cơ bản giữa hai loại này là cách dữ liệu được lưu trữ và phân tích. Dữ liệu có cấu trúc tuân theo một lược đồ/mô hình được xác định trước, trong khi dữ liệu không có cấu trúc là dạng tự do, không có tổ chức và không có lược đồ.

**4 CPHẦN1*Tổng quan***

**SDỮ LIỆU CẤU TRÚC**

Dữ liệu có cấu trúc được tổ chức rất tốt, có hình dạng và định dạng xác định, và phù hợp với các mẫu kiểu dữ liệu được xác định trước. Dữ liệu tuân theo một lược đồ được xác định và dễ tìm kiếm vì được tổ chức tốt. Dữ liệu trong cơ sở dữ liệu được coi là dữ liệu có cấu trúc vì trước khi được lưu trữ trong cơ sở dữ liệu, dữ liệu được mong đợi sẽ tuân theo một lược đồ nghiêm ngặt. Ví dụ, dữ liệu biểu diễn ngày tháng, số hoặc Boolean phải có định dạng cụ thể.

Truy vấn dữ liệu có cấu trúc trả về kết quả khớp chính xác. Nghĩa là, chúng ta quan tâm đến việc tìm các tài liệu khớp với tiêu chí tìm kiếm—nhưng không quan tâm đến mức độ khớp của chúng. Các kết quả tìm kiếm như vậy là nhị phân: chúng ta có kết quả hoặc không có kết quả—không có kết quả "có thể". Ví dụ, không hợp lý khi mong đợi "có thể" các chuyến bay bị hủy khi tìm kiếm "các chuyến bay bị hủy trong tháng qua". Có thể không có hoặc nhiều hơn, nhưng tìm kiếm không nên trả về kết quả "gần khớp với các chuyến bay bị hủy".

Chúng tôi không lo lắng về việc các tài liệu khớp nhau như thế nào—chỉ lo rằng chúng khớp nhau. Do đó, không có điểm liên quan (một số dương được đưa ra cho mỗi kết quả cho biết kết quả khớp với truy vấn chặt chẽ như thế nào) được đính kèm vào các kết quả. Tìm kiếm cơ sở dữ liệu truyền thống là loại này: lấy tất cả các chuyến bay đã bị hủy trong tháng trước, sách bán chạy nhất hàng tuần, hoạt động của người dùng đã đăng nhập, v.v.

**SỰ ĐỊNH NGHĨA***Sự liên quan*đề cập đến mức độ mà kết quả của công cụ tìm kiếm khớp với truy vấn của người dùng. Đây là cơ chế cho biết kết quả khớp với truy vấn gốc chặt chẽ như thế nào. Công cụ tìm kiếm sử dụng thuật toán liên quan để xác định tài liệu nào có liên quan chặt chẽ với truy vấn của người dùng (tức là mức độ liên quan của chúng) và tạo ra một số dương gọi là điểm liên quan cho mỗi kết quả dựa trên mức độ kết quả khớp với truy vấn. Lần tới khi bạn tìm kiếm trên Google, hãy xem kỹ kết quả tìm kiếm: các kết quả hàng đầu có liên quan rất chặt chẽ với những gì bạn đang tìm kiếm và do đó chúng ta có thể nói rằng chúng có liên quan hơn các mục dưới cùng trong danh sách kết quả. Google tự động gán điểm liên quan cho mỗi kết quả và rất có thể sắp xếp chúng theo điểm này: điểm càng cao thì kết quả càng có liên quan và do đó, khả năng kết quả ở đầu trang càng cao.

**TRONGDỮ LIỆU CÓ CẤU TRÚC**

Ngược lại, dữ liệu phi cấu trúc là dữ liệu không có tổ chức và không tuân theo bất kỳ sơ đồ hay định dạng nào. Dữ liệu phi cấu trúc không có cấu trúc được xác định trước. Dữ liệu phi cấu trúc là nền tảng của hầu hết các công cụ tìm kiếm hiện đại. Ví dụ bao gồm các bài đăng trên blog, bài báo nghiên cứu, email, PDF, tệp âm thanh và video, v.v.

**GHI CHÚ**Ngoài dữ liệu có cấu trúc và không có cấu trúc, còn có một loại khác: dữ liệu bán cấu trúc. Dữ liệu này nằm giữa dữ liệu có cấu trúc và không có cấu trúc. Nó không gì khác ngoài dữ liệu không có cấu trúc với một số siêu dữ liệu mô tả nó.

Đối với dữ liệu không có cấu trúc, Elasticsearch cung cấp khả năng tìm kiếm toàn văn bản cho phép chúng ta tìm kiếm các thuật ngữ hoặc cụm từ cụ thể trong một lượng lớn văn bản không có cấu trúc. Các truy vấn toàn văn (không có cấu trúc) cố gắng tìm các kết quả có liên quan đến truy vấn. Nghĩa là, Elasticsearch tìm kiếm tất cả các tài liệu phù hợp nhất với truy vấn. Ví dụ, nếu

* 1. ***Tìm kiếm là cái mớiBình thường* 5**

Khi người dùng tìm kiếm từ khóa vắc-xin, công cụ tìm kiếm không chỉ tìm kiếm các tài liệu liên quan đến tiêm chủng mà còn đưa ra các tài liệu có liên quan về tiêm chủng, tiêm chích và các thuật ngữ khác liên quan đến vắc-xin.

Elasticsearch sử dụng thuật toán tương tự để tạo điểm liên quan cho các truy vấn toàn văn. Điểm là số dấu phẩy động dương được đính kèm vào kết quả, với tài liệu có điểm cao nhất cho biết mức độ liên quan nhiều hơn đến tiêu chí truy vấn.

Elasticsearch xử lý hiệu quả cả dữ liệu có cấu trúc và không có cấu trúc. Một trong những tính năng chính của nó là khả năng lập chỉ mục và tìm kiếm cả dữ liệu có cấu trúc và không có cấu trúc trong cùng một chỉ mục. Điều này cho phép chúng ta tìm kiếm và phân tích cả hai loại dữ liệu cùng nhau và có được những hiểu biết khó có thể có được nếu không có nó.

* + 1. Tìm kiếm được hỗ trợ bởi cơ sở dữ liệu

Tìm kiếm theo kiểu cũ chủ yếu dựa trên cơ sở dữ liệu quan hệ truyền thống. Các công cụ tìm kiếm cũ dựa trên kiến ​​trúc nhiều lớp được triển khai trong các ứng dụng nhiều lớp, như thể hiện trong hình 1.1.

Truy vấn tìm kiếm Phản hồi tìm kiếm

Lớp cơ sở dữ liệu

Lớp kinh doanh

Lớp trình bày

Giao diện người dùng

**Người dùng gọi mộttìm kiếm**

**truy vấn trên UI/ứng dụng.**

**Lớp trình bày xử lý yêu cầu đến và chuyển giao cho lớp nghiệp vụ để xử lý tiếp.**

**Ứng dụngmáy chủ**

**Sự đến**

**yêu cầu được phân tích để tạo ra một truy vấn cơ sở dữ liệu**

Dữ liệu

Cơ sở dữ liệu

**Lớp cơ sở dữ liệu giao tiếp với cơ sở dữ liệu để thực hiện truy vấn và trả về kết quả.**

**Hình 1.1 Tìm kiếm dựa trênmột cơ sở dữ liệu truyền thống**

Các truy vấn được viết bằng SQL sử dụng các mệnh đề như where và like đã cung cấp nền tảng cho tìm kiếm. Các giải pháp này không nhất thiết phải hiệu quả và có hiệu suất cao để tìm kiếm dữ liệu toàn văn bản nhằm cung cấp chức năng tìm kiếm hiện đại.

Tuy nhiên, một số cơ sở dữ liệu hiện đại (ví dụ như Oracle và MySQL) hỗ trợ tìm kiếm toàn văn (truy vấn đối với văn bản tự do như bài đăng trên blog, bài đánh giá phim, bài nghiên cứu, v.v.), nhưng chúng có thể gặp khó khăn trong việc cung cấp các tìm kiếm hiệu quả trong thời gian gần như thực tế khi tải nặng. Xem thanh bên “Tìm kiếm toàn văn với cơ sở dữ liệu” để biết thêm chi tiết.

**6 CPHẦN1*Tổng quan***

Bản chất phân tán của một công cụ tìm kiếm như Elasticsearch cung cấp khả năng mở rộng tức thời mà hầu hết các cơ sở dữ liệu không được thiết kế cho. Một công cụ tìm kiếm được phát triển với cơ sở dữ liệu hỗ trợ (không có khả năng tìm kiếm toàn văn) có thể không cung cấp được kết quả tìm kiếm có liên quan cho các truy vấn, chưa nói đến việc đối phó với sự tăng trưởng về khối lượng và cung cấp kết quả theo thời gian thực.

**Tìm kiếm toàn văn với cơ sở dữ liệu**

Các cơ sở dữ liệu quan hệ như Oracle và MySQL hỗ trợ chức năng tìm kiếm toàn văn bản, mặc dù có ít chức năng hơn so với công cụ tìm kiếm toàn văn bản hiện đại như Elasticsearch. Chúng về cơ bản khác nhau khi nói đến việc lưu trữ và truy xuất dữ liệu, vì vậy bạn phải đảm bảo tạo nguyên mẫu các yêu cầu của mình trước khi chọn một trong số chúng. Thông thường, nếu các lược đồ không thay đổi hoặc tải dữ liệu thấp và bạn đã có một công cụ cơ sở dữ liệu có khả năng tìm kiếm toàn văn bản, thì việc bắt đầu bằng tìm kiếm toàn văn bản trên cơ sở dữ liệu có thể hợp lý.

* + 1. Cơ sở dữ liệu so với công cụ tìm kiếm

Khi xây dựng dịch vụ tìm kiếm trên cơ sở dữ liệu truyền thống, chúng ta cần cân nhắc và hiểu liệu các yêu cầu của mình có thể được cơ sở dữ liệu đáp ứng hiệu quả hay không. Hầu hết các cơ sở dữ liệu được thiết kế để lưu trữ lượng dữ liệu lớn nhưng không may là không phù hợp để sử dụng làm công cụ tìm kiếm toàn văn vì một số lý do:

* + - * *Hiệu suất lập chỉ mục và tìm kiếm—*Tìm kiếm toàn văn bản đòi hỏi khả năng lập chỉ mục hiệu quả và tìm kiếm hiệu suất cao cũng như khả năng phân tích, mà các cơ sở dữ liệu truyền thống không được tối ưu hóa. Cơ sở dữ liệu có thể gặp khó khăn khi lập chỉ mục khối lượng dữ liệu lớn và do đó, có thể thể hiện hiệu suất truy vấn kém. Các công cụ tìm kiếm như Elasticsearch và Solr được thiết kế riêng để xử lý khối lượng lớn dữ liệu văn bản và cung cấp kết quả tìm kiếm gần như theo thời gian thực. Các công cụ tìm kiếm có thể xử lý dữ liệu quy mô lớn, lập chỉ mục và tìm kiếm dữ liệu nhanh hơn nhiều so với các cơ sở dữ liệu truyền thống, vì chúng được thiết kế khá đầy đủ từ đầu để tối ưu hóa các hoạt động tìm kiếm. Thật không may, các cơ sở dữ liệu quan hệ thiếu các tính năng tìm kiếm nâng cao như logic mờ, từ gốc, từ đồng nghĩa, v.v.
      * *Tìm kiếm-*Tìm kiếm bằng cơ sở dữ liệu truyền thống ít nhiều dựa trên sự khớp chính xác của các giá trị dữ liệu. Mặc dù điều này phù hợp với các hoạt động tìm kiếm không liên quan đến tìm kiếm trên dữ liệu có cấu trúc, nhưng chắc chắn không được sử dụng đối với các truy vấn ngôn ngữ tự nhiên, thường phức tạp. Các truy vấn của người dùng thường bị viết sai chính tả, cấu trúc ngữ pháp sai hoặc không đầy đủ và có thể chứa các từ đồng nghĩa và các cấu trúc ngôn ngữ khác mà cơ sở dữ liệu không hiểu được

Trong các truy vấn ngôn ngữ tự nhiên, người dùng có thể không sử dụng chính xác các thuật ngữ họ đang tìm kiếm (lỗi chính tả) và thật không may, các cơ sở dữ liệu truyền thống không được thiết kế để hỗ trợ đầu vào của người dùng bị viết sai chính tả. Tính năng này được hỗ trợ bởi chức năng tìm kiếm khớp mờ (các từ tương tự nhưng không hoàn toàn giống nhau) trong các công cụ tìm kiếm hiện đại

* 1. ***Tìm kiếm hiện đạiđộng cơ* 7**

Trong cơ sở dữ liệu truyền thống, dữ liệu thường được chuẩn hóa, nghĩa là dữ liệu được phân tán trên nhiều bảng và cột. Điều này có thể gây khó khăn cho việc tìm kiếm dữ liệu trên nhiều trường trong một truy vấn duy nhất. Cơ sở dữ liệu truyền thống không được thiết kế để xử lý các loại dữ liệu phi cấu trúc và bán cấu trúc thường gặp trong các tình huống tìm kiếm toàn văn.

* + - * + *Phân tích và xử lý văn bản—*Công cụ tìm kiếm thường phải xử lý nhiều ngôn ngữ và bộ ký tự mà cơ sở dữ liệu truyền thống có thể không hỗ trợ. Công cụ tìm kiếm thực hiện phân tích và xử lý văn bản để trích xuất ý nghĩa từ văn bản, nhưng cơ sở dữ liệu truyền thống không được thiết kế hoặc tối ưu hóa cho mục đích này.
        + *Khả năng mở rộng và linh hoạt****—***Công cụ tìm kiếm toàn văn được thiết kế để xử lý

lượng lớn dữ liệu và tải truy vấn cao. Cơ sở dữ liệu truyền thống có thể gặp vấn đề về khả năng mở rộng khi xử lý lượng lớn dữ liệu văn bản.

Công cụ tìm kiếm được thiết kế từ đầu để xử lý dữ liệu phi cấu trúc, trong khi cơ sở dữ liệu được tối ưu hóa để xử lý dữ liệu có cấu trúc. Những hạn chế này làm cho cơ sở dữ liệu truyền thống ít phù hợp hơn để sử dụng làm công cụ tìm kiếm toàn văn; các công nghệ công cụ tìm kiếm chuyên biệt như Elasticsearch, Solr, Lucene, v.v., thường được sử dụng để cung cấp chức năng tìm kiếm nâng cao cho dữ liệu văn bản.

**GHI CHÚ** Nhiều cơ sở dữ liệu đã thêm chức năng tìm kiếm văn bản vào bộ tính năng của họ. Tuy nhiên, chúng vẫn có thể không cung cấp hiệu suất, khả năng mở rộng và chức năng ngang bằng với các công cụ tìm kiếm toàn văn chuyên dụng.

Không có gì ngăn cản chúng ta tiếp cận cả hai thế giới: trong một số trường hợp sử dụng, có thể sử dụng kết hợp cơ sở dữ liệu truyền thống và công cụ tìm kiếm. Ví dụ, có thể sử dụng cơ sở dữ liệu cho mục đích giao dịch và công cụ tìm kiếm cho mục đích tìm kiếm và phân tích. Nhưng trọng tâm của chúng tôi trong cuốn sách này là công cụ tìm kiếm—và đặc biệt là Elasticsearch. Trong phần tiếp theo, chúng tôi sẽ xem xét kỷ nguyên của công cụ tìm kiếm hiện đại trước khi giới thiệu Elasticsearch.

#### Công cụ tìm kiếm hiện đại

Các công cụ tìm kiếm hiện đại đang nỗ lực hết sức để đáp ứng các yêu cầu kinh doanh ngày càng tăng bằng cách áp dụng các tính năng mới và thú vị mỗi ngày. Phần cứng giá rẻ kết hợp với sự bùng nổ dữ liệu đang dẫn đến sự xuất hiện của những con quái thú tìm kiếm hiện đại này. Hãy cùng xem xét các công cụ tìm kiếm hiện đại và các tính năng cũng như chức năng mà chúng cung cấp. Chúng ta có thể tóm tắt những gì một công cụ tìm kiếm hiện đại tốt nên cung cấp như sau:

* + - Hỗ trợ hạng nhất cho dữ liệu toàn văn (không có cấu trúc) và có cấu trúc
    - Gợi ý nhập trước,tự động sửa lỗi và các đề xuất "ý bạn là"
    - Sự tha thứ cho lỗi chính tả của người dùng
    - Khả năng tìm kiếm trên vị trí địa lý
    - Khả năng mở rộng dễ dàng, tăng hoặc giảm, dựa trên nhu cầu thay đổi
    - Hiệu suất tuyệt vời: khả năng lập chỉ mục và tìm kiếm nhanh chóng
    - Kiến trúc cung cấp hệ thống phân tán có khả năng chịu lỗi và tính khả dụng cao
    - Hỗ trợ chức năng học máy

**8 CPHẦN1*Tổng quan***

Trong phần này, chúng tôi sẽ thảo luận ngắn gọn về các tính năng cấp cao của một công cụ tìm kiếm hiện đại. Sau đó, phần tiếp theo sẽ giới thiệu một số công cụ tìm kiếm có sẵn trên thị trường, bao gồm cả Elasticsearch.

* + 1. Chức năng

Các công cụ tìm kiếm hiện đại được phát triển để đáp ứng các yêu cầu tìm kiếm toàn văn bản đồng thời cung cấp các chức năng nâng cao khác. Chúng được thiết kế để cung cấp kết quả tìm kiếm nhanh chóng và có liên quan cho người dùng bằng cách lập chỉ mục và tìm kiếm khối lượng lớn dữ liệu văn bản (từ nay về sau, chúng tôi sẽ bỏ từ hiện đại khi đề cập đến công cụ tìm kiếm).

Công cụ tìm kiếm có thể nhanh chóng lập chỉ mục một lượng lớn dữ liệu văn bản và làm cho nó có thể tìm kiếm được. Quá trình này thường bao gồm việc chia dữ liệu văn bản thành các mã thông báo và xây dựng một chỉ mục đảo ngược, trong đó ánh xạ từng mã thông báo với các tài liệu có chứa nó.

Các công cụ tìm kiếm cũng được kỳ vọng sẽ thực hiện phân tích và xử lý văn bản nâng cao, chẳng hạn như từ đồng nghĩa, từ gốc, từ dừng và các kỹ thuật xử lý ngôn ngữ tự nhiên khác, để trích xuất ý nghĩa từ văn bản và cải thiện kết quả tìm kiếm. Chúng có thể xử lý các truy vấn của người dùng và xếp hạng kết quả tìm kiếm dựa trên nhiều yếu tố như mức độ liên quan và mức độ phổ biến. Chúng cũng có thể xử lý lượng truy vấn cao và lượng dữ liệu lớn và có thể mở rộng theo chiều ngang bằng cách thêm nhiều nút hơn vào một cụm.

Cuối cùng, các công cụ tìm kiếm cung cấp khả năng phân tích nâng cao, xem xét dữ liệu để cung cấp tóm tắt, kết luận và thông tin tình báo cho doanh nghiệp. Chúng cũng hỗ trợ hình ảnh trực quan phong phú, tìm kiếm gần như thời gian thực, giám sát hiệu suất và thông tin chi tiết dựa trên máy học.

* + 1. Công cụ tìm kiếm phổ biến

Mặc dù có một số ít công cụ tìm kiếm có sẵn trên thị trường, tôi sẽ chỉ đề cập đến ba trong số đó, tất cả đều được xây dựng trên Apache Lucene. Các phần sau sẽ xem xét Elasticsearch, Solr và OpenSearch.

**VÀ****TÌM KIẾM LASTIC**

Shay Banon, người sáng lập Elastic, đã phát triển một sản phẩm tìm kiếm có tên là Compass vào đầu năm 2000. Nó dựa trên một thư viện công cụ tìm kiếm nguồn mở có tên là Apache Lucene ([https://lucene.apache.org](https://lucene.apache.org/)). Lucene là thư viện tìm kiếm toàn văn của Doug Cutting, được viết bằng Java. Vì đây là thư viện nên chúng ta phải nhập và tích hợp nó với ứng dụng bằng API của nó. Compass và các công cụ tìm kiếm khác sử dụng Lucene để cung cấp dịch vụ công cụ tìm kiếm tổng quát để chúng ta không phải tích hợp Lucene từ đầu vào các ứng dụng. Cuối cùng, Shay quyết định từ bỏ Compass và tập trung vào Elasticsearch vì nó có nhiều tiềm năng hơn.

**MỘTPACHESOLR**

Apache Solr là một công cụ tìm kiếm nguồn mở được xây dựng trên Apache Lucene vào năm 2004. Solr là đối thủ cạnh tranh mạnh mẽ của Elasticsearch và có cộng đồng người dùng phát triển mạnh, và gần với nguồn mở hơn Elasticsearch (Elastic đã chuyển từ Apache sang Giấy phép Elastic và Giấy phép Công cộng Phía Máy chủ ([SSPL] vào đầu năm 2021). Cả Solr và Elasticsearch đều xuất sắc trong việc tìm kiếm toàn văn bản; tuy nhiên, Elasticsearch có thể có lợi thế hơn khi nói đến phân tích.

* 1. ***Tìm kiếm đàn hồiTổng quan* 9**

Trong khi cả hai sản phẩm đều cạnh tranh ở hầu hết mọi chức năng, Solr là lựa chọn ưa thích cho các tập dữ liệu tĩnh lớn hoạt động trong hệ sinh thái dữ liệu lớn. Rõ ràng là chúng ta phải chạy qua các nguyên mẫu và phân tích để chọn một sản phẩm; xu hướng chung là các dự án tích hợp với công cụ tìm kiếm lần đầu tiên sẽ cân nhắc Elasticsearch do tài liệu hướng dẫn hàng đầu, cộng đồng và khởi nghiệp gần như không có rào cản. Bạn phải so sánh chi tiết các trường hợp sử dụng dự định của mình cho công cụ tìm kiếm trước khi áp dụng và áp dụng một công cụ.

**MỘTNGƯỜI ĐÀN ÔNG****THECÁI BÚTSTÌM KIẾM**

Elastic đã thay đổi chính sách cấp phép vào năm 2021. Việc cấp phép, áp dụng cho các phiên bản phát hành Elasticsearch 7.11 trở lên, đã được chuyển từ nguồn mở sang giấy phép kép theo Giấy phép Elastic và SSPL. Giấy phép này cho phép cộng đồng sử dụng sản phẩm miễn phí, như mong đợi, nhưng các nhà cung cấp dịch vụ được quản lý không còn có thể cung cấp các sản phẩm dưới dạng dịch vụ. Đã có một cuộc cãi vã giữa Elastic và Amazon Web Services (AWS) khi AWS tạo ra một phiên bản phân nhánh của Elasticsearch—được gọi là Open Distro for Elasticsearch—và cung cấp nó dưới dạng dịch vụ được quản lý. Cuộc cãi vã này đã dẫn đến sự thay đổi trong giấy phép, cuối cùng dẫn đến sự ra đời của OpenSearch.

Khi Elastic chuyển từ mô hình cấp phép nguồn mở sang mô hình SSPL, một sản phẩm mới có tên là OpenSearch ([https://opensearch.org](https://opensearch.org/)) được phát triển để lấp đầy lỗ hổng do thỏa thuận cấp phép mới để lại. Mã cơ sở cho OpenSearch được tạo từ Elasticsearch mã nguồn mở và Kibana phiên bản 7.10.2. Phiên bản Khả dụng chung đầu tiên của sản phẩm 1.0 được phát hành vào tháng 7 năm 2021. Hãy chú ý đến việc OpenSearch trở thành đối thủ cạnh tranh của Elasticsearch trong không gian công cụ tìm kiếm.

Bây giờ chúng ta đã hiểu rõ về công cụ tìm kiếm hiện đại và hình thái của bối cảnh tìm kiếm, hãy cùng tìm hiểu tổng quan về Elasticsearch.

#### 1.4 Tổng quan về Elasticsearch

Elasticsearch là một công cụ tìm kiếm và phân tích mã nguồn mở. Được phát triển bằng Java, đây là một công cụ tìm kiếm cực nhanh, có tính khả dụng cao được xây dựng trên thư viện toàn văn phổ biến Apache Lucene ([https://lucene.apache.org](https://lucene.apache.org/)). Elasticsearch bao quanh chức năng mạnh mẽ của Lucene bằng cách cung cấp một hệ thống phân tán với giao diện RESTful. Lucene là sức mạnh của Elasticsearch và Kibana là giao diện người dùng quản trị để quản lý và làm việc với Elasticsearch. Chúng tôi làm việc với trình soạn thảo mã của Kibana (Dev Tools) trong suốt cuốn sách này.

Tìm kiếm toàn văn là nơi Elasticsearch nổi trội như một công cụ tìm kiếm hiện đại. Nó có thểlấy các tài liệu có liên quan để đáp ứng tiêu chí tìm kiếm của người dùng với tốc độ tuyệt vời. Chúng ta cũng có thể tìm kiếm các thuật ngữ chính xác, như từ khóa, ngày tháng hoặc một phạm vi số hoặc ngày tháng. Elasticsearch được tích hợp nhiều tính năng hàng đầu như tính liên quan, gợi ý "ý bạn là", tự động hoàn thành, tìm kiếm mờ và không gian địa lý, tô sáng và nhiều tính năng khác.

Ngoài việc là người đi đầu trong việc cung cấp khả năng tìm kiếm gần như thời gian thực, Elasticsearch còn nổi bật trong việc tổng hợp thống kê trên dữ liệu lớn. Tất nhiên, chúng ta phải cân nhắc trường hợp sử dụng trước khi áp dụng sản phẩm, vì Elasticsearch có thể không phù hợp nhất với mọi trường hợp sử dụng (tham khảo phần 1.4.3 để tìm hiểu về các trường hợp sử dụng). Elasticsearch cũng tự hào có các tính năng đáng khen ngợi như hiệu suất ứng dụng

**10 CPHẦN1*Tổng quan***

giám sát, phân tích dự đoán và phát hiện giá trị ngoại lai, cũng như giám sát và phát hiện mối đe dọa bảo mật.

Elasticsearch tập trung vào việc tìm kiếm ý nghĩa sâu sắc hơn trong dữ liệu đã được thu thập. Nó có thể tổng hợp dữ liệu, thực hiện các phép tính thống kê và tìm ra trí thông minh trong dữ liệu. Chúng ta có thể tạo ra các hình ảnh trực quan và bảng thông tin phong phú và chia sẻ chúng với những người khác bằng cách sử dụng công cụ Kibana. Elasticsearch có thể tìm ra các giá trị trung bình, tổng, phương sai và chế độ cũng như thực hiện các phân tích phức tạp như phân loại dữ liệu trong biểu đồ tần suất và các hàm phân tích khác.

Hơn nữa, Elasticsearch chạy các thuật toán học máy có giám sát và không giám sát trên dữ liệu của chúng tôi. Các mô hình giúp phát hiện các bất thường, tìm ra các giá trị ngoại lệ và dự báo các sự kiện. Ở chế độ học có giám sát, chúng tôi có thể cung cấp các tập huấn luyện để mô hình học và đưa ra dự đoán.

Elasticsearch cũng có khả năng quan sát các ứng dụng và tình trạng của chúng bằng cách theo dõi các số liệu hiệu suất như bộ nhớ và chu kỳ CPU của máy chủ web trong mạng. Nó cho phép chúng ta sàng lọc hàng triệu nhật ký máy chủ web để tìm hoặc gỡ lỗi các sự cố ứng dụng. Elasticsearch cũng đầu tư thời gian và nguồn lực vào việc xây dựng các giải pháp bảo mật: ví dụ, cảnh báo chúng ta về các mối đe dọa bảo mật, lọc IP, ngăn chặn điểm cuối, v.v.

* + 1. Các lĩnh vực cốt lõi

Elastic, công ty đứng sau Elasticsearch, đã định vị mình chủ yếu ở ba lĩnh vực cốt lõi: tìm kiếm, khả năng quan sát và bảo mật, như thể hiện trong hình 1.2. Chúng ta hãy cùng xem xét từng lĩnh vực này theo lượt.



**Hình 1.2 Các lĩnh vực ứng dụng cốt lõi của Elastic, công ty đứng sau Elasticsearch**

**VÀLASTICVÀ****DOANH NGHIỆPSTÌM KIẾM**

Liệucho phép người dùng tìm kiếm trên nhiều nhà cung cấp nội dung khác nhau (như Slack, Confluence, Google Drive và các nhà cung cấp khác) hoặc kích hoạt khả năng tìm kiếm cho các ứng dụng, ứng dụng và trang web của chúng tôi, bộ Elastic Enterprise Search giúp xây dựng các mô hình và công cụ tìm kiếm tùy chỉnh.

Tìm kiếm có thể được tích hợp sâu vào vô số ứng dụng trong nhiều lĩnh vực khác nhau—kinh doanh, cơ sở hạ tầng, ứng dụng, v.v. Người dùng có thể tạo ứng dụng web được Elasticsearch hỗ trợ, ứng dụng di động được Elasticsearch hỗ trợ hoặc dịch vụ tìm kiếm phía máy chủ với Elasticsearch là xương sống cho khả năng tìm kiếm. Sau này trong cuốn sách này, chúng tôi sẽ làm việc trên các ví dụ về tích hợp với Elasticsearch như một máy chủ tìm kiếm cho các ứng dụng.

***1.4 Tìm kiếm đàn hồiTổng quan* 11**

**VÀLASTICKHẢ NĂNG QUAN SÁT**

Các ứng dụng chạy trên cơ sở hạ tầng tạo ra nhiều số liệu thường được sử dụng để quan sát và giám sát ứng dụng. Chúng ta có thể sử dụng Elasticsearch trong không gian quan sát: trạng thái của các ứng dụng, máy chủ, giá đỡ và mô-đun đều có thể được giám sát, ghi nhật ký, theo dõi và cảnh báo. Chúng ta cũng có thể sử dụng các công cụ Elastic để thực hiện quản lý và giám sát ứng dụng trên quy mô lớn.

**VÀBẢO MẬT LASTIC**

Elastic bước vào lĩnh vực bảo mật bằng cách cho phép phát hiện và ngăn chặn mối đe dọa và cung cấp các tính năng nâng cao như khả năng loại bỏ phần mềm độc hại tại nguồn, mã hóa khi không hoạt động, v.v. Là một công cụ quản lý sự kiện và thông tin bảo mật (SIEM), Elastic đang định vị mình để bảo vệ các tổ chức bằng các bộ công cụ bảo mật tiên tiến của mình.

* + 1. Ngăn xếp đàn hồi

Elasticsearch là cốt lõi của công cụ tìm kiếm và một số sản phẩm Elastic bổ sung cho nó. Bộ sản phẩm này được gọi là Elastic Stack và bao gồm Kibana, Logstash, Beats và Elasticsearch. (Trước đây nó được gọi là ELK Stack nhưng được đổi tên thành Elastic Stack sau khi Beats được đưa vào bộ sản phẩm.)

Sự kết hợp của bốn sản phẩm này giúp xây dựng một ứng dụng doanh nghiệp bằng cách tích hợp, sử dụng, xử lý, phân tích, tìm kiếm và lưu trữ nhiều tập dữ liệu khác nhau từ các nguồn khác nhau. Như minh họa trong hình 1.3, Beats và Logstash đưa dữ liệu vào Elasticsearch, trong khi Kibana là giao diện người dùng trực quan hoạt động trên dữ liệu đó.

Beats (người vận chuyển dữ liệu)

**Beats giúp đưa dữ liệu vào Elasticsearch.**

**Kibana là một trang web đa diện**

**ứng dụng (UI).**

Khách hàng

Tập tin

**Elasticsearch, một công cụ tìm kiếm và phân tích, là trung tâm của Elastic Stack.**

Nhịp tim

Nội bộứng dụng người dùng

Nhịp điệu

Chức năngbeat

Kiểm toán nhịp

Bộ xử lý dữ liệu

**Logstash là một công cụ xử lý dữ liệu ETL (trích xuất, chuyển đổi, tải).**

Dịch vụ tìm kiếm

**Ứng dụng do người dùng thiết kế**

Bên ngoàihướng tới người dùng

ứng dụng

**Hình 1.3 Elastic Stack: Beats, Logstash, Elasticsearch và Kibana**



**12 CPHẦN1*Tổng quan***

Trước khi chúng ta chuyển sang xem xét các trường hợp sử dụng Elasticsearch, hãy cùng xem qua các bộ phận chuyển động thiết yếu này ở cấp độ cao. Ngoài Elasticsearch, chúng ta không thảo luận về ngăn xếp sản phẩm này trong cuốn sách này.

**BĂN UỐNG**

Beats là những trình vận chuyển dữ liệu có mục đích đơn lẻ; chúng tải dữ liệu từ nhiều hệ thống bên ngoài khác nhau và đưa vào Elasticsearch. Có nhiều loại beat có sẵn ngay khi xuất xưởng. Chúng bao gồm Filebeat, Metricbeat, Heartbeat, v.v., và mỗi loại thực hiện một tác vụ tiêu thụ dữ liệu cụ thể. Đây là các thành phần có mục đích đơn lẻ: ví dụ, Filebeat được thiết kế cho các phương thức vận chuyển dựa trên tệp và Metricbeat dành cho thông tin CPU và bộ nhớ hệ điều hành và máy quan trọng. Các tác nhân của beat được cài đặt trên máy chủ để chúng có thể tiêu thụ dữ liệu từ nguồn và gửi đến đích.

**LOGSTASH**

Logstash là một công cụ xử lý dữ liệu nguồn mở. Nó trích xuất dữ liệu có nguồn gốc từ nhiều nguồn, xử lý và gửi đến nhiều đích đến khác nhau. Trong quá trình xử lý dữ liệu, Logstash sẽ chuyển đổi và làm giàu dữ liệu. Nó hỗ trợ vô số nguồn và đích đến bao gồm tệp, HTTP, JMS, Kafka, Amazon S3, Twitter và hàng chục nguồn và đích đến khác. Nó thúc đẩy kiến ​​trúc đường ống và mọi sự kiện đi qua đường ống đều được phân tích cú pháp theo các quy tắc được cấu hình trước, do đó tạo ra một đường ống thời gian thực để thu thập dữ liệu.

**KNHỮNG ĐỨA TRẺ**

Kibana là một bảng điều khiển web đa năng cung cấp nhiều tùy chọn như thực hiện truy vấn; phát triển bảng thông tin, đồ thị và hình ảnh biểu đồ; và tạo danh sách thả xuống và tổng hợp. Tuy nhiên, chúng ta có thể sử dụng bất kỳ máy khách REST nào để giao tiếp với Elasticsearch để gọi API, không chỉ Kibana. Ví dụ, chúng ta có thể gọi API bằng cURL, Postman hoặc máy khách ngôn ngữ gốc.

* + 1. Các trường hợp sử dụng Elasticsearch

Việc xác định Elasticsearch cho một trường hợp sử dụng hoặc miền cụ thể là rất khó. Nó hiện diện ở khắp mọi nơi trong nhiều lĩnh vực từ tìm kiếm đến phân tích đến các công việc học máy. Nó được sử dụng rộng rãi trong nhiều ngành công nghiệp, bao gồm tài chính, quốc phòng, giao thông vận tải, chính phủ, bán lẻ, đám mây, giải trí, không gian, v.v. Hãy cùng xem xét tổng quan về cách Elasticsearch có thể được sử dụng trong một tổ chức.

**SCÔNG CỤ TÌM KIẾM**

Elasticsearch đã trở thành công nghệ được sử dụng cho khả năng tìm kiếm toàn văn bản. Sản phẩm không chỉ giới hạn ở tìm kiếm toàn văn bản mà còn có thể được sử dụng cho dữ liệu có cấu trúc và tìm kiếm dựa trên vị trí địa lý. Nói chung, khách hàng sử dụng Elasticsearch trong ba lĩnh vực: Tìm kiếm ứng dụng, Tìm kiếm doanh nghiệp và Tìm kiếm trang web.

Trong App Search, Elasticsearch đóng vai trò là xương sống, cung cấp khả năng tìm kiếm và phân tích cho các ứng dụng. Một dịch vụ tìm kiếm được Elasticsearch hỗ trợ có thể được thiết kế như một dịch vụ siêu nhỏ phục vụ các yêu cầu tìm kiếm của ứng dụng, chẳng hạn như tìm kiếm khách hàng, đơn đặt hàng, hóa đơn, email, v.v.

***1.4 Tìm kiếm đàn hồiTổng quan* 13**

Trong hầu hết các tổ chức, dữ liệu nằm rải rác trên nhiều kho dữ liệu, ứng dụng và cơ sở dữ liệu. Ví dụ, các tổ chức thường được tích hợp với Confluence, không gian mạng nội bộ, Slack, email, cơ sở dữ liệu, ổ đĩa đám mây (ổ đĩa iCloud, Google Drive, v.v.) và các dịch vụ khác. Việc đối chiếu và tìm kiếm qua lượng dữ liệu khổng lồ với các tích hợp với nhiều nguồn khác nhau là một thách thức đối với các tổ chức này. Đây là nơi Elasticsearch có thể được sử dụng cho Tìm kiếm doanh nghiệp và tổ chức dữ liệu.

Nếu chúng ta có một trang web kinh doanh trực tuyến thu thập dữ liệu, việc cung cấp tìm kiếm là điều cần thiết để thu hút khách hàng và giữ chân họ. Site Search là phần mềm dưới dạng dịch vụ (SaaS) do Elastic cung cấp, sau khi được bật, sẽ thu thập dữ liệu qua các trang web đã cho, tìm nạp dữ liệu và xây dựng các chỉ mục được hỗ trợ bởi Elasticsearch. Sau khi quá trình thu thập dữ liệu và lập chỉ mục hoàn tất, trang web có thể dễ dàng tích hợp với tiện ích tìm kiếm. Mô-đun Site Search cũng giúp tạo thanh tìm kiếm và đoạn mã liên quan đến thanh tìm kiếm đó. Quản trị viên trang web có thể sao chép đoạn mã đã tạo vào trang chủ của họ để kích hoạt thanh tìm kiếm ngay lập tức, do đó giúp trang web hoạt động đầy đủ với chức năng tìm kiếm tích hợp.

**BPHÂN TÍCH DOANH NGHIỆP**

Các tổ chức thu thập hàng tấn dữ liệu từ nhiều nguồn khác nhau và dữ liệu đó thường nắm giữ chìa khóa để tồn tại và thành công. Elasticsearch có thể giúp trích xuất xu hướng, số liệu thống kê và số liệu từ dữ liệu, cung cấp cho các tổ chức kiến ​​thức về hoạt động, doanh số, doanh thu, lợi nhuận và nhiều tính năng khác để quản lý kịp thời.

**SPHÂN TÍCH CHỨNG KHOÁN VÀ PHÁT HIỆN MỐI ĐE DỌA VÀ GIAN LẬN**

Bảo mật dữ liệu và các vi phạm tiềm ẩn là cơn ác mộng đối với các tổ chức. Phân tích bảo mật của Elastic-search giúp các tổ chức phân tích mọi thông tin—cho dù là từ ứng dụng, mạng, điểm cuối hay đám mây. Phân tích này có thể cung cấp thông tin chi tiết về các mối đe dọa và lỗ hổng, cho phép tổ chức săn lùng phần mềm độc hại và phần mềm tống tiền, do đó giảm nguy cơ trở thành con mồi của tin tặc.

**LOGGING VÀ GIÁM SÁT ỨNG DỤNG**

Các ứng dụng đưa ra rất nhiều dữ liệu dưới dạng nhật ký và số liệu ứng dụng. Các nhật ký này cung cấp thông tin chi tiết về tình trạng của ứng dụng. Với sự ra đời của thế giới đám mây và dịch vụ vi mô, nhật ký được phân tán trên các dịch vụ và việc phân tích có ý nghĩa là một công việc cồng kềnh. Elasticsearch là bạn của chúng ta ở đây. Một trong những trường hợp sử dụng phổ biến của Elasticsearch là lập chỉ mục nhật ký và phân tích chúng để tìm lỗi ứng dụng và mục đích gỡ lỗi.

Elasticsearch là một công cụ tìm kiếm và phân tích mạnh mẽ và linh hoạt, nhưng không phù hợp với mọi trường hợp sử dụng. Chúng ta hãy cùng xem qua các vấn đề chúng ta có thể gặp phải và các trường hợp sử dụng mà Elasticsearch không phải là lựa chọn phù hợp.

* + 1. Không phù hợpElasticsearch sử dụng

Không phải mọi trường hợp sử dụng đều có thể được Elasticsearch đáp ứng. Đây là một công cụ tìm kiếm và phân tích mạnh mẽ và linh hoạt, nhưng thật không may, công cụ này có những hạn chế mà chúng ta phải cân nhắc trước khi lựa chọn nó cho các yêu cầu của mình. Sau đây là một số tình huống mà Elasticsearch có thể là giải pháp không chính xác hoặc không hiệu quả:

**14 CPHẦN1*Tổng quan***

* *Dữ liệu quan hệ*—Elasticsearch không phải là công cụ phù hợp để hỗ trợ tìm kiếm trên dữ liệu có mối quan hệ và cần thực hiện các phép nối cơ sở dữ liệu phức tạp. Elasticsearch không được thiết kế để xử lý các cấu trúc dữ liệu quan hệ phức tạp. Nếu dữ liệu của bạn có nhiều mối quan hệ, một cơ sở dữ liệu quan hệ như MySQL hoặc PostgreSQL có thể phù hợp hơn. Hầu hết các cơ sở dữ liệu hiện đại (MySQL, PostgreSQL, v.v.) cũng cung cấp khả năng tìm kiếm toàn văn bản, mặc dù các tính năng không tiên tiến như trong một công cụ tìm kiếm hiện đại như Elasticsearch.
* *Dữ liệu giao dịch*—Elasticsearch là một công cụ tìm kiếm “cuối cùng là nhất quán”, khiến nó không phù hợp với các ứng dụng yêu cầu tính nhất quán ngay lập tức, chẳng hạn như giao dịch tài chính. Đối với các loại trường hợp sử dụng này, hãy cân nhắc sử dụng cơ sở dữ liệu quan hệ truyền thống hoặc cơ sở dữ liệu NoSQL như MongoDB.
* *Dữ liệu không gian địa lý*—Mặc dù Elasticsearch có hỗ trợ tích hợp cho dữ liệu không gian địa lý, nhưng nó có thể không phải là giải pháp hiệu quả nhất cho phân tích không gian địa lý quy mô lớn. Đối với những trường hợp sử dụng này, hãy cân nhắc sử dụng cơ sở dữ liệu không gian địa lý chuyên dụng như PostGIS hoặc nền tảng phân tích không gian địa lý như ArcGIS.
* *Khối lượng công việc ghi cao*—Elasticsearch có thể xử lý khối lượng công việc đọc cao, nhưng không được tối ưu hóa cho khối lượng công việc ghi cao. Nếu bạn cần lập chỉ mục khối lượng dữ liệu lớn theo thời gian thực, hãy cân nhắc sử dụng công cụ lập chỉ mục chuyên dụng như Apache Flume hoặc Apache Kafka.
* *Dữ liệu xử lý phân tích trực tuyến (OLAP)*—Nếu bạn cần thực hiện phân tích đa chiều phức tạp trên các tập dữ liệu lớn, cơ sở dữ liệu OLAP truyền thống như Microsoft Analysis Services hoặc IBM Cognos có thể phù hợp hơn Elasticsearch.
* *Dữ liệu nhị phân lớn*—Mặc dù Elasticsearch có thể xử lý lượng lớn dữ liệu văn bản, nhưng nó có thể không phải là giải pháp tốt nhất để lập chỉ mục và tìm kiếm dữ liệu nhị phân lớn như video hoặc hình ảnh. Đối với những trường hợp sử dụng này, hãy cân nhắc sử dụng kho dữ liệu nhị phân chuyên dụng như Hadoop Distributed File System (HDFS), Amazon S3 hoặc Azure Files.
* *Phân tích thời gian thực*—Elasticsearch rất tuyệt vời để thực hiện tìm kiếm và phân tích theo thời gian thực trên các tập dữ liệu lớn, nhưng có thể không phải là giải pháp hiệu quả nhất để xử lý và phân tích dữ liệu theo thời gian thực. Thay vào đó, hãy cân nhắc một nền tảng phân tích theo thời gian thực chuyên biệt như Apache Spark hoặc Apache Flink.
* *Ứng dụng nhạy cảm với độ trễ*—Mặc dù Elasticsearchđược thiết kế để xử lý các truy vấn tìm kiếm và phân tích khối lượng lớn, nhưng vẫn có thể gặp vấn đề về độ trễ khi xử lý lượng dữ liệu lớn. Đối với các ứng dụng yêu cầu thời gian phản hồi dưới một phần nghìn giây, một công cụ tìm kiếm chuyên biệt như Apache Solr hoặc cơ sở dữ liệu dạng cột như Apache Cassandra có thể phù hợp hơn.
* *Các loại khác*—Elasticsearch không phải là giải pháp được ưa chuộng cho dữ liệu chuỗi thời gian, dữ liệu đồ thị, dữ liệu trong bộ nhớ và nhiều loại dữ liệu khác. Nếu bạn cần lưu trữ và phân tích dữ liệu chuỗi thời gian, một cơ sở dữ liệu chuỗi thời gian chuyên biệt như InfluxDB hoặc TimescaleDB có thể phù hợp hơn. Tương tự như vậy, một cơ sở dữ liệu đồ thị như Neo4j có thể giúp bạn xử lý dữ liệu đồ thị.

Điều quan trọng là phải đánh giá trường hợp sử dụng và yêu cầu cụ thể của bạn trước khi chọn Elasticsearch làm công nghệ và công cụ của bạn. Trong phần tiếp theo, chúng tôi thảo luận về những quan niệm sai lầm phổ biến về Elasticsearch như một công cụ, một công nghệ và một giải pháp tìm kiếm.

* 1. ***Tìm kiếm đàn hồiTổng quan* 15**
     1. Những quan niệm sai lầm

Một quan niệm sai lầm lớn về Elasticsearch là nhầm lẫn nó với một cơ sở dữ liệu quan hệ truyền thống. Một hiểu lầm phổ biến khác là việc thiết lập Elasticsearch rất dễ dàng, trong khi thực tế, cần phải có nhiều điều chỉnh để thiết lập một cụm có kích thước phù hợp. Ngoài ra, Elasticsearch thường được coi là một công nghệ được sử dụng để tìm kiếm văn bản, trong khi thực tế nó có thể được sử dụng cho nhiều trường hợp sử dụng tìm kiếm và phân tích. Danh sách sau đây tóm tắt một số quan niệm sai lầm phổ biến về Elasticsearch:

* + - * *Elasticsearch dễ thiết lập và quản lý.*Trong khi Elasticsearch tương đối dễ thiết lập và bắt đầu, thì việc quản lý và mở rộng quy mô khi dữ liệu tăng lên và các trường hợp sử dụng tăng lên có thể là một thách thức. Mặc dù mọi thứ đều hoạt động ngay khi xuất xưởng, giúp cuộc sống của kỹ sư dễ dàng hơn, nhưng việc đưa Elasticsearch vào môi trường sản xuất đòi hỏi nhiều nỗ lực. Chúng ta có thể cần điều chỉnh cấu hình và tinh chỉnh bộ nhớ, quản lý lỗi nút hoặc thậm chí mở rộng cụm để xử lý hàng petabyte dữ liệu khi dữ liệu của chúng ta tăng lên.
      * *Elasticsearch là một cơ sở dữ liệu quan hệ.*Elasticsearch không phải là cơ sở dữ liệu quan hệ và không hỗ trợ các tính năng cơ sở dữ liệu quan hệ truyền thống như giao dịch, khóa ngoại và các hoạt động liên kết phức tạp. Ví dụ, chúng tôi không thể thực thi tính toàn vẹn tham chiếu hoặc thực hiện các hoạt động liên kết phức tạp trong Elasticsearch. Nếu bạn cần những tính năng này, một cơ sở dữ liệu quan hệ đã được chứng minh như MySQL hoặc PostgreSQL chắc chắn là giải pháp của bạn.
      * *Elasticsearch có thể xử lý mọi loại dữ liệu.*Elasticsearch rất linh hoạt và có thể xử lý nhiều loại dữ liệu, nhưng nó không được thiết kế để xử lý mọi loại dữ liệu một cách dễ dàng như nhau. Ví dụ, nó có thể không phải là giải pháp tốt nhất để xử lý và phân tích dữ liệu theo thời gian thực hoặc để xử lý dữ liệu nhị phân lớn. Nếu bạn cần lưu trữ và xử lý dữ liệu nhị phân lớn như video hoặc hình ảnh, hãy cân nhắc sử dụng kho dữ liệu nhị phân chuyên dụng như HDFS hoặc Amazon S3.
      * *Elasticsearch chỉ dùng để tìm kiếm văn bản.*Trong khi Elasticsearch rất tuyệt vời cho tìm kiếm văn bản, nó cũng có thể thực hiện phân tích phức tạp trên dữ liệu có cấu trúc và không có cấu trúc. Ví dụ, chúng ta có thể sử dụng Elasticsearch để thực hiện tổng hợp, phân tích dữ liệu nhật ký và trực quan hóa dữ liệu bằng Kibana.
      * *Elasticsearch có thể thay thế tất cả các công nghệ khác.*Elasticsearch là một công cụ mạnh mẽ và linh hoạtcông nghệ, nhưng nó không phải là giải pháp phù hợp với mọi trường hợp và không phải là lựa chọn tốt nhất cho mọi trường hợp sử dụng. Ví dụ, nó không bao giờ có thể thay thế cơ sở dữ liệu quan hệ truyền thống.
      * *Elasticsearch luôn nhanh hơn các công nghệ khác*. Elasticsearch được thiết kế để có hiệu suất cao và dự kiến ​​sẽ hoạt động tốt khi tải nặng. Tuy nhiên, Elasticsearch chỉ có thể làm được một số việc nhất định và hiệu suất của nó chủ yếu phụ thuộc vào cách các kỹ sư nền tảng tinh chỉnh nó.
      * *Elasticsearch chỉ xử lý dữ liệu lớn.*Elasticsearch có thể xử lý hàng petabyte dữ liệu trong các tập dữ liệu lớn, nhưng cũng có hiệu suất tương đương khi xử lý các tập dữ liệu nhỏ theo thứ tự vài gigabyte. Ví dụ, chúng ta có thể sử dụng Elasticsearch để tìm kiếm và phân tích dữ liệu cho cơ sở dữ liệu email nhỏ của một tổ chức hoặc một công ty khởi nghiệp mà không cần nhiều nỗ lực.

**16 CPHẦN1 *Tổng quan***

Đây chỉ là một vài ví dụ về những quan niệm sai lầm về Elasticsearch. Như đã đề cập trước đó, bạn phải đánh giá cẩn thận các yêu cầu cụ thể và trường hợp sử dụng của mình trước khi chọn Elasticsearch hoặc bất kỳ công nghệ nào khác.

#### Sự áp dụng phổ biến

Một danh sách dài các tổ chức sử dụng Elasticsearch cho mọi thứ, từ tìm kiếm đến phân tích kinh doanh, phân tích nhật ký, giám sát cảnh báo bảo mật và quản lý ứng dụng, cũng như sử dụng nó như một kho lưu trữ tài liệu. Hãy cùng xem xét một số tổ chức này và cách họ sử dụng Elasticsearch trong hoạt động của mình.

Uber cung cấp năng lượng cho dự đoán người đi xe và sự kiện bằng Elasticsearch. Nó thực hiện điều này bằng cách lưu trữ hàng triệu sự kiện, tìm kiếm qua chúng và phân tích dữ liệu với tốc độ gần như thời gian thực. Uber dự đoán nhu cầu dựa trên vị trí, thời gian, ngày và các biến số khác, bao gồm cả việc xem xét dữ liệu trong quá khứ. Điều này giúp Uber cung cấp các chuyến đi khá.

Netflix đã áp dụng Elastic Stack để cung cấp thông tin chi tiết về khách hàng cho các nhóm nội bộ của mình. Họ cũng sử dụng Elasticsearch để phân tích sự kiện nhật ký nhằm hỗ trợ gỡ lỗi, cảnh báo và quản lý các dịch vụ nội bộ. Các chiến dịch email và hoạt động của khách hàng đều được hỗ trợ bởi công cụ Elasticsearch. Lần tới khi bạn nhận được email từ Netflix đề cập đến một bộ phim hoặc loạt phim truyền hình mới được thêm vào, hãy nhớ rằng các phân tích chiến dịch đằng sau email đơn giản đó đều được Elasticsearch hỗ trợ.

PayPal đã sử dụng Elasticsearch như một công cụ tìm kiếm để cho phép khách hàng lưu trữ và tìm kiếm thông qua các giao dịch của họ. Công ty đã triển khai các tính năng tìm kiếm giao dịch cùng với phân tích để các thương gia, khách hàng cuối và nhà phát triển sử dụng.

Tương tự như vậy, công ty thương mại điện tử trực tuyến eBay đã áp dụng Elasticsearch để hỗ trợ tìm kiếm toàn văn bản của người dùng cuối. Là người dùng, chúng tôi đang sử dụng Elasticsearch trực tiếp khi tìm kiếm qua kho hàng của eBay. Công ty cũng sử dụng Elastic Stack để phân tích, giám sát nhật ký và lưu trữ dữ liệu giao dịch trong kho lưu trữ tài liệu.

GitHub, một kho lưu trữ mã phổ biến dành cho các nhà phát triển, lập chỉ mục cho 8 triệu kho lưu trữ mã (và đang tiếp tục tăng) của mình—bao gồm hơn 2 tỷ tài liệu—bằng Elastic-search để cho phép người dùng có trải nghiệm tìm kiếm mạnh mẽ. Tương tự như vậy, Stack Overflow sử dụng Elasticsearch để cung cấp cho các nhà phát triển các câu trả lời nhanh chóng và có liên quan, và Medium (một nền tảng blog phổ biến) sử dụng Elastic Stack để phục vụ các truy vấn của người đọc ở chế độ gần như thời gian thực.

Trước khi kết thúc chương này, chúng ta chỉ nên đề cập đến một chủ đề đang thịnh hành gần đây: các công cụ trí tuệ nhân tạo (AI) như ChatGPT của OpenAI và Bard của Google. Theo tôi, việc giới thiệu các công cụ này sẽ thay đổi đáng kể không gian tìm kiếm. Hãy cùng thảo luận về tác động của chúng đối với tìm kiếm hiện đại, bao gồm các công cụ tìm kiếm như Elasticsearch.

#### Trí tuệ nhân tạo và tìm kiếm hiện đại

Trừ khi bạn đang sống trong hang động, bạn chắc chắn đã nghe nói về một cuộc cách mạng internet gần đây: ChatGPT. ChatGPT là một công cụ AI tạo sinh được nhóm OpenAI phát triển và phát hành vào tháng 11 năm 2022. Trong 25 năm kinh nghiệm trong lĩnh vực CNTT, tôi chưa bao giờ thấy một công cụ công nghệ nào thắp sáng internet như ChatGPT. Không thường xuyên như vậy về mặt kỹ thuật

***1.6 Trí tuệ nhân tạo và hiện đạitìm kiếm* 17**

công cụ tuyệt vời nằm trong tay công chúng có thể giúp họ theo những cách không thể tưởng tượng được, chẳng hạn như lập hành trình du lịch cho chuyến đi mùa hè tới Athens, tóm tắt một tài liệu pháp lý theo cách dễ hiểu, phát triển kế hoạch tự lực để giảm cân, phân tích mã để tìm lỗi bảo mật và hiệu suất, thiết kế mô hình dữ liệu của ứng dụng, so sánh và đối chiếu các công nghệ cho một trường hợp sử dụng cụ thể, viết thư khiếu nại tới CEO của Twitter và nhiều hơn nữa.

Trò chuyệnGPT ([https://chat.openai.com](https://chat.openai.com/)) là một tác nhân đàm thoại (chatbot) được xây dựng trên kiến ​​trúc GPT (biến áp được đào tạo trước tạo sinh) và có khả năng tạo văn bản giống con người dựa trên lời nhắc của người dùng. Đây là một ví dụ về mô hình ngôn ngữ lớn (LLM) được thiết kế cho cuộc trò chuyện với mục tiêu cụ thể là tạo ra nội dung an toàn, có liên quan trong khi tham gia vào một cuộc đối thoại có ý nghĩa. Mô hình được cung cấp một lượng lớn dữ liệu văn bản và học cách dự đoán từ tiếp theo trong một câu. Nó được đào tạo bằng cách sử dụng nhiều loại văn bản internet khác nhau, nhưng cũng có thể được tinh chỉnh bằng các tập dữ liệu cụ thể cho nhiều tác vụ khác nhau. Thông qua quá trình này, mô hình học các phần và bưu kiện của văn bản ngôn ngữ của con người: ngữ pháp, dấu câu, cú pháp, sự thật về thế giới và một số mức độ khả năng lập luận.

**GHI CHÚ***Thạc sĩ Luật*là một thuật ngữ rộng dùng để chỉ bất kỳ mô hình quy mô lớn nào được đào tạo để hiểu hoặc tạo ra văn bản giống con người. Các mô hình này được đặc trưng bởi số lượng lớn các tham số và khả năng xử lý nhiều tác vụ xử lý ngôn ngữ tự nhiên. LLM có thể dựa trên nhiều kiến ​​trúc và phương pháp đào tạo khác nhau.

Với việc phát hành ChatGPT ra công chúng, một cuộc đua bất ngờ trong không gian tìm kiếm sử dụng AI đã xuất hiện chỉ sau một đêm. ChatGPT đã trở thành kẻ phá hoại nhiều ngành công nghiệp và không kém phần đe dọa đối với tìm kiếm của Google. Các công cụ được AI hỗ trợ tương tự như ChatGPT sẽ phá hoại nhiều ngành công nghiệp trong những năm tới. Dưới áp lực rất lớn—và có thể là để giữ vị thế dẫn đầu trong tìm kiếm—Google đã quyết định tung ra phiên bản AI tạo ra hội thoại của mình: tác nhân của nó có tên là Bard ([https://bard.google.com](https://bard.google.com/)) được công bố rộng rãi vào tháng 5 năm 2023.

Trong khi đó, Microsoft cam kết đầu tư 10 tỷ đô la vào ChatGPT, cao hơn khoản đầu tư ban đầu là 3 tỷ đô la kể từ năm 2019. Trình duyệt Edge của Microsoft được tích hợp với ChatGPT thông qua công cụ tìm kiếm Bing, cũng được công khai vào tháng 5 năm 2023. Ngoài ra, Microsoft đã triển khai các ứng dụng Microsoft 365 hỗ trợ AI, do đó, tác nhân AI có sẵn trong Microsoft Word, Excel, email và các công cụ khác. LLaMA của Meta là một công cụ khác đã bắt đầu cạnh tranh trong cuộc đua AI tạo sinh.

Các mô hình GPT-3 và -4 đã được đào tạo trên hàng tỷ bản sao kỹ thuật số của sách, bài báo, báo cáo, blog, v.v. Mô hình GPT-4 được cung cấp dữ liệu cho đến tháng 9 năm 2021 (nó không thể truy xuất dữ liệu sau ngày đó). Mặc dù không có quyền truy cập internet để GPT-4 lấy thông tin theo thời gian thực, khi tôi viết bài này, OpenAI vừa phát hành phiên bản beta của phiên bản trình duyệt web dành cho những người đăng ký Plus. Vì vậy, tôi hy vọng một trợ lý AI tạo sinh hỗ trợ internet từ OpenAI sẽ sớm có sẵn cho công chúng.

Các kỹ sư tìm kiếm đang được hỏi những câu hỏi cơ bản, bao gồm cách các tác nhân AI tạo ra sẽ thay đổi quá trình tìm kiếm. Hãy trả lời câu hỏi này bằng cách hỏi

**18 CPHẦN1*Tổng quan***

ChatGPT cách các tác nhân AI có thể bổ sung hoặc hỗ trợ tìm kiếm hiện đại hoặc thay đổi hướng đi của nó. Đây là những lĩnh vực mà các công cụ như AI tạo sinh sẽ định hình lại không gian tìm kiếm:

* *Tìm kiếm trực quan*—Các truy vấn tìm kiếm sẽ trở nên mang tính đàm thoại và trực quan hơn. Các mô hình AI tạo sinh như GPT-4 có khả năng hiểu ngôn ngữ tự nhiên tiên tiến, cho phép chúng diễn giải các truy vấn phức tạp hiệu quả hơn. Người dùng sẽ không còn cần phải dựa vào các từ khóa hoặc cụm từ cụ thể nữa; họ có thể chỉ cần đặt câu hỏi như khi trò chuyện với người khác. Điều này sẽ cho phép có kết quả tìm kiếm chính xác và phù hợp hơn, vì AI có thể hiểu rõ hơn ngữ cảnh và mục đích của truy vấn. Với sự ra mắt và phát hành các tác nhân và mô hình AI tạo sinh có khả năng cao, sẽ có nhiều phạm vi hơn để định hình lại đáng kể các khả năng tìm kiếm toàn văn do các công cụ tìm kiếm hiện đại như Elasticsearch cung cấp. Chúng ta có thể mong đợi thấy một số thay đổi quan trọng sẽ định nghĩa lại trải nghiệm tìm kiếm cho cả người dùng và nhà phát triển khi công nghệ này ngày càng được tích hợp vào các nền tảng tìm kiếm.
* *Tìm kiếm được cá nhân hóa—*Với việc kết hợp AI tạo sinh, kết quả tìm kiếm có thể trở nên cá nhân hóa và thích ứng hơn. Các công cụ tìm kiếm sẽ có thể học được nhiều điểm dữ liệu có giá trị từ sở thích, hành vi và lịch sử tìm kiếm của người dùng, từ đó giúp các công cụ điều chỉnh kết quả để đáp ứng nhu cầu của từng người dùng. Khi AI thu thập nhiều dữ liệu hơn, nó sẽ liên tục tinh chỉnh sự hiểu biết của mình về những gì người dùng đang tìm kiếm, dẫn đến trải nghiệm tìm kiếm ngày càng được tùy chỉnh.
* *Tìm kiếm dự đoán—*Trí tuệ nhân tạo tạo ra tiềm năng giúp các công cụ tìm kiếm chủ động hơn trong việc dự đoán nhu cầu của người dùng. Thay vì chỉ phản hồi các truy vấn, các công cụ tìm kiếm do AI điều khiển có thể dự đoán thông tin mà người dùng quan tâm, dựa trên các tương tác trước đó hoặc bối cảnh hiện tại của họ. Điều này sẽ cho phép các nền tảng tìm kiếm chủ động đưa ra các gợi ý có liên quan, tăng giá trị của trải nghiệm tìm kiếm và giảm nhu cầu người dùng thực hiện các truy vấn bổ sung.
* *Tìm kiếm nâng cao—*Trí tuệ nhân tạo tạo ra sẽ cho phép các công cụ tìm kiếm cung cấp kết quả tìm kiếm đa dạng và phong phú hơn. Bằng cách hiểu ngữ cảnh và ngữ nghĩa của truy vấn, các công cụ tìm kiếm do AI điều khiển có thể tạo ra các bản tóm tắt nội dung và hình ảnh trực quan có liên quan và thậm chí tổng hợp thông tin mới để giúp trả lời câu hỏi của người dùng. Điều này sẽ dẫn đến trải nghiệm tìm kiếm toàn diện và nhiều thông tin hơn, vượt xa việc chỉ liên kết đến nội dung hiện có.

Theo tôi, việc giới thiệu AI tạo sinh sẽ cách mạng hóa khả năng tìm kiếm toàn văn, khiến công cụ tìm kiếm trở nên đàm thoại hơn, cá nhân hóa hơn, thích ứng hơn và chủ động hơn. Điều này không chỉ nâng cao trải nghiệm của người dùng mà còn mang đến những cơ hội mới cho các doanh nghiệp và nhà phát triển để tạo ra các ứng dụng và dịch vụ tìm kiếm sáng tạo. Các nhóm đang nỗ lực để áp dụng những thay đổi sắp tới trong không gian tìm kiếm với sự ra đời của AI. Vì vậy, hãy mong đợi một cuộc cách mạng không gian tìm kiếm!

Chương này đặt nền tảng cho việc sử dụng Elasticsearch bằng cách giới thiệu khả năng tìm kiếm của nó và xem xét cách tìm kiếm đã trở thành một phần không thể thiếu của nhiều ứng dụng. Trong chương tiếp theo, chúng tôi cài đặt, cấu hình và chạy Elasticsearch và

***Bản tóm tắt* 19**

Kibana và chúng tôi sử dụng Elasticsearch bằng cách lập chỉ mục một số tài liệu và chạy truy vấn tìm kiếm và phân tích. Hãy theo dõi!

#### Bản tóm tắt

* + Tìm kiếm là chức năng bình thường mới và được các tổ chức tìm kiếm nhiều nhất, mang lại lợi thế cạnh tranh.
  + Các công cụ tìm kiếm được xây dựng bằng cách sử dụng cơ sở dữ liệu quan hệ làm nền tảng để phục vụ mục đích tìm kiếm của chúng ta nhưng không thể thực hiện đầy đủ chức năng tìm kiếm như trong các công cụ tìm kiếm hiện đại.
  + Các công cụ tìm kiếm hiện đại cung cấp khả năng tìm kiếm toàn văn, đa diện và nhiều lợi ích từ tìm kiếm cơ bản đến tìm kiếm nâng cao và các chức năng phân tích, tất cả đều có hiệu suất trong tích tắc. Chúng cũng được kỳ vọng có thể xử lý tera-byte đến petabyte dữ liệu và mở rộng quy mô nếu cần.
  + Elasticsearch là một công cụ tìm kiếm và phân tích mã nguồn mở được xây dựng trên Apache Lucene. Đây là một ứng dụng phía máy chủ có tính khả dụng cao được phát triển bằng Java.
  + Vì Elasticsearch được thiết kế như một sản phẩm không phụ thuộc vào ngôn ngữ lập trình, nên việc giao tiếp với máy chủ diễn ra qua HTTP bằng cách sử dụng các API RESTful phong phú. Các API này nhận và gửi dữ liệu ở định dạng JSON.
  + Elastic Stack là một bộ sản phẩm bao gồm Beats, Logstash, Elasticsearch và Kibana. Beats là trình vận chuyển dữ liệu có mục đích duy nhất, Logstash là công cụ ETL (trích xuất, chuyển đổi, tải) xử lý dữ liệu, Kibana là công cụ UI quản trị và Elasticsearch là trái tim và linh hồn của ngăn xếp.
  + Elastic Stack cho phép một tổ chức định vị mình trong ba lĩnh vực cốt lõi: tìm kiếm, khả năng quan sát và bảo mật.
  + Elasticsearch đã trở nên phổ biến trong vài năm trở lại đây do khả năng tìm kiếm và phân tích có cấu trúc/không có cấu trúc; bộ API RESTful phong phú; bản chất không có lược đồ; cùng các đặc điểm về hiệu suất, tính khả dụng cao và khả năng mở rộng.
  + Tìm kiếm hỗ trợ AI đã có mặt. Với sự ra đời của AI tạo sinh và ChatGPT, không gian tìm kiếm sẽ được khám phá sâu hơn và tìm kiếm sẽ trở nên trực quan và có khả năng dự đoán hơn.

*Bắt đầu*

***Chương này bao gồm***

* Lập chỉ mục các tài liệu mẫu với Elasticsearch
* Lấy lại, xóa,và cập nhật tài liệu
* Tìm kiếm với các truy vấn cơ bản đến nâng cao
* Đang chạy tổng hợptrên dữ liệu

Chương này nói về việc trải nghiệm Elasticsearch. Elasticsearch là một tệp nhị phân Java có thể tải xuống từ trang web của công ty Elastic. Sau khi máy chủ được cài đặt và chạy, chúng ta có thể tải dữ liệu kinh doanh của mình, dữ liệu này sẽ được Elasticsearch phân tích và lưu trữ. Sau khi chuẩn bị dữ liệu cho Elasticsearch, chúng ta có thể thực hiện các truy vấn tìm kiếm cũng như tổng hợp trên dữ liệu đó.

Mặc dù bất kỳ máy khách nào có khả năng gọi lệnh gọi REST (cURL, Postman, SDK lập trình, v.v.) đều có thể giao tiếp với Elasticsearch, chúng tôi sử dụng Kibana làm máy khách ưa thích trong suốt cuốn sách này. Kibana là một ứng dụng web UI phong phú từ Elastic. Đây là trình chỉnh sửa trực quan đi kèm với tất cả các tính năng để giúp khám phá, phân tích, quản lý và duy trì cụm và dữ liệu của chúng tôi. Với Kibana, chúng tôi có được nhiều khả năng như các hàm phân tích và thống kê nâng cao, hình ảnh trực quan và bảng điều khiển phong phú, mô hình học máy, v.v. Khi Elasticsearch phơi bày tất cả các chức năng của nó

**20**

***2.1 Chuẩn bị Elasticsearch vớidữ liệu* 21**

thông qua API RESTful, chúng ta có thể xây dựng các truy vấn bằng các API này trong trình soạn thảo Kibana và giao tiếp với máy chủ qua HTTP.

Để thực hiện các mẫu trong chương này, bạn cần một môi trường chạy với Elasticsearch và Kibana. Nếu bạn chưa thiết lập môi trường đó, hãy làm theo hướng dẫn trong Phụ lục A để tải xuống và cài đặt phần mềm của bạn và đưa máy chủ Elasticsearch và Kibana UI lên.

**GHI CHÚ**Cài đặt Elasticsearch và Kibana có thể có nhiều cách, từ tải xuống các tệp nhị phân và giải nén rồi cài đặt chúng vào máy cục bộ của bạn theo cách truyền thống cho đến sử dụng trình quản lý gói, Docker hoặc thậm chí là đám mây. Chọn cách cài đặt phù hợp để phát triển để bắt đầu.

**Sao chép toàn bộ mã vào trình soạn thảo Kibana của bạn**

Để làm cho các bài tập mã hóa dễ dàng, tôi đã tạo ra một[ch02\_getting\_started.txt](https://github.com/madhusudhankonda/elasticsearch-in-action/blob/9f292ffcb96e6a5736ae80103dc4565f204502e5/kibana_scripts/ch02_getting_started.txt)lưu trữ dướithư mục kibana\_scripts ở gốc của kho lưu trữ. Sao chép nội dung của tệp này vào bản cài đặt Kibana của bạn. Bạn có thể làm việc thông qua các ví dụ bằng cách thực thi các đoạn mã riêng lẻ trong khi theo dõi nội dung của chương.

Cuối cùng, chúng ta thu nhỏ và phân tích dữ liệu bằng cách thực hiện hai loại tổng hợp: metric và bucket. Với các loại tổng hợp này, chúng ta sử dụng các truy vấn để lấy các số liệu như giá trị trung bình, tổng, giá trị tối thiểu và tối đa, v.v. Sau khi bạn đã chạy ứng dụng, hãy bắt đầu với Elasticsearch.

#### Chuẩn bị Elasticsearch với dữ liệu

Một công cụ tìm kiếm không thể hoạt động trên không khí! Nó cần dữ liệu làm đầu vào để có thể tạo ra kết quả khi được truy vấn. Chúng ta cần đổ dữ liệu của mình vào Elasticsearch, đây là bước đầu tiên để chuẩn bị cho công cụ. Nhưng trước khi bắt đầu lưu trữ dữ liệu trong Elasticsearch, chúng ta hãy tìm hiểu ứng dụng mẫu mà chúng ta sử dụng trong chương này.

Đối với các ví dụ của chúng tôi, chúng tôi cần hiểu biết cơ bản về miền vấn đề và mô hình dữ liệu. Giả sử chúng ta đang xây dựng một hiệu sách trực tuyến; rõ ràng là chúng ta không thiết kế toàn bộ ứng dụng—chúng ta chỉ quan tâm đến phần mô hình dữ liệu để thảo luận. Chúng ta sẽ xem xét chi tiết về hiệu sách hư cấu này trong phần tiếp theo như một điều kiện tiên quyết cho mục tiêu làm việc với Elasticsearch của chúng ta.

* + 1. Một hiệu sách trực tuyến

Để chứng minh các tính năng của Elasticsearch, chúng ta hãy sử dụng một hiệu sách hư cấu bán sách kỹ thuật trực tuyến. Tất cả những gì chúng ta muốn làm là tạo một danh mục sách và viết một số truy vấn để tìm kiếm trong số đó.

**22 CPHẦN2*Bắt đầu***

**GHI CHÚ**Mã được trình bày trong chương này có sẵn trong kho lưu trữ GitHub của cuốn sách (<http://mng.bz/2Dyw>) và trên trang web của cuốn sách ([www.manning](http://www.manning.com/books/elasticsearch-in-action-second-edition)

[.com/books/elasticsearch-in-action-second-edition](http://www.manning.com/books/elasticsearch-in-action-second-edition)). Thực hiện theo các hướng dẫn được liệt kê trong kho lưu trữ để lập chỉ mục dữ liệu.

Mô hình dữ liệu cho ứng dụng hiệu sách của chúng tôi rất đơn giản. Chúng tôi có một cuốn sách làm thực thể, với một vài thuộc tính như tiêu đề, tác giả, v.v. như được mô tả trong bảng 2.1. Chúng tôi không cần phải làm phức tạp mọi thứ bằng cách tạo ra các thực thể phức tạp; thay vào đó, chúng tôi sẽ tập trung vào mục tiêu có được trải nghiệm thực tế với Elasticsearch.

**Bảng 2.1 Mô hình dữ liệu cho mộtsáchthực thể**

|  |  |  |
| --- | --- | --- |
| **Cánh đồng** | **Giải thích** | **Ví dụ** |
| tiêu đề tác giả ngày phát hành  amazon\_rating giá bán chạy nhất | Tên sách Tác giả sách Dữ liệu phát hành  Đánh giá trung bình trên Amazon  Cờ chứng minh cuốn sách này là sách bán chạy nhất  Đối tượng bên trong với cá nhângiá cả bằng ba loại tiền tệ | "Java hiệu quả" "Joshua Bloch" 01-06-2001  4.7  "giá" thực sự:{  "USD":9,95,  "gbp":7.95,  "eur":8.95  } |

Elasticsearch là kho dữ liệu tài liệu và nó mong đợi các tài liệu được trình bày ở định dạng JSON. Vì chúng ta cần lưu trữ sách của mình trong Elasticsearch, chúng ta phải mô hình hóa các thực thể của mình dưới dạng tài liệu dựa trên JSON. Chúng ta có thể biểu diễn một cuốn sách trong tài liệu JSON như thể hiện trong hình 2.1.

**Tiêu đề và tác giả của cuốn sách, cả hai**

**thông tin văn bản**

**Đánh giá của cuốn sách trong dữ liệu dấu phẩy động**

**Một đối tượng bên trong-bên trong đại diện cho giá cá nhân**

**của cuốn sách**

{

"title":"Java hiệu quả", "author":"Joshua Bloch", "release\_date":"2001-06-01", "amazon\_rating":4.7, "best\_seller":true, "prices": {

"USD":9,95,

"gbp":7.95,

"eur":8.95

}

}

**Tài liệu sách**

**Ngày phát hành của cuốn sách**

**Một cờ boolean để chỉ ra trạng thái của cuốn sách**

**Giá riêng của cuốn sách trong**

**các loại tiền tệ khác nhau**

**Nhân vật2.1 Một biểu diễn JSON của mộtsáchthực thể**

***2.1 Chuẩn bị Elasticsearch vớidữ liệu* 23**

Định dạng JSON biểu diễn dữ liệu theo cặp tên-giá trị đơn giản. Đối với ví dụ của chúng tôi, tiêu đề (tên) của cuốn sách là Effective Java và tác giả (dưới dạng giá trị) là Joshua Bloch. Chúng tôi có thể thêm các trường bổ sung (bao gồm các đối tượng lồng nhau) vào tài liệu: ví dụ, chúng tôi đãthêm giá vào như một đối tượng lồng nhau.

Bây giờ chúng ta đã có ý tưởng về hiệu sách và mô hình dữ liệu của nó, đã đến lúc bắt đầu điền Elasticsearch với một tập hợp sách để tạo một kho sách. Chúng ta thực hiện việc này trong phần tiếp theo.

* + 1. Lập chỉ mục tài liệu

Để làm việc với máy chủ, chúng ta cần lập chỉ mục dữ liệu của máy khách vào Elasticsearch. Có một số cách chúng ta có thể đưa dữ liệu vào Elasticsearch trong thế giới thực: tạo bộ điều hợp để truyền dữ liệu từ cơ sở dữ liệu quan hệ, trích xuất dữ liệu từ hệ thống tệp, phát trực tuyến sự kiện từ nguồn thời gian thực, v.v. Bất kể lựa chọn nào chúng ta đưa ra cho nguồn dữ liệu của mình, chúng ta đều gọi API RESTful của Elasticsearch từ ứng dụng máy khách để tải dữ liệu vào Elasticsearch.

Bất kỳ máy khách dựa trên REST nào (cURL, Postman, máy khách REST nâng cao, mô-đun HTTP cho JavaScript/NodeJS, SDK ngôn ngữ lập trình, v.v.) đều có thể giúp chúng ta giao tiếp với Elastic-tìm kiếm thông qua API. May mắn thay, Elastic có một sản phẩm thực hiện chính xác điều này (và hơn thế nữa): Kibana. Kibana là một ứng dụng web có giao diện người dùng phong phú, cho phép người dùnglập chỉ mục, truy vấn, trực quan hóa và làm việc với dữ liệu. Đây là tùy chọn ưa thích của chúng tôi và chúng tôi sử dụng Kibana rộng rãi trong cuốn sách này.

**Truy cập RESTful**

Giao tiếp với Elasticsearch diễn ra thông qua các API RESTful dựa trên JSON. Trong thế giới kỹ thuật số hiện tại, bạn rất khó có thể tìm thấy một ngôn ngữ lập trình nào không hỗ trợ truy cập các dịch vụ RESTful. Trên thực tế, việc thiết kế Elasticsearch với các API được hiển thị dưới dạng các điểm cuối RESTful dựa trên JSON là một lựa chọn thông minh, vì nó cho phép áp dụng không phụ thuộc vào ngôn ngữ lập trình.

**DTÀI LIỆUGiao diện lập trình ứng dụng (API)S**

API tài liệu của Elasticsearch giúp tạo, xóa, cập nhật và truy xuất tài liệu. API có thể truy cập thông qua HTTP transport bằng các hành động RESTful. Nghĩa là, để lập chỉ mục một tài liệu, chúng ta cần sử dụng HTTP PUT hoặc POST (sẽ nói thêm về POST sau) trên một điểm cuối. Hình 2.2 hiển thị cú pháp của định dạng URL đầy đủ cho phương thức HTTP PUT.

Như bạn có thể thấy, URL bao gồm một số thành phần:

* + - * Một hành động HTTP nhưĐẶT,LẤY, hoặcBƯU KIỆN
      * Tên máy chủ và cổng của máy chủ
      * Tên chỉ mục
      * Điểm cuối của API tài liệu (\_tài liệu)
      * ID tài liệu
      * Yêu cầu cơ thể

**24 CPHẦN2*Bắt đầu***

**Phương pháp HTTP**

**Máy chủ Elasticsearch đang chạy trên một cổng. Ví dụ: localhost:9200**

**Tên của chỉ số**

**Điểm cuối của URL API tài liệu (mặc định là\_tài liệu)**

**ID tài liệu**

**(ĐẶT/GỬI/LẤYvân vân)**

<PHƯƠNG\_PHÁP\_HTTP> <SERVER:PORT>/<INDEX\_NAME>/\_doc/<DOC\_ID>

{

#Nội dung yêu cầu

}

**Nội dung yêu cầu ở định dạng JSON**

**Hình 2.2 Điểm cuối gọi URL Elasticsearch sử dụng phương thức HTTP**

Elasticsearch API chấp nhận một tài liệu JSON làm nội dung yêu cầu, do đó, cuốn sách chúng ta muốn lập chỉ mục phải đi kèm với yêu cầu này. Ví dụ: khối mã sau đây lập chỉ mục một tài liệu sách có ID 1 vào một chỉ mục sách.

**Liệt kê 2.1 Lập chỉ mục asáchtài liệu vàosáchchỉ số**

ĐẶT sách/\_doc/1

{

"title":"Java hiệu quả", "author":"Joshua Bloch", "release\_date":"2001-06-01", "amazon\_rating":4.7, "best\_seller":true, "prices": {

"USD":9,95,

"gbp":7.95,

"eur":8.95

**Mục lục là sách và ID tài liệu là 1.**

**Nội dung của yêu cầu bao gồm dữ liệu JSON.**

}

}

Có thể hơi choáng ngợp nếu bạn lần đầu tiên thấy một yêu cầu như thế này, nhưng tin tôi đi—nó không khó đến vậy khi bạn phân tích nó. Dòng đầu tiên là lệnh cho Elasticsearch biết phải làm gì với yêu cầu. Chúng ta nói rằng hãy đưa tài liệu sách (được đính kèm dưới dạng phần thân của yêu cầu) vào một chỉ mục (hãy tưởng tượng một chỉ mục như một bảng trong cơ sở dữ liệu; đó là một tập hợp để chứa tất cả các tài liệu sách) được gọi là sách. Cuối cùng, khóa chính của sách được biểu thị bằng ID 1.

Chúng ta cũng có thể sử dụng cURL (nếu bạn không biết về cURL, đây là công cụ truyền dữ liệu dòng lệnh, thường được sử dụng để trao đổi với nhiều dịch vụ khác nhau trên internet) cho cùng một yêu cầu để lưu trữ tài liệu sách vào Elasticsearch. Điều này sẽ được thảo luận sau.

**TRONG****HÁT CĐịa chỉ URL**

Chúng ta cũng có thể sử dụng cURL để làm việc với Elasticsearch và lập chỉ mục tài liệu sách của chúng ta vào chỉ mục sách của chúng ta. Lưu ý rằng chúng ta thích Kibana hơn cURL trong cuốn sách này, vì vậy tất cả các mã là

***2.1 Chuẩn bị Elasticsearch vớidữ liệu* 25**

được trình bày trong các tập lệnh có thể thực thi bằng trình soạn thảo mã Kibana. Lệnh cURL đầy đủ được hiển thị trong hình 2.3 (Kibana ẩn URL đầy đủ).

**Gọi API**

**sử dụng tiện ích cURL**

**Giao thức HTTPĐẶTđể chèn một tài liệu mới**

**Tên máy chủ và cổng máy chủ**

**sáchchỉ sốtên**

curl -XPUT "http://localhost:9200/books/\_doc/1"

-H 'Loại nội dung: ứng dụng/json' -d'

{

"title":"Java hiệu quả", "author":"Joshua Bloch", "release\_date":"2001-06-01", "amazon\_rating":4.7, "best\_seller":true, "prices": {

"USD":9,95,

"gbp":7.95,

"eur":8.95

**Mã số sách**

**Xác định loại nội dung JSON**

**Tài liệu sáchnhư là cơ quan yêu cầu**

}

}'

**Hình 2.3 Điểm cuối gọi URL Elasticsearch sử dụng cURL**

Như bạn có thể thấy, cURL mong đợi chúng ta cung cấp các tham số yêu cầu như loại nội dung, tài liệu, v.v. Vì các lệnh cURL là lệnh đầu cuối (gọi dòng lệnh), việc chuẩn bị yêu cầu rất phức tạp và đôi khi dễ xảy ra lỗi.

May mắn thay, Kibana cho phép chúng ta bỏ qua các chi tiết máy chủ, loại nội dung và các tham số khác, do đó lệnh gọi trông giống như trong hình 2.4. Như tôi đã đề cập, trong cuốn sách này, chúng ta sử dụng Kibana để tương tác với Elasticsearch.

**cURLyêu cầuYêu cầu Kibana**

**curl -XPUT "http://localhost:9200/books/\_doc/1" -H 'Loại nội dung: application/json' -d'**

{

"title":"Java hiệu quả",

...

}'

**ĐẶT sách/\_doc/1**

{

"title":"Java hiệu quả",

...

}

**Cung cấp thông tin chi tiết về máy chủ và thông tin tiêu đề trong yêu cầu cURL.**

**Chi tiết máy chủ và thông tin tiêu đề được Kibana thêm vào yêu cầu.**

**Hình 2.4 Chuyển đổi từ lệnh cURL sang lệnh yêu cầu của Kibana**

**26 CPHẦN2*Bắt đầu***

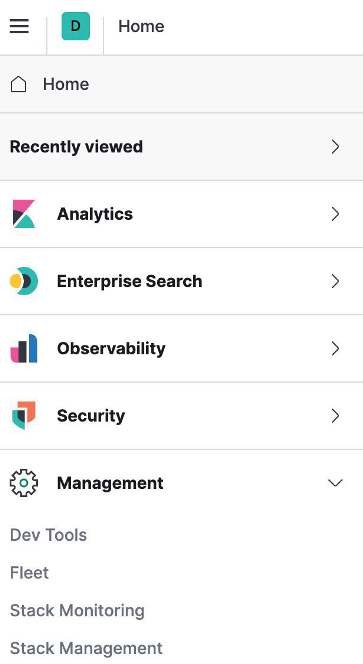
Đã đến lúc lập chỉ mục cho tài liệu đầu tiên của chúng ta. Hãy chuyển sang phần tiếp theo để thực hiện hành động.

* + 1. Lập chỉ mục tài liệu đầu tiên của chúng tôi

Để sử dụng Kibana để lập chỉ mục tài liệu, chúng ta sẽ đến ứng dụng Dev Tools của Kibana để thực hiện truy vấn. Chúng ta dành nhiều thời gian trên trang Dev Tools, vì vậy đến cuối cuốn sách này, bạn sẽ khá quen thuộc với nó!

Giả sử Elasticsearch và Kibana đang chạy trên máy cục bộ của bạn, hãy truy cập bảng điều khiển Kibana tại http:/./localhost:5601. Ở góc trên bên trái là menu chính với các liên kết và liên kết phụ. Đối với mục đích của chúng tôi, hãy điều hướng đến Quản lý > Công cụ phát triển, như thể hiện trong hình 2.5.

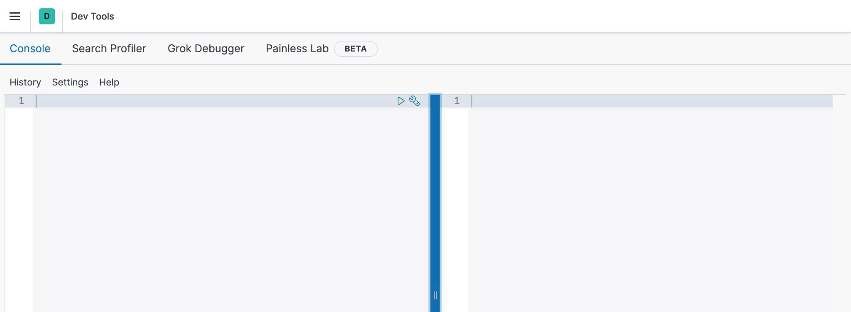
**Hình 2.5 Truy cập trang điều hướng Kibana Dev Tools**



Vì đây có thể là lần đầu tiên bạn truy cập trang này, hãy để tôi giải thích các thành phần. Khi chúng ta điều hướng đến trang này (xem hình 2.6), trình soạn thảo mã sẽ mở ra, hiển thị hai ngăn. Ở ngăn bên trái, chúng ta viết mã với cú pháp đặc biệt do Elasticsearch cung cấp. Sau khi viết một đoạn mã, chúng ta có thể gọi URL trong đoạn mã bằng cách nhấp vào nút Phát ở giữa trang (hình tam giác hướng sang phải).

***2.1 Chuẩn bị Elasticsearch vớidữ liệu* 27**

**Viết mã của bạn vào khung bên trái và nhấp vào nút Phát.**



**Thực hiện truy vấn bằng cách nhấp vào nút Phát.**

**Khi truy vấn của bạn được thực hiện, phản hồi sẽ hiển thị ở khung bên phải.**

**Hình 2.6 Trình soạn thảo mã Dev Tools của Kibana**

Kibana phải được kết nối với một phiên bản Elasticsearch (chi tiết phiên bản được xác định trong tệp cấu hình Kibana) trước khi sẵn sàng hoạt động. Kibana sử dụng chi tiết của máy chủ để gói đoạn mã với các URL Elasticsearch phù hợp và gửi đến máy chủ để thực thi.

Để lập chỉ mục tài liệu của chúng ta cho Elasticsearch, hãy tạo một đoạn mã (mã được đưa ra trong danh sách 2.1). Hình 2.7 hiển thị yêu cầu lập chỉ mục và phản hồi.

**Yêu cầuPhản hồi**

ĐẶT sách/\_doc/1

{

"title":"Java hiệu quả", "author":"Joshua Bloch", "release\_date":"2001-06-01", "amazon\_rating":4.7, "best\_seller":true, "prices": {

"USD":9,95,

"gbp":7.95,

"eur":8.95

}

}

{

"\_index" : "sách",

"\_type" : "\_doc",

"\_id" : "1",

"\_version" : 1, "result" : "đã tạo", "\_shards" : {

"tổng cộng" : 2,

"thành công" : 1,

"thất bại" : 0

},

"\_seq\_no" : 0,

"\_primary\_term" : 1

}

**Yêu cầu lập chỉ mục với tài liệu sách của chúng tôi được gửi đến Elasticsearch**

**Phản hồi xác nhận việc tạo tài liệu của chúng tôi trongsáchchỉ mục có ID là 1**

**Hình 2.7 Lập chỉ mục một tài liệu trong Kibana (bên trái) và phản hồi từ Elasticsearch (bên phải)**

**28 CPHẦN2*Bắt đầu***

Khi mã đã sẵn sàng, hãy nhấp vào nút Phát (hình 2.6). Kibana gửi yêu cầu này đến máy chủ Elasticsearch. Sau khi nhận được yêu cầu, Elasticsearch xử lý yêu cầu đó (hình 2.7), lưu trữ thông báo và gửi phản hồi trở lại máy khách (Kibana). Bạn có thể xem phản hồi trên bảng điều khiển bên phải trong trình soạn thảo.

Phản hồi là một tài liệu JSON. Trong hình 2.7, thuộc tính result cho biết tài liệu đã được tạo thành công. Bạn cũng sẽ thấy mã trạng thái HTTP là “200 OK” cho biết yêu cầu đã thành công. Phản hồi có một số siêu dữ liệu bổ sung (chẳng hạn như chỉ mục, ID và phiên bản tài liệu), điều này tự giải thích được. Chúng ta sẽ thảo luận chi tiết về các thành phần của yêu cầu và phản hồi trong các chương sắp tới, nhưng hãy để tôi giải thích luồng yêu cầu và phản hồi ở cấp độ cao tại đây. Toàn bộ quy trình được minh họa trong hình 2.8.

ĐẶT sách/\_doc/1

{

...

}

Lập chỉ mụcyêu cầu Phản hồi máy chủ

**Cụm**

Tài liệu

{

...

"\_index" : "sách",

"\_type" : "\_doc",

"kết quả" : "đã tạo",

...

}

Xác nhận phản hồi của khách hàng

Lưu trữ hệ thống tập tin

Duy trì hệ thống tập tin bền vững

Lưu trữ trong bộ đệm trong bộ nhớ

Tạo cấu trúc dữ liệu (chỉ mục đảo ngược, cây BKD)

Phân tích và xử lý tài liệu (phân loại và chuẩn hóa)

**Hình 2.8 Luồng yêu cầu và phản hồi của Elasticsearch ở cấp độ cao**

Các bước thực hiện như sau:

**1**Kibana gửi yêu cầu tới máy chủ Elasticsearch với các tham số đầu vào cần thiết.

**2** Khi nhận được yêu cầu, máy chủ

* Phân tích dữ liệu tài liệu và để truy cập nhanh hơn, lưu trữ dữ liệu đó trong một chỉ mục đảo ngược (một cấu trúc dữ liệu hiệu suất cao, là trái tim và linh hồn của công cụ tìm kiếm)
* Tạo một chỉ mục mới (lưu ý rằng chúng tôi không tạo chỉ mục trước) và lưu trữ tài liệu
* Tạo các ánh xạ và lược đồ dữ liệu cần thiết
* Gửi phản hồi lại cho khách hàng

**3**Kibana nhận được phản hồi và hiển thị ở bảng bên phải (hình 2.6) để chúng ta sử dụng.

* 1. ***Chuẩn bị Elasticsearch vớidữ liệu* 29**

Chúng tôi đã lập chỉ mục cho tài liệu đầu tiên của mình cho Elasticsearch! Việc lập chỉ mục cho một tài liệu tương tự như việc chèn một bản ghi vào cơ sở dữ liệu quan hệ.

**CCÁC HỌC VIÊN YÊU CẦU**

Yêu cầu chúng tôi đã gửi (PUT books/\_doc/1) cần được chắt lọc một chút. Yêu cầu này có năm phần và chúng tôi sẽ nhanh chóng xem qua chúng ở đây (nếu bạn muốn, bạn có thể bỏ qua phần này và quay lại sau khi đọc hết phần còn lại của chương):

* ĐẶT*phương pháp*—ĐẶTlà một động từ HTTP (còn được gọi là một phương thức) cho biết chúng ta đang gửi một yêu cầu đến máy chủ để tạo một tài nguyên (asáchtài liệu, trong trường hợp này). Elasticsearch sử dụng giao thức HTTP cho các lệnh gọi API RESTful của nó, vì vậy chúng tôi mong đợiĐẶT,BƯU KIỆN,LẤY,XÓA BỎvà các phương thức HTTP chuẩn khác trong cú pháp URL yêu cầu.
* sách*chỉ số*—Cácsáchmột phần của URL của chúng tôi được gọi là chỉ mục: một thùng để thu thập tất cảsáchtài liệu. Nó tương tự như một bảng trong cơ sở dữ liệu quan hệ. Chỉsáchtài liệu được lưu trữ trong nàysáchchỉ số (mặc dùsáchvề mặt lý thuyết, index chỉ được dùng để lưu trữ các tài liệu dạng sách, không có gì ngăn cản chúng ta lập chỉ mục các loại tài liệu khác nữa—chúng ta có trách nhiệm không trộn lẫn các loại tài liệu).
* \_tài liệu*điểm cuối*—Điểm cuối là một phần không đổi của đường dẫn liên quan đến hoạt động đang được thực hiện. Trong các phiên bản trước của Elasticsearch (phiên bản < 7.0),\_tài liệu'nơi được sử dụng để được lấp đầy bởi loại ánh xạ của tài liệu.các kiểu ánh xạ đã bị loại bỏ và\_tài liệuthay thế chúng bằng đường dẫn điểm cuối hằng số chung trong URL (xem thanh bên “Các loại tài liệu và điểm cuối \_doc” để biết thêm).
* *ID tài liệu*—Số lượng1trong URL của chúng tôi biểu thị ID của tài liệu. Nó giống như khóa chính cho một bản ghi trong cơ sở dữ liệu. Chúng tôi sử dụng mã định danh này để truy xuất tài liệu.
* *Yêu cầu cơ thể*—Phần thân của yêu cầu là biểu diễn JSON của dữ liệu sách. Elasticsearch mong đợi tất cả dữ liệu được gửi dưới dạng tài liệu định dạng JSON. Nó cũng hỗ trợ các đối tượng lồng nhau được gửi theo định dạng đối tượng lồng nhau của JSON.

**Các loại tài liệu và điểm cuối \_doc**

Trước phiên bản 7.x của Elasticsearch, một chỉ mục có thểgiữ nhiều loại thực thể (ví dụ, mộtsáchmục lục có thể chứa không chỉ sách mà còn cả các bài đánh giá sách, sáchbán hàng, hiệu sách, v.v.). Việc có tất cả các loại tài liệu trong một chỉ mục duy nhất dẫn đến sự phức tạp. Ánh xạ trường được chia sẻ trên nhiều loại, dẫn đến lỗi cũng như sự thưa thớt dữ liệu. Để tránh các vấn đề với các loại và việc quản lý chúng, Elastic đã quyết định loại bỏ các loại.

Trong các phiên bản trước, URL gọi với một loại trông như thế này:*<tên\_chỉ\_mục>/*

*<kiểu>/<id>*(ví dụ: books/book/1). Các kiểu đã bị loại bỏ trong phiên bản 7.x. Bây giờ, một chỉ mục được mong đợi sẽ dành riêng cho một và chỉ một kiểu:\_tài liệulà một kết thúc-điểm được khắc vào URL. Chúng ta tìm hiểu về việc loại bỏ các loại trong chương 5.

Ý chính là chúng tôi đã sử dụng phương thức HTTP PUT để lập chỉ mục một tài liệu có ID là 1

vào mục lục sách cho Elasticsearch. Khi lập chỉ mục tài liệu đầu tiên của chúng tôi, bạn đã

**30 CPHẦN2*Bắt đầu***

lưu ý rằng chúng tôi không tạo lược đồ? Chúng tôi đã lập chỉ mục tài liệu bằng cách gọi API, nhưng Elasticsearch không bao giờ yêu cầu chúng tôi xác định lược đồ dữ liệu trước khi lập chỉ mục dữ liệu.

Không giống như cơ sở dữ liệu quan hệ, Elasticsearch không yêu cầu chúng ta tạo lược đồ trước (cái này làcđồng minh hiện tại *không có sơ đồ*) Và là vui mừngtcái rút ra cái sơ đồ từ tài liệu đầu tiên được lập chỉ mục. Nó cũng tạo ra chỉ mục (chính xác là chỉ mục sách).

Tóm lại là Elasticsearch không thích cản trở chúng ta trong quá trình phát triển. Điều đó có nghĩa là chúng ta phải tuân theo phương pháp hay nhất là tạo sơ đồ của riêng mình trước trong môi trường sản xuất; chúng ta sẽ tìm hiểu thêm về điều này khi chúng ta tiến triển trong cuốn sách.

Trong phần này, chúng ta đã lập chỉ mục thành công một tài liệu. Hãy làm theo quy trình tương tự và lập chỉ mục thêm một vài tài liệu nữa.

* + 1. Đang lập chỉ mục thêm tài liệu

Để các ví dụ tiếp theo hoạt động, chúng ta cần thêm một vài tài liệu để lập chỉ mục. Hãy đến trình soạn thảo mã của Kibana và viết mã sau cho hai tài liệu nữa.

**Liệt kê 2.2 Hai cái nữasáchtài liệu để lập chỉ mục**

PUT sách/\_doc/2

{

"title":"Java Core Tập I - Cơ bản", "author":"Cay S. Horstmann", "release\_date":"2018-08-27", "amazon\_rating":4.8,

"best\_seller":true, "giá": {

"USD":19,95,

"gbp":17,95,

"eur":18,95

**Tài liệu cuốn sách thứ hai**

}

} **Cuốn sách thứ ba**

PUT sách/\_doc/3

{

**tài liệu**

"title":"Java: Hướng dẫn cho người mới bắt đầu", "author":"Herbert Schildt", "release\_date":"2018-11-20", "amazon\_rating":4.2, "best\_seller":true,

"giá": {

"USD":19,99,

"gbp":19,99,

"eur":19,99

}

}

Thực hiện các truy vấn trong danh sách này để lập chỉ mục chúng vào chỉ mục sách. Bây giờ chúng ta đã lập chỉ mục một số tài liệu, hãy cùng tìm hiểu cách chúng ta có thể truy xuất hoặc tìm kiếm chúng.

#### Lấy dữ liệu

Chúng tôi đã chuẩn bị máy chủ của mình, mặc dù với một tập hợp tài liệu rất hạn chế. Đã đến lúc hành động để kiểm tra cách chúng tôi có thể truy xuất, tìm kiếm và tổng hợp các tài liệu này. Từ phần tiếp theo trở đi, chúng tôi sẽ thực hành bằng cách thực hiện truy vấn và truy xuất dữ liệu.

***2.2 Truy xuấtdữ liệu* 31**

Hãy bắt đầu với yêu cầu cơ bản là tìm ra tổng số sách chúng ta có trong kho (trong mục lục sách, trong trường hợp này). Elasticsearch đưa ra API \_count để đáp ứng yêu cầu này. Khá đơn giản, như chúng ta thấy trong phần tiếp theo.

* + 1. Đếmtài liệu

Biết tổng số tài liệu trong một chỉ mục là một yêu cầu được đáp ứng bởi

API \_count. Gọi điểm cuối \_count trên chỉ mục sách sẽ cho kết quả là số lượng tài liệu được lưu trong chỉ mục đó.

**Liệt kê 2.3 Đếm tài liệu bằng cách sử dụng\_đếmGiao diện lập trình ứng dụng (API)**

NHẬN sách/\_số lượng

**Đếm số lượng sách**

Lệnh này sẽ trả về kết quả như trong hình 2.9.

NHẬN sách/\_số lượng

{

**"đếm" : 3,**

"\_mảnh vỡ" : {

"tổng cộng" : 1,

"thành công" : 1,

"bỏ qua" : 0,

"thất bại" : 0

**Cácđếmbiến chỉ ra tổng số tài liệu.**

}

}

**Hình 2.9 Phản hồi JSON cho một\_đếmGọi API**

Biến đếm được tô sáng cho biết tổng số tài liệu trong chỉ mục sách. Chúng ta cũng có thể sử dụng cùng một API để lấy từ nhiều chỉ mục cùng một lúc, bằng cách thêm các chỉ mục được phân tách bằng dấu phẩy như GET books1,books2/\_count. (Lưu ý rằng nếu bạn thực hiện-Nếu truy vấn này đúng ngay lúc này, hệ thống sẽ đưa ra lỗi index\_not\_found\_exception cho biết chỉ mục books2 không tồn tại.)

Chúng ta cũng có thể lấy số lượng tài liệu trong tất cả các chỉ mục bằng cách thực hiện lệnh gọi GET \_count. Lệnh này trả về tất cả các tài liệu có sẵn, bao gồm cả chỉ mục hệ thống và chỉ mục ẩn. Trong phần tiếp theo, chúng ta sử dụng API để lấy tài liệu.

* + 1. Lấy lại tài liệu

Cũng giống như chúng ta gửi tiền vào tài khoản ngân hàng, chúng ta đã lập chỉ mục các tài liệu vào Elastic-search trước đó. Bây giờ chúng ta cần truy cập các tài liệu này (giống như chúng ta rút tiền từ tài khoản ngân hàng).

Mỗi tài liệu được lập chỉ mục đều có một mã định danh duy nhất; một số mã định danh do chúng tôi cung cấp và một số khác do Elasticsearch tạo ra. Các thao tác để truy xuất

**32 CPHẦN2*Bắt đầu***

tài liệu phụ thuộc vào quyền truy cập vào ID tài liệu. Đây là các hoạt động chúng tôi đề cập trong phần này:

* Nhận một tài liệu duy nhất, được cấp một ID
* Nhận nhiều tài liệu, cung cấp một bộ ID
* Nhận tất cả các tài liệu cùng một lúc (không cần ID, không cần tiêu chí tìm kiếm)

Chúng ta có thể lấy một tài liệu duy nhất từ ​​Elasticsearch bằng cách gọi lệnh gọi API tài liệu được cung cấp một ID và chúng ta có thể sử dụng một truy vấn API khác (id) để lấy nhiều tài liệu. Chúng tacũng có thể sử dụng \_search API để tìm kiếm tất cả tài liệu cùng một lúc.

**RLẤY MỘT TÀI LIỆU DUY NHẤT**

Tương tự như việc lấy một bản ghi từ cơ sở dữ liệu bằng khóa chính, việc lấy tài liệu từ Elasticsearch rất đơn giản, miễn là chúng ta có ID của tài liệu. Để làm như vậy, chúng ta đưa ra lệnh GET trong bảng điều khiển Kibana với URL API và ID tài liệu. Định dạng chung để lấy tài liệu có mã định danh là GET <index>/

\_doc/<id>. Để lấy một tài liệu duy nhất, hãy đưa ra lệnh GET, cung cấp ID 1.

**Liệt kê 2.4 Truy xuất một tài liệu riêng lẻ theo ID của nó**

NHẬN sách/\_doc/1

Nếu lệnh này được thực thi thành công, bạn sẽ thấy phản hồi ở phía bên phải của trình soạn thảo mã, như thể hiện trong hình 2.10.

Phản hồi chứa hai thông tin: tài liệu gốc, dưới thẻ \_source và siêu dữ liệu của tài liệu này (chẳng hạn như chỉ mục, id, tìm thấy,

**Vấn đềMộtLẤYyêu cầu lấy tài liệu**

{

"\_index" : "sách",

"\_type" : "\_doc",

"\_id" : "1",

"\_phiên bản" : 1,

**với ID 1**

...

"\_nguồn" : {

**Siêu dữ liệu**

NHẬN sách/\_doc/1

"title" : "Java hiệu quả",

...

**Phản hồi trả về dữ liệu nguồn cũng như siêu dữ liệu**

"giá" : {

"USD" : 9,95,

...

}

}

}

**Tài liệu gốc (dữ liệu nguồn)**

**Hình 2.10 Lấy mộtsáchtài liệu theo ID**

***2.2 Truy xuấtdữ liệu* 33**

phiên bản, v.v.). Nếu chúng ta không muốn tất cả siêu dữ liệu mà chỉ muốn nguồn gốc, hãy chạy lại truy vấn bằng cách thay đổi điểm cuối \_doc bằng \_source: GET books/\_source/1.

**RLẤY NHIỀU TÀI LIỆU BẰNG****NHẬN DẠNG**

Khi chúng ta cần lấy một tập hợp các tài liệu được cung cấp một tập hợp các mã định danh, chúng ta có thể sử dụng truy vấn id. Truy vấn này sẽ lấy các tài liệu được cung cấp một tập hợp các ID tài liệu. Đây là cách đơn giản hơn nhiều để lấy tất cả chúng cùng một lúc, với danh sách các ID tài liệu.

Tuy nhiên, một điều cần lưu ý là không giống như các truy vấn khác sử dụng API tài liệu đểlấy tài liệu, truy vấn id sử dụng API tìm kiếm—cụ thể là điểm cuối \_search. Hãy cùng xem cách thực hiện.

**GHI CHÚ**Elasticsearch cung cấp một ngôn ngữ miền cụ thể (DSL) để viết truy vấn, thường được gọi là Query DSL. Đây là một ngôn ngữ viết truy vấn đơn giản dựa trên JSON, được sử dụng rộng rãi để viết truy vấn trong Kibana. Hầu hết các truy vấn trong chương này (và phần còn lại của cuốn sách) đều dựa trên Query DSL.

**Liệt kê 2.5 Lấy nhiều tài liệu bằng cách sử dụngidtruy vấn**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"ID": {

"giá trị": [1,2,3]

}

}

}

Nội dung yêu cầu trong truy vấn được xây dựng bằng cách sử dụng đối tượng truy vấn, có đối tượng bên trong có tên là ids. ID tài liệu là các giá trị được cung cấp trong truy vấn dưới dạng một mảng. Phản hồi chỉ ra các lần truy cập thành công (kết quả), như thể hiện trong hình 2.11, trả về ba tài liệu có các ID đó.

**MỘTLẤYyêu cầu trên một\_tìm kiếmđiểm cuối với truy vấn để lấy nhiều tài liệu**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"ID": {

"giá trị": [1,2,3]

}

}

"lượt truy cập" : [{

"\_index" : "sách",

"\_type" : "\_doc",

**"\_id" : "1",**

"\_nguồn" : {

"title" : "Java hiệu quả",

..

}

},{

"\_index" : "sách",

**"\_id" : "2",**

} **Cácidtruy vấn mong đợi một mảng có ID tài liệu.**

**Phản hồi trở lại**

**cả batài liệu.** ]

...

}

...

**Hình 2.11 Lấy lại các tài liệu có một tập hợp ID bằng cách sử dụngidtruy vấn, trong đó gọi một\_tìm kiếmđiểm cuối**

**34 CPHẦN2*Bắt đầu***



**Loại bỏ dữ liệu nguồn gốc trong phản hồi**

Đôi khi chúng ta có thể tránh làm tắc nghẽn mạng và lãng phí băng thông bằng cách không truy xuấting dữ liệu nguồn không bắt buộc. Ví dụ, hãy tưởng tượng nếu tài liệu nguồn của chúng ta có 500 trường nhưng không phải tất cả đều có liên quan để truy xuất. Chúng ta có thể kiểm soát những trường nào được gửi đến máy khách và những trường nào không bằng cách điều chỉnh các trường trong phản hồi.

Chúng ta cũng có thể thiết lập lá cờ"\_source": saitrên yêu cầu để ngăn chặn hoàn toàn việc gửi dữ liệu nguồn trong phản hồi. Cờ được đặt ở cấp gốc (giống như ở cấp truy vấn):

NHẬN sách/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"ID": {

"giá trị": [1,2,3]

}

}

}

**Đang đàn áp**

**nguồn gốc**

Tất nhiên, cách lấy lại tài liệu này rất cồng kềnh. Hãy tưởng tượng nếu chúng ta muốn lấy lại 1.000 tài liệu với 1.000 ID! May mắn thay, chúng ta không cần sử dụngtruy vấn id. Thay vào đó, chúng ta có thể sử dụng các tính năng tìm kiếm khác mà chúng ta sẽ đề cập sau trong sách.

Cho đến nay, chúng ta đã xem xét một API tài liệu có chức năng tìm kiếm tài liệu bằng ID. Đây không hẳn là một tính năng tìm kiếm mà là truy xuất dữ liệu. Elasticsearch cung cấp vô số tính năng tìm kiếm để tìm kiếm kết quả dựa trên nhiều điều kiện và tiêu chí khác nhau. Chúng tôi đã dành một số chương để tìm kiếm, từ cơ bản đến nâng cao, ở phần sau của cuốn sách. Nhưng với tư cách là những nhà phát triển tò mò, chúng ta muốn thấy các truy vấn tìm kiếm cấp cao đang hoạt động, phải không? Hãy cùng thảo luận về tập hợp các truy vấn dựa trên các hàm tìm kiếm.

**R****LẤY TẤT CẢ CÁC TÀI LIỆU**

Trong phần trước, chúng ta đã làm việc với điểm cuối \_search cơ bản. Sử dụng cú pháp tương tự, chúng ta có thể viết truy vấn để lấy tất cả sách từ chỉ mục của mình.

**Liệt kê 2.6 Lấy tất cả các tài liệu từsáchchỉ số**

NHẬN sách/\_tìm kiếm

**URL không cần phần nội dung.**

Như bạn có thể thấy, khi bạn tìm kiếm tất cả các bản ghi, bạn không cần phải đính kèm phần thân vàotruy vấn (xem thanh bên trên truy vấn match\_all, giải thích cú pháp của truy vấn này). Phản hồi, được hiển thị trong hình 2.12, trả về ba tài liệu trong chỉ mục sách của chúng tôi như mong đợi.

Đừng lo lắng nếu bạn bối rối với cú pháp truy vấn hoặc các trường phản hồi. Chúng tôi sẽ đề cập đến ngữ nghĩa trong các chương tiếp theo.

* 1. ***Lấy lạidữ liệu* 35**

**Tìm kiếm chung (không có nội dung yêu cầu) sẽ trả về tất cả các cuốn sách từ chỉ mục.**

{

...

"lượt truy cập" : {

"tổng cộng" : {

"giá trị" : 3,

"quan hệ" : "eq"

},

"điểm\_tối\_đa" : 1.0,

"lượt truy cập" : [

{

**Cáclượt truy cậpđối tượng biểu thị số lượng kết quả trả về.**

**Cáclượt truy cậpmảng được tạo thành từ các giá trị thực tế được trả về**

NHẬN sách/\_tìm kiếm

"\_index" : "sách",

"\_type" : "\_doc",

"\_id" : "1",

"\_điểm" : 1.0,

"**\_nguồn**" : {

**kết quả.**

**Cuốn sách thực tế là**

**được bao bọc trong**

**\_nguồnsự vật.**

}

},

...

}

"title": "Java hiệu quả", "author": "Joshua Bloch"

**Hình 2.12 Truy xuất tất cả tài liệu bằng API tìm kiếm**



**Tìm kiếm chung là một dạng rút gọn của truy vấn match\_all**

Truy vấnNHẬN sách/\_tìm kiếmmà bạn đã thấy trước đó là một dạng rút gọn của một đặc biệttruy vấn có tênkhớp\_tất\_cả. Thông thường, một nội dung yêu cầu được thêm vào URL với mộttruy vấnmệnh đề, chẳng hạn như mệnh đề trong danh sách 2.5. Khi chúng ta bỏ qua phần thân truy vấn trong khi gọi\_tìm kiếmđiểm cuối, đó là một hướng dẫn cho Elasticsearch rằng truy vấn làkhớp\_tất\_cảtruy vấn—có nghĩa là*phù hợp với mọi thứ*. Truy vấn đầy đủ được hiển thị ở đây, nhưngchúng ta hiếm khi cần phải viết phần thân nếu chúng ta đang có kế hoạch lấy tất cả các bản ghi mà không có bất kỳtruy vấncác điều khoản:

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"match\_all": { }

}

}

Tuy nhiên, nếu chúng ta cần tăng điểm của tất cả các kết quả vì bất kỳ lý do gì, chúng ta có thể tạokhớp\_tất\_cảtruy vấn và khai báo thêmTăngtham số:

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"khớp\_tất\_cả": {

"Tăng cường":2

}

**Tất cả kết quả sẽ**

**có số điểm là 2**

}

}

Điểm mấu chốt là bất cứ khi nào chúng ta gọi một\_tìm kiếmđiểm cuối không có thân,đó là mộtkhớp\_tất\_cảtruy vấn.

**36 CPHẦN2*Bắt đầu***

Cho đến nay, chúng ta đã thấy các truy vấn không thực sự thể hiện toàn bộ sức mạnh của Elasticsearch; chúng chỉ lấy dữ liệu mà không có nhiều trí thông minh. Thỉnh thoảng, chúng ta có thể muốn đưa ra một truy vấn có tiêu chí: ví dụ, tìm kiếm sách phiên bản đầu tiên của một tác giả cụ thể có xếp hạng chung cao hơn bốn sao. Sức mạnh thực sự của Elasticsearch ẩn trong tính linh hoạt và chiều sâu của chức năng tìm kiếm của nó. Một vài phần tiếp theo sẽ xem xét điều này ở cấp độ cao.

#### Tìm kiếm toàn văn

Sau khi bạn đã lập chỉ mục một số tài liệu, điều quan trọng là có thể tìm thấy những tài liệu đáp ứng các tiêu chí cụ thể. Elasticsearch cung cấp chức năng tìm kiếm để tìm kiếm văn bản phi cấu trúc theo tên truy vấn toàn văn.

**GHI CHÚ**Văn bản phi cấu trúc (còn gọi là văn bản đầy đủ), giống như ngôn ngữ tự nhiên, không tuân theo một lược đồ hoặc mô hình cụ thể như văn bản có cấu trúc. Các công cụ tìm kiếm hiện đại làm việc chăm chỉ trên văn bản đầy đủ để đưa ra kết quả có liên quan. Elasticsearch cung cấp một tập hợp các kết quả phù hợp và các truy vấn khác để làm việc với dữ liệu văn bản đầy đủ.

* + 1. Truy vấn khớp: Sáchbởi một tác giả

Ví dụ, giả sử một độc giả ghé thăm hiệu sách của chúng tôi muốn tìm tất cả các tựa sách do Joshua Bloch biên soạn. Chúng tôi có thể xây dựng truy vấn bằng cách sử dụng API \_search với một kết quả khớptruy vấn: truy vấn \_match giúp tìm kiếm các từ trong toàn bộ văn bản không có cấu trúc hoặc toàn văn. Truy vấn sau đây tìm kiếm các cuốn sách do “Joshua” biên soạn.

**Liệt kê 2.7 Truy vấn sách do một tác giả cụ thể biên soạn**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tác giả": "Joshua"

}

}

}

Trong phần thân yêu cầu, chúng tôi tạo mộttruy vấnđối tượng xác định mộtcuộc thi đấutruy vấn. Trong mệnh đề truy vấn này, chúng tôi yêu cầu Elasticsearch khớp tất cả các tài liệu do Joshua biên soạn trên tất cả các cuốn sách trong chỉ mục của chúng tôi. Đọc truy vấn như thể bạn đang đọc văn xuôi bằng tiếng Anh thông thường và cố gắng lĩnh hội bản chất của nó.

Sau khi truy vấn được gửi đi, máy chủ sẽ phân tích truy vấn, khớp truy vấn với các cấu trúc dữ liệu nội bộ (chỉ mục đảo ngược), lấy tài liệu từ kho lưu trữ tệp và trả về cho máy khách. Trong ví dụ này, máy chủ tìm thấy một cuốn sách do Joshua Bloch biên soạn và trả về, như thể hiện trong hình 2.13.

***2.3 Toàn văntìm kiếm* 37**

**MỘTcuộc thi đấutruy vấn tìm kiếm tất cả các cuốn sách được viết bởi Joshua**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tác giả": "Joshua"

}

}

} **Các cuộc thi đấutruy vấn với mệnh đề tác giả**

**Phản hồi trở lại**

**một tài liệu có sự trùng khớp.**

"lượt truy cập" : [{

"\_index" : "sách",

"\_type" : "\_doc",

"\_id" : "1",

"\_score" : 1.0417082, "\_source" : {

"title": "Java hiệu quả", "author": "Joshua Bloch"

...

}

}]

**Hình 2.13 Đang tải sách của tác giả Joshua**

**Truy vấn tiền tố**

Bạn có thể viết lại truy vấn từ danh sách 2.7 với nhiều kết hợp tìm kiếm khác nhau, chẳng hạn như tên viết thường, tên viết hoa và viết thường, v.v.:

"tác giả":"JoShUa" "tác giả":"joshua" (hoặc "JOSHUA") "tác giả":"Bloch"

Tất cả các truy vấn này đều thành công, nhưng việc tìm kiếm tên rút gọn sẽ không thành công:

"tác giả":"josh"

Để trả về các truy vấn kiểu regex như vậy, chúng tôi sử dụngtiền tốtruy vấn. Chúng ta có thể thay đổi mộtcuộc thi đấu

truy vấn đến mộttiền tốtruy vấn như sau để lấy tên rút gọn:

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"tiền tố": {

"tác giả": "josh"

}

}

}

Giá trị truy vấn phải là chữ thường vìtiền tốtruy vấn là truy vấn ở cấp độ thuật ngữ.

* + 1. Truy vấn khớp với toán tử AND

Nếu chúng ta điều chỉnh truy vấn bằng cách đổi tên thành "Joshua Doe", bạn mong đợi kết quả gì? Chúng tôi không có bất kỳ cuốn sách nào do Joshua Doe viết, vì vậy truy vấn không nên trả về

**38 CPHẦN2*Bắt đầu***

có kết quả nào không? Không phải vậy: cuốn sách do Joshua Bloch viết vẫn sẽ được trả về. Lý do là vì công cụ tìm kiếm đang tìm kiếm tất cả các cuốn sách do Joshua hoặc Doe viết. Toán tử OR được sử dụng ngầm định trong trường hợp này.

Hãy cùng xem cách chúng ta tìm kiếm sách có tên đầy đủ chính xác bằng cách sử dụng toán tử. Hãy lấy ví dụ về truy vấn tìm kiếm tác giả Joshua Schildt (trộn lẫn tên của Joshua và họ của Herbert). Rõ ràng, chúng ta biết chỉ mục của mình không chứa sách do tác giả hư cấu đó viết. Nếu bạn chạy truy vấn trong danh sách 2.7 với "Joshua Schildt", bạn sẽ nhận được hai cuốn sách (vì Elasticsearch tìm kiếm sách do Joshua HOẶC Schildt viết): một của Joshua Bloch và một của Herbert Schildt. Truy vấn đã sửa đổi này được hiển thị tiếp theo.

**Liệt kê 2.8 Tìm kiếm sách do một tác giả hư cấu viết**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tác giả": "Joshua Schildt"

}

**Tìm kiếm một hư cấutác giả sẽ cho ra mắt hai cuốn sách.**

}

}

Chúng ta có thể điều chỉnh truy vấn này để xác định một tham số được gọi là toán tử và đặt rõ ràng thành AND, như danh sách sau đây cho thấy. Có một thay đổi nhỏ đối với truy vấn ở chỗ chúng tacần thêm một đối tượng bao gồm truy vấn và toán tử vào đối tượng tác giả (không giống như trong danh sách 2.7, trong đó chúng ta chỉ cung cấp giá trị truy vấn cho trường).

**Liệt kê 2.9 Truy vấn với mộtVÀtoán tử để lấy các kết quả khớp chính xác**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tác giả": {

**Trường tác giả hiện đã được xác định các thuộc tính bên trong.**

"truy vấn": "Joshua Schildt",

**Câu hỏi của chúng tôi**

"toán tử": "VÀ"

}

}

}

}

**Toán tử AND (mặc định là OR)**

Thực hiện truy vấn không mang lại kết quả nào (không có cuốn sách nào do Joshua Schildt viết).

Theo cùng logic đó, nếu chúng ta muốn lấy một cuốn sách có tiêu đề chính xác, chẳng hạn như Effective Java, hình 2.14 cho thấy mã. Chúng ta sẽ nhận được danh sách tất cả các cuốn sách có một trong hai từ Effective OR Java trong trường tiêu đề nếu chúng ta không thay đổi toán tử. Bằng cách cung cấp toán tử AND để nối cả hai từ, truy vấn sẽ tìm thấy những cuốn sách có cả hai từ tìm kiếm trong trường tiêu đề.

***2.3 Toàn văntìm kiếm* 39**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

**MỘT cuộc thi đấutruy vấnlấy một tiêu đề cụ thể.Vàtoán tử tìm kiếm tiêu đề có cả hai từ trong tiêu đề (“Hiệu quả và Java”).**

"cuộc thi đấu": {

"tiêu đề": {

"query": "Java hiệu quả", "operator": "và"

}

}

}

}

**Hình 2.14 Lấy kết quả khớp chính xác cho một tiêu đề bằng cách sử dụngVàngười điều hành**

Trước khi thực hiện các truy vấn tìm kiếm phức tạp hơn một chút, chúng ta cần làm một việc nhỏ: lập chỉ mục nhiều tài liệu hơn. Cho đến nay, chúng ta chỉ có ba tài liệu và sẽ hữu ích nếu có nhiều hơn cho bất kỳ truy vấn có ý nghĩa nào. Elasticsearch cung cấp một API số lượng lớn tiện lợi (\_bulk) để lập chỉ mục các tài liệu theo số lượng lớn. Hãy đi vòng một chút để lập chỉ mục thêm một vài tài liệu, lần này sử dụng API \_bulk.

* + 1. Lập chỉ mục tài liệu bằng API \_bulk

Chúng ta cần thêm nhiều tài liệu hơn vào kho lưu trữ của mình khi chúng ta bắt đầu làm quen với nhiều truy vấn tìm kiếm khác nhau. Chúng ta có thể làm theo cùng một quy trình sử dụng API tài liệu như chúng ta đã làm trong phần 2.1.2. Tuy nhiên, như bạn có thể tưởng tượng, việc tải nhiều tài liệu riêng lẻ là một quy trình cồng kềnh.

May mắn thay, có một API \_bulk tiện dụng cho phép chúng ta lập chỉ mục các tài liệu cùng lúc. Chúng ta có thể sử dụng Kibana hoặc cURL để thực thi API \_bulk khi lập chỉ mục nhiều tài liệu, nhưng các phương pháp này có sự khác biệt về định dạng dữ liệu. Chúng tôi sẽ thảo luận chi tiết về API \_bulk trong chương 5; ở đây, chúng tôi sẽ tóm tắt một số điểm nổi bật.

**Hoạt động hàng loạt ghi đè dữ liệu hiện có**

Nếu bạn thực hiện\_số lượng lớnhoạt động trên sách, nó lập chỉ mục chúng vào chỉ mục hiện có (sách) được tạo ra ở đầu chương. Mớisáchtài liệu có các trườngchẳng hạn nhưgiá ,xếp hạngvà những thứ khác; cấu trúc đối tượng được làm phong phú thêm bởi những thuộc tính bổ sung này và khác với cấu trúc trước đó.

Nếu bạn không muốn hiện tạisáchchỉ mục cần chạm vào, bạn có thể tạo một chỉ mục mới(sách\_mới, có lẽ) để tránh ghi đè lên chỉ mục hiện có. Để thực hiện việc này, hãy sửa đổi tên chỉ mục trong tệp dữ liệu hàng loạt bằng dòng sau:

{"index":{"\_index":"sách\_mới","\_id":"1"}}

**40 CPHẦN2*Bắt đầu***

***(tiếp theo)***

Đảm bảo tất cả các dòng chỉ mục được cập nhật, không chỉ dòng trên cùng. Ngoài ra,bạn có thể loại bỏ\_chỉ mụctrường hoàn toàn và thêm "chỉ số" vào URL:

POST sách\_mới/\_số lượng lớn

{"chỉ mục":{"\_id":"1"}}

..

Ví dụ trong văn bản cập nhật hiện cósáchchỉ mục thay vì tạo một chỉ mục mới, do đó tất cả các truy vấn trong chương này được thực hiện trênsáchmục lục.

Ví dụ sử dụng các tập dữ liệu được cung cấp cùng với cuốn sách này, có sẵn tại GitHub (<https://github.com/madhusudhankonda/elasticsearch-in-action>) và trang web của cuốn sách (<https://www.manning.com/books/elasticsearch-in-action-second-edition>). Sao chép nội dung của tệp books-kibana-dataset.txt và dán vào Dev Tools của Kibana. Hình 2.15 hiển thị một phần nội dung của tệp.



**Hình 2.15 Lập chỉ mục hàng loạt tài liệu bằng cách sử dụng\_số lượng lớnđiểm cuối**

Thực hiện truy vấn và bạn sẽ nhận được thông báo xác nhận rằng tất cả tài liệu đã được lập chỉ mục thành công.

**Định dạng tài liệu lập chỉ mục hàng loạt trông kỳ lạ**

Nếu bạn nhìn kỹ vào các tài liệu được tải bằng\_số lượng lớn, bạn sẽ thấy một số cú pháp kỳ lạ. Mỗi hai dòng tương ứng với một tài liệu, như thế này:

{"chỉ mục":{"\_id":"1"}}

{"thương hiệu": "Samsung","tên":"UHD","kích thước\_inches":65,"giá":1400}

Dòng đầu tiên là siêu dữ liệu về bản ghi, bao gồm thao tác (chỉ số,xóa bỏ,cập nhật) chúng ta sắp thực hiện (chỉ số, trong trường hợp này), ID tài liệu và chỉ mục nơi bản ghi sẽ đến. Dòng thứ hai là tài liệu thực tế. Chúng ta sẽ xem lại định dạng này trong chương 5 khi thảo luận về các hoạt động của tài liệu.

***2.3 Toàn văntìm kiếm* 41**

Bây giờ chúng ta đã lập chỉ mục thêm một vài tài liệu, chúng ta có thể quay lại đúng hướng và thử nghiệm một số tính năng tìm kiếm khác. Hãy bắt đầu bằng cách tìm kiếm một từ trên nhiều trường.

* + 1. Tìm kiếm trên nhiều trường

Khi khách hàng tìm kiếm thứ gì đó trong thanh tìm kiếm, tìm kiếm không nhất thiết chỉ giới hạn ở một trường. Ví dụ, giả sử chúng ta muốn tìm kiếm tất cả các tài liệunơi từ Java xuất hiện: không chỉ trong trường tiêu đề mà còn trong các trường khác như tóm tắt, thẻ, v.v. Để thực hiện việc này, chúng ta có thể bật tìm kiếm đa trường.

Hãy xem ví dụ về truy vấn tìm kiếm Java trong hai trường: tiêu đề và tóm tắt. Tương tự như truy vấn khớp mà chúng ta đã thấy trước đó, Elasticsearch cung cấp truy vấn multi\_match sẽ phục vụ mục đích của chúng ta. Chúng ta cần cung cấp các từ tìm kiếm trong đối tượng truy vấn bên trong, cùng với các trường mà chúng ta quan tâm.

**Liệt kê 2.10 Tìm kiếm trên nhiềucánh đồng**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "multi\_match": {

"truy vấn": "Java",

**Truy vấn Multi\_match tìm kiếm**

**trên nhiều lĩnh vực**

**Tìm kiếm từ khóa**

"các trường": ["tiêu đề","tóm tắt"]

}

}

}

**Tìm kiếm trên hai trường**

Khi chúng tôi thực hiện truy vấn đa trường này, kết quả của chúng tôi với từ tìm kiếm Java sẽ xuất hiệntrong cả trường tiêu đề và tóm tắt. Nhưng giả sử chúng ta muốn tăng mức độ ưu tiên của một kết quả dựa trên một trường cụ thể. Ví dụ, nếu Java được tìm thấy trong trường tiêu đề, chúng ta muốn tăng mức độ ưu tiên của kết quả tìm kiếm đó lên gấp ba lần trong khi vẫn giữ mức độ ưu tiên bình thường của các tài liệu khác. Để làm như vậy, chúng ta có thể sử dụng tính năng tăng mức độ ưu tiên (tự nhiên) (người dùng có thể vui mừng khi thấy các tài liệu ở đầu danh sách có từ tìm kiếm trong tiêu đề).

**Điểm liên quan**

Kết quả truy vấn toàn văn có giá trị chấm điểm số cho các tài liệu, được thể hiện dưới dạng

\_điểmthuộc tính được gắn vào các kết quả riêng lẻ.\_điểmlà một số dấu phẩy động dương cho biết mức độ liên quan của tài liệu kết quả đến truy vấn. Đầu tiêntài liệu trả về sẽ có điểm cao nhất và tài liệu cuối cùng sẽ có điểm thấp nhất. Đây là*điểm liên quan*và nó cho biết các tài liệu khớp nhau tốt như thế nàotruy vấn. Điểm càng cao thì độ khớp càng lớn.

Elasticsearch có một thuật toán gọi là Okapi Best Match 25 (BM25), đây là thuật toán tăng cường về độ tương đồng giữa tần suất thuật ngữ/tần suất tài liệu nghịch đảo (TF/IDF) giúp tính toán điểm liên quan cho các kết quả và sắp xếp chúng theo thứ tự đó khi trình bày cho khách hàng.

**42 CPHẦN2*Bắt đầu***

Ở phần tiếp theo, chúng ta sẽ xem cách có thể tăng cường kết quả bằng hành động.

* + 1. Tăng cường kết quả

Chúng tôi muốn đưa ra mức độ ưu tiên cao hơn (mức độ liên quan) cho một số trường nhất định khi đưa ra truy vấn đối với nhiều trường. Theo cách này, ngay cả khi người dùng không chỉ định rõ ràng những trường nào cần được tăng cường, chúng tôi vẫn có thể cung cấp kết quả tốt nhất. Elasticsearch cho phép chúng tôi tăng cường mức độ ưu tiên của các trường trong truy vấn của mình bằng cách cung cấp hệ số tăng cường bên cạnh các trường. Đối với kỳ thi-Ví dụ, để tăng trường tiêu đề lên gấp ba lần, chúng ta đặt mức tăng cho trường thành title^3 (tên trường theo sau là ký hiệu dấu mũ và hệ số tăng).

**Liệt kê 2.11 Anhiều\_trận\_hợptruy vấn làm tăng tầm quan trọng của một trường**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "multi\_match": {

"truy vấn": "Java",

**Tìm kiếm bởi vì**

**nhiều trường**

**"các trường": ["title^3","tóm tắt"]**

}

}

}

**Dấu mũ theo sau là số tăng cường cho biết trường đang được tăng cường.**

Kết quả sẽ cho thấy trọng số của điểm trường tiêu đề đã tăng lên. Điều này có nghĩa là chúng tôi đã đưa tài liệu lên vị trí cao hơn bằng cách tăng điểm của nó.

Chúng ta có thể muốn tìm kiếm một cụm từ: ví dụ, "thời tiết ở London sáng nay thế nào" hoặc "công thức làm khoai tây nghiền". Để thực hiện như vậy, chúng ta sử dụng một loại truy vấn khác gọi là match\_phrase—tiếp theo.

* + 1. Tìm kiếm cụm từ

Đôi khi chúng ta muốn tìm kiếm một chuỗi các từ theo đúng thứ tự nhất định, chẳng hạn như tìm tất cả các cuốn sách có cụm từ “sách phải có cho mọi lập trình viên Java” trongtrường tóm tắt. Chúng ta có thể viết truy vấn match\_phrase cho mục đích này.

**Liệt kê 2.12 Tìm kiếm sách với một chính xáccụm từ**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "cụm từ khớp": {

**truy vấn match\_phrase lấy một chính xác**

**sự phù hợp của một cụm từ (chuỗi từ)**

"synopsis": "cuốn sách phải có cho mọi lập trình viên Java"

}

}

}

**Cụm từđể tìm kiếm trong phần tóm tắt của mỗi cuốn sách**

Truy vấn này tìm kiếm chuỗi từ đó trong trường tóm tắt trên tất cả các cuốn sách của chúng tôi và trả về cuốn sách Effective Java:

"lượt truy cập" : [{

"\_score" : 7.300332,

* 1. ***Toàn văntìm kiếm* 43**

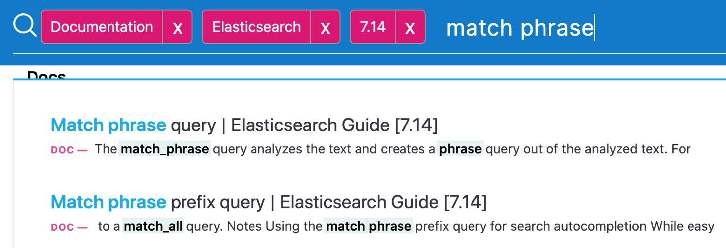
"\_nguồn" : {

"title" : "Java hiệu quả",

"tóm tắt": "Một cuốn sách phải có cho mọi lập trình viên Java vàJava",

}]}

Phần được tô sáng chứng minh rằng truy vấn đã tìm được cuốn sách chúng ta đang tìm kiếm.



**Làm nổi bật kết quả**

Hãy xem cách làm nổi bật một phần văn bản trong tài liệu trả về khớp với truy vấn gốc của chúng ta. Ví dụ, khi chúng ta tìm kiếm một từ hoặc cụm từ trên một trang blog, trang web thường hiển thị văn bản khớp với một số loại tô sáng bằng màu sắc hoặc đổ bóng. Hình minh họa cho thấy tô sáng đang hoạt động: chúng ta đang tìm kiếm "cụm từ khớp" trong trang tài liệu của Elasticsearch.

**Các trận đấu được đánh dấu bằng màu sắc và bóng đổ.**

Chúng ta có thể đạt được hiệu ứng tương tự cho kết quả của mình bằng cách sử dụng một tính năng tiện lợi được gọi là*cao-*

*chiếu sáng*[. Để thực hiện điều này, chúng tôi sửa đổi truy vấn tìm kiếm của mình bằng cách cung cấp](http://mng.bz/pPo2)điểm nổi bậtđối tượng trong thân yêu cầu ở cùng cấp độ vớitruy vấnsự vật:

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "cụm từ khớp": {

"synopsis": "cuốn sách phải có cho mọi lập trình viên Java"

}

},

"điểm nổi bật": {

"các trường": {

"Tóm tắt": {}

}

}

**Đối tượng nổi bật tại**

**cùng cấp độ với đối tượng truy vấn**

**Các lĩnh vực chúng tôi muốn làm nổi bật**

}

Trongđiểm nổi bậtđối tượng, chúng ta có thể thiết lập các trường mà chúng ta muốn áp dụng điểm nổi bật. Ví dụ, ở đây chúng ta yêu cầu công cụ thiết lập điểm nổi bật trênTóm tắtsự vật.

**44 CPHẦN2*Bắt đầu***

***(tiếp theo)***

Kết quả cuối cùng trông giống như thế này, trong đó các kết quả khớp được đánh dấu bằng thẻ đánh dấu HTML (TRONG) chỉ ra những từ được nhấn mạnh:

"lượt truy cập" : [

"\_nguồn" : {

...

"title" : "Java hiệu quả",

"tóm tắt": "Một cuốn sách phải có cho mọi lập trình viên Java

},

"điểm nổi bật" : {

"Tóm tắt" : [

"Một <em>phải</em>-<em>có</em> <em>cuốn sách</em> <em>cho</em>

<em>mọi</em> <em>Java</em> <em>lập trình viên</em> và những người mong muốn sử dụng Java.."]}}

]

Chúng ta có thể dựa vàomatch\_phraseđể tìm kiếm cụm từ chính xác. Nhưng điều gì xảy ra nếu chúng tabỏ qua một hoặc hai từ của cụm từ? Ví dụ, truy vấn có hoạt động không nếu chúng ta yêu cầu nó tìm kiếm "must-have book Java programming" (bỏ các từ for và every)? Truy vấn sẽ không mang lại kết quả nếu bạn chạy lại sau khi xóa hai từ đó. May mắn thay, chúng ta có thể yêu cầu Elasticsearch một cách lịch sự bằng cách đặtbùn lầytham số trênmatch\_phrasetruy vấn. Điều này sẽ được thảo luận ở phần tiếp theo.

* + 1. Các cụm từ bị thiếu từ

Truy vấn match\_phrase mong đợi một cụm từ đầy đủ: một cụm từ không thiếu bất kỳ từ nào. Tuy nhiên, người dùng không phải lúc nào cũng có thể nhập các cụm từ chính xác. Để xử lý vấn đề này, giải pháp của Elasticsearch là đặt tham số slop trên truy vấn match\_phrase: một số nguyên dương cho biết cụm từ bị thiếu bao nhiêu từ khi tìm kiếm. Truy vấn sau có slop là 2, cho biết truy vấn tìm kiếm có thể mong đợi tối đa hai từ bị thiếu hoặc không theo thứ tự khi truy vấn được thực thi.

**Liệt kê 2.13 Ghép một cụm từ với các từ còn thiếu (sử dụngbùn lầy)**

GET bsThực hiện truy vấn này sẽ trả về cuốn sách của chúng tôi, như mong đợi. Bạn có thể thử nghiệm với các giá trị khác củabùn lầyvớisáchtập dữ liệu trong kho lưu trữ.

* + 1. Xử lý lỗi chính tả

Đôi khi người dùng nhập sai chính tả trong tiêu chí tìm kiếm của họ. Các công cụ tìm kiếm rất dễ tha thứ; chúng trả về kết quả bất chấp lỗi chính tả. Các công cụ tìm kiếm hiện đại chấp nhận các vấn đề về chính tả và cung cấp chức năng hỗ trợ chúng một cách ân cần. Elasticsearch hoạt động khá chăm chỉ để tìm điểm tương đồng giữa các từ bằng thuật toán khoảng cách chỉnh sửa Levenshtein.

Elasticsearch xử lý lỗi chính tả bằng cách sử dụng truy vấn khớp với cài đặt fuzziness. Nếu fuzziness được đặt thành 1, một lỗi chính tả (một chữ cái đặt sai chỗ, bị bỏ sót hoặc được thêm vào) có thể được bỏ qua. Ví dụ, nếu người dùng tìm kiếm "Komputer", theo mặc định

* 1. ***Cấp độ học kỳtruy vấn* 45**

truy vấn không nên trả về bất kỳ kết quả nào vì “Komputer” bị viết sai chính tả. Có thể sửa lỗi này bằng cách viết truy vấn khớp sau.

**Liệt kê 2.14 Ghép một cụm từ với lỗi chính tả (sử dụnghơi quăn)**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"thẻ": {

"truy vấn": "Máy tính", "mờ": 1

}

}

}

}

**Truy vấn viết sai chính tả**

**Xử lý một lỗi chính tảlỗi bằng cách đặt độ mờ thành 1**

**Độ mờ và khoảng cách chỉnh sửa của Levenshtein**

Truy vấn mờ tìm kiếm các thuật ngữ tương tự với truy vấn bằng cách sử dụng cái gọi là*Khoảng cách chỉnh sửa của Levenshtein*. Trong ví dụ trước, chúng ta cần thay thếchữ cái đơn*K*với*C*để tìm các tài liệu khớp với truy vấn. Khoảng cách chỉnh sửa là một cơ chế để thay đổi một từ thành một từ khác bằng cách thực hiện các thay đổi ký tự đơn. Hàm fuzziness không chỉ thay đổi một từ; nó chèn và xóa các ký tự để khớp với các từ. Ví dụ, "Komputer" có thể được thay đổi thành "Computer" bằng cách thay thế một chữ cái duy nhất. Hãy xem bài báo này trên Code Project để tìm hiểu thêm về khoảng cách chỉnh sửa của Levenshtein:<http://mng.bz/pPo2>.

Cho đến nay, chúng ta đã xem xét việc tìm kiếm dữ liệu phi cấu trúc (toàn văn bản) bằng cách sử dụng các truy vấn toàn văn bản. Ngoài các truy vấn toàn văn bản, Elasticsearch hỗ trợ các truy vấn để tìm kiếm dữ liệu có cấu trúc: truy vấn cấp thuật ngữ. Các truy vấn cấp thuật ngữ giúp chúng ta tìm kiếm dữ liệu có cấu trúc như số, ngày tháng, địa chỉ IP, v.v. Chúng ta sẽ xem xét các truy vấn này trong một vài phần tiếp theo.

#### Cấp độ học kỳ truy vấn

Elasticsearch có một loại truy vấn riêng, truy vấn cấp thuật ngữ, để hỗ trợ truy vấn dữ liệu có cấu trúc. Số, ngày, phạm vi, địa chỉ IP, v.v. thuộc về danh mục văn bản có cấu trúc. Elasticsearch xử lý dữ liệu có cấu trúc và không có cấu trúc khác nhau: dữ liệu không có cấu trúc (toàn văn) được phân tích, trong khi các trường có cấu trúc được lưu trữ nguyên trạng.

Chúng ta hãy xem lạisáchtài liệu và để mắt đếnấn bản,Đánh giá của amazon,

Vàngày phát hànhcác trường (được tô đậm).

**Liệt kê 2.15 Một mẫusáchtài liệu**

{

"title" : "Java hiệu quả", "author" : "Joshua Bloch",

**46 CPHẦN2*Bắt đầu***

"synopsis": "Một cuốn sách cần phải có cho mọi lập trình viên Java và Java, ...",

**"phiên bản" : 3,**

**"amazon\_rating" : 4.7,**

**"ngày phát hành" : "2017-12-27",**

...

}

Khi chúng ta lập chỉ mục tài liệu lần đầu tiên, vì chúng ta không tạo chỉ mục trước, Elasticsearch sẽ suy ra lược đồ bằng cách phân tích các giá trị của các trường. Ví dụ, trường phiên bản được biểu diễn dưới dạng trường số (không phải văn bản) và do đó làđược công cụ tìm kiếm coi là kiểu dữ liệu dài.

Sử dụng logic tương tự, trường amazon\_rating được xác định là kiểu dữ liệu float vì trường này bao gồm các giá trị thập phân. Cuối cùng, release\_date được xác định là kiểu dữ liệu date vì giá trị được biểu diễn theo định dạng ngày ISO8601 (yyyy- MM-dd). Cả ba trường đều được phân loại là trường không phải văn bản, nghĩa là các giá trị không được mã hóa (không được chia thành các mã thông báo) và không được chuẩn hóa (không có từ đồng nghĩa hoặc không có từ gốc) mà được lưu trữ nguyên trạng.

Truy vấn cấp thuật ngữ tạo ra đầu ra nhị phân: kết quả được lấy nếu truy vấn khớp với tiêu chí; nếu không, không có kết quả nào được gửi. Các truy vấn này không xem xét mức độ khớp của tài liệu (mức độ liên quan); thay vào đó, chúng tập trung vào việc truy vấn có khớp hay không. Vì mức độ liên quan không được xem xét, nên truy vấn cấp thuật ngữ không tạo ra điểm liên quan. Chúng ta hãy xem xét một số truy vấn cấp thuật ngữ trong hai phần tiếp theo.

* + 1. Thuật ngữ truy vấn

Một truy vấn thuật ngữ làđược sử dụng để tìm kiếm kết quả khớp chính xác với giá trị được cung cấp trong tiêu chí tìm kiếm.Ví dụ, để lấy tất cả các sách phiên bản thứ ba, chúng ta có thể viết truy vấn thuật ngữ như danh sách sau đây hiển thị.

**Liệt kê 2.16 Đang tải sách phiên bản thứ ba**

NHẬN sách/\_tìm kiếm

{

"\_source": ["title","edition"], "query": {

"thuật ngữ": {

**Chúng tôi chỉ trả về hai trường trong tài liệu phản hồi.**

**Khai báo truy vấn**

"phiên bản": {

"giá trị": 3

}

}

}

}

**Cung cấp trường và giá trị làm tiêu chí tìm kiếm**

**như một truy vấn cấp độ thuật ngữ**

Truy vấn này trả về tất cả các sách phiên bản thứ ba (chỉ mục của chúng tôi chỉ bao gồm một sách: Java hiệu quả), như được hiển thị ở đây:

"lượt truy cập" : [{

...

**"\_điểm" : 1.0,**

"\_nguồn" : {

* 1. ***Hợp chấttruy vấn* 47**

"title" : "Java hiệu quả", "edition" : 3,

...

}

}]

Nếu bạn xem xét kỹ kết quả, điểm số mặc định là 1.0 vì các truy vấn cấp thuật ngữ không quan tâm đến tính liên quan, như tôi đã đề cập trước đó.

* + 1. Truy vấn phạm vi

Truy vấn phạm vi sẽ lấy ra các kết quả khớp với một phạm vi: ví dụ, lấy các chuyến bay trong khoảng từ 1:00 sáng đến 1:00 chiều hoặc tìm danh sách thanh thiếu niên từ 14 đến 19 tuổi. Truy vấn phạm vi có thể được áp dụng cho ngày tháng, số và các thuộc tính khác, khiến nó trở thành công cụ hữu ích khi tìm kiếm dữ liệu phạm vi.

Tiếp tục ví dụ về cuốn sách của chúng tôi, chúng ta có thể sử dụng truy vấn phạm vi để lấy tất cả các cuốn sách có

amazon\_rating lớn hơn hoặc bằng 4,5 sao và nhỏ hơn hoặc bằng 5 sao.

**Liệt kê 2.17 Aphạm vitruy vấn để lấy sách xếp hạng 4,5 đến 5các ngôi sao**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"phạm vi": {

**Truy vấn phạm vi**

**tuyên ngôn Phạm vi đến**

"amazon\_rating": { "gte": 4.5,

**cuộc thi đấu**

**gte: lớn hơn**

"lte": 5

}

}

}

}

**lte: nhỏ hơn hoặc bằng**

**hoặc bằng**

Truy vấn phạm vi này sẽ lấy ra ba cuốn sách, vì chúng ta có ba cuốn sách có xếp hạng lớn hơn hoặc bằng 4,5 (tôi đã bỏ qua kết quả đầu ra để ngắn gọn).

Có một số ít truy vấn cấp thuật ngữ, bao gồm các thuật ngữ, ID, tồn tại, tiền tố và các thuật ngữ khác. Chúng tôi sẽ xem xét chúng chi tiết hơn trong các chương 8–10.

Cho đến nay, chúng ta đã xem qua các truy vấn có thể hữu ích khi tìm kiếm kết quả dựa trên các tiêu chí cơ bản: khớp với tiêu đề, tìm kiếm một từ trong nhiều trường, tìm người bán được đánh giá cao nhất, v.v. Nhưng trên thực tế, các truy vấn có thể phức tạp như bạn có thể tưởng tượng: ví dụ, tìm kiếm sách phiên bản đầu tiên do Joshua biên soạn với xếp hạng lớn hơn 4,5 và được xuất bản sau năm 2015. May mắn thay, Elasticsearch có các loại truy vấn nâng cao mà chúng ta có thể sử dụng để tìm kiếm tiêu chí phức tạp dưới dạng truy vấn hợp chất. Hãy cùng xem một ví dụ về truy vấn hợp chất trong phần tiếp theo.

#### 2,5 Truy vấn hợp chất

Truy vấn hợp chất trong Elasticsearch cung cấp một cơ chế để tạo các truy vấn tìm kiếm phức tạp. Chúng kết hợp các truy vấn riêng lẻ, được gọi là truy vấn lá (giống như những truy vấn chúng ta đã thấy cho đến nay), để xây dựng các truy vấn mạnh mẽ, bền bỉ phục vụ cho các tình huống phức tạp.

**48 CPHẦN2*Bắt đầu***

(Chúng tôi thảo luận về các truy vấn hợp chất ở cấp độ cao tại đây, nhưng chúng tôi sẽ thảo luận chi tiết hơn trong chương 11.)

Các truy vấn hợp chất như sau:

* Boolean (bool)
* Điểm số hằng số (điểm số hằng số)
* Điểm chức năng (điểm\_hàm)
* Tăng cường (tăng cường)
* Phân ly tối đa (dis\_max)

Trong số đó, truy vấn bool được sử dụng phổ biến nhất, vì vậy chúng ta hãy xem xét truy vấn bool trong thực tế ở đây.

* + 1. Truy vấn Boolean (bool)

Truy vấn Boolean (bool) được sử dụng để tạo logic truy vấn phức tạp bằng cách kết hợp các truy vấn khác dựa trên các điều kiện Boolean. Truy vấn bool mong đợi tìm kiếm được xây dựng bằng cách sử dụng một tập hợp bốn mệnh đề: must, must\_not, should và filter. Sau đâydanh sách hiển thị định dạng của truy vấn bool.

**Liệt kê 2.18 Định dạng của mộtbooltruy vấnvới sự mong đợicác điều khoản**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"bool": { "phải": [{ }],

**Một truy vấn bool là một sự kết hợp**

**của boolean có điều kiệncác điều khoản. Các tiêu chí phải**

**khớp các tài liệu.**

"must\_not": [{ }],

"nên": [{ }],

"lọc": [{ }]

}

} **Truy vấn phải phù hợp**

} **(không có điểm đóng góp).**

**Truy vấn phải khớp.**

**Các tiêu chí không được trùng nhau (không có điểm đóng góp).**

Như bạn có thể thấy, truy vấn bool mong đợi một hoặc nhiều mệnh đề must, must\_not, should và filter để xác định tiêu chí (xem bảng 2.2). Có thể biểu thị nhiều hơn một tiêu chí bằng cách kết hợp các mệnh đề này.

**Bảng 2.2 Boolean (bool) mệnh đề truy vấn**

|  |  |
| --- | --- |
| **Điều khoản** | **Giải thích** |
| phải  phải\_không | Cácphảimệnh đề có nghĩa là tiêu chí tìm kiếm trong truy vấn phải khớp với các tài liệu.sự phù hợp tích cực làm tăng điểm liên quan. Chúng tôixây dựng mộtphảimệnh đề có càng nhiều truy vấn lá càng tốt.  Trong mệnh đề must\_not, các tiêu chí không được khớp với các tài liệu. Mệnh đề này khôngkhông đóng góp vào điểm số (nó được chạy trong bối cảnh thực thi bộ lọc; thông tin thêm về bối cảnh trong chương 11). |

***2.5 Hợp chấttruy vấn* 49**

**Bảng 2.2 Boolean (bool) mệnh đề truy vấn (tiếp theo)**

|  |  |
| --- | --- |
| **Điều khoản** | **Giải thích** |
| nên lọc | Không bắt buộc phải có tiêu chí được xác định trong mệnh đề should để khớp. Tuy nhiên, nếuNếu khớp, điểm liên quan sẽ được tăng lên.  Trong mệnh đề lọc, các tiêu chí phải khớp với các tài liệu, tương tự nhưphảiđiều khoản.Sự khác biệt duy nhất là điểm số không liên quan trong mệnh đề bộ lọc (nó được chạy trong bối cảnh thực thi bộ lọc). |

Giả sử yêu cầu của chúng ta là tìm kiếm những cuốn sách đáp ứng các tiêu chí sau:

* + - * Được biên soạn bởi Joshua
      * Đánh giá lớn hơn 4.7
      * Xuất bản sau năm 2015

Chúng ta cần sử dụng truy vấn bool để kết hợp các tiêu chí này thành một truy vấn với sự trợ giúp của một số mệnh đề từ bảng 2.2. Trong các phần sau, chúng tôi xây dựng truy vấn bool phức hợp cho tiêu chí tìm kiếm của mình. Thay vì giới thiệu cho bạn toàn bộ truy vấn cùng một lúc (có thể rất khó hiểu), chúng tôi chia nhỏ truy vấn thành các tiêu chí tìm kiếm riêng lẻ trước khi ghép tất cả lại với nhau. Để bắt đầu, phần tiếp theo sẽ giới thiệu mệnh đề must để tìm tác giả.

* + 1. Phảiđiều khoản

Chúng tôi muốn tìm tất cả các cuốn sách do Joshua biên soạn, vì vậy chúng tôi có thể tạo truy vấn bool với mệnh đề must. Bên trong mệnh đề must, chúng tôi viết truy vấn khớp để tìm kiếm các cuốn sách do Joshua biên soạn (chúng tôi đã xem xét các truy vấn khớp trong phần 2.2.3). Đây là mã.

**Liệt kê 2.19 Abooltruy vấn với mộtphảiđiều khoản chophù hợp**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{

"cuộc thi đấu": {

**Một bool**

**truy vấn Điều khoản phải: các tài liệu phải phù hợp với tiêu chí.**

**Một trong những truy vấn (truy vấn khớp)**

"tác giả": "Joshua Bloch"

}

}]

}

}

}

**sách phù hợp được viết bởi Joshua**

Lưu ý rằng truy vấn bool được bao gồm trong đối tượng truy vấn. Nó có một mệnh đề must, đến lượt nó, lấy nhiều truy vấn dưới dạng một mảng: ở đây, khớp với tất cả các cuốn sách do Joshua Bloch viết. Truy vấn sẽ trả về hai cuốn sách (Effective Java và Java Concurrency in Practice), cho biết đây là hai cuốn sách duy nhất của Joshua Bloch trong kho tài liệu của chúng tôi.

**50 CPHẦN2*Bắt đầu***

Điều đó có nghĩa là gì khi chúng ta nói rằng mệnh đề must chấp nhận một tập hợp nhiều truy vấn dưới dạng một mảng? Điều đó có nghĩa là chúng ta có thể thêm các truy vấn vào mệnh đề must để làm cho nó phức tạp hơn nhiều. Ví dụ, đoạn mã sau bao gồm một mệnh đề must với hai truy vấn lá: một truy vấn match tìm kiếm tác giả và một truy vấn match\_phrase tìm kiếm một cụm từ.

**Liệt kê 2.20 Aphảimệnh đề có nhiều truy vấn lá**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{

"cuộc thi đấu": {

**phải truy vấn với hai truy vấn lá**

**truy vấn tìm kiếm sách phù hợp**

"tác giả": "Joshua Bloch"

}

},

**được biên soạn bởi Joshua**

{

"cụm từ khớp": {

**Một truy vấn thứ hai tìm kiếm**

**cho một cụm từ trong một lĩnh vực**

"synopsis": "những cuốn sách lập trình Java hay nhất"

}

}]

}

}

}

Chúng ta hãy tiếp tục và thêm một must\_notmệnh đề tạo ra tiêu chí phủ định.

* + 1. Mệnh đề must\_not

Hãy cải thiện tiêu chuẩn của chúng ta. Chúng ta không nên lấy sách của Joshua nếu xếp hạng thấp hơn

4.7. Để đáp ứng điều kiện này, chúng tôi sử dụng mệnh đề must\_not với truy vấn phạm vi đặt xếp hạng nhỏ hơn (lt) 4.7. Danh sách sau đây minh họa mệnh đề must\_not, cùng với mệnh đề must từ truy vấn gốc.

**Liệt kê 2.21 Abooltruy vấn vớiphảiVàphải\_khôngcác điều khoản**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{ "phù hợp": { "tác giả": "Joshua" } }],

**phải nêu rõ điều đó**

**tìm kiếm sách của Joshua**

"must\_not": [{ "range": { "amazon\_rating": { "lt": 4.7}}}]

}

} **mệnh đề must\_not với truy vấn phạm vi**

} **điều đó loại trừ những cuốn sách có xếp hạng thấp hơn**

Truy vấn này chỉ cho ra một cuốn sách, Effective Java. Cuốn sách khác của Joshua Bloch (Java*Đồng thời trong thực tế*) bị loại khỏi danh sách vì nó không phù hợpphải\_khôngtiêu chí (xếp hạng của nó là 4,3, thấp hơn tiêu chuẩn tiên quyết là 4,7).

* 1. ***Hợp chấttruy vấn* 51**

Ngoài việc tìm kiếm những cuốn sách do Joshua viết với xếp hạng không dưới 4,7,chúng ta có thể thêm một điều kiện khác để kiểm tra xem sách có khớp với thẻ không (ví dụ, thẻ = "Phần mềm"). Nếu khớp, chúng ta mong đợi điểm số tăng lên; nếu không, nó không ảnh hưởng đến kết quả. Đối với điều này, chúng ta sử dụng mệnh đề should, được thảo luận trong phần tiếp theo.

* + 1. Mệnh đề nên

Mệnh đề should hoạt động giống như toán tử OR. Nghĩa là, nếu các từ tìm kiếm khớp với truy vấn should, điểm liên quan sẽ tăng lên. Nếu các từ không khớp, truy vấn sẽ không thất bại; mệnh đề sẽ bị bỏ qua. Mệnh đề should liên quan nhiều hơn đến việc tăng điểm liên quan hơn là ảnh hưởng đến kết quả.

Danh sách tiếp theo thêm mệnh đề should vào truy vấn bool. Nó cố gắng khớp với văn bản tìm kiếm nếu tài liệu có thẻ Software.

**Liệt kê 2.22 Sử dụng mộtnêntruy vấn để tăng tính liên quan**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{"match": {"author": "Joshua"}}],

"must\_not":[{"range":{"amazon\_rating":{"lt":4.7}}}],

**"nên": [{"match": {"tags": "Phần mềm"}}]**

}

}

}

**mệnh đề should với truy vấn khớp**

Truy vấn này trả về kết quả (bỏ qua ở đây) với điểm tăng: 2.267993, so với điểm trước đó là 1.9459882 (bạn có thể chạy truy vấn để quan sát điểm). Nếu bạn tò mò, hãy thay đổi truy vấn để bao gồm một từ không khớp (ví dụ: thẻ bằng "Công thức nấu ăn") và thử lại truy vấn. Truy vấn sẽ không thất bại, nhưng điểm sẽ

giống nhau, điều này chứng tỏ rằng truy vấn should chỉ ảnh hưởng đến điểm số.

Tiếp tục với dòng chảy, mệnh đề cuối cùng là bộ lọc, hoạt động chính xác như

mệnh đề must nhưng không ảnh hưởng đến điểm. Chúng ta hãy xem nó hoạt động như thế nào trong phần tiếp theo.

* + 1. Điều khoản lọc

Hãy cải thiện truy vấn của chúng ta thêm một chút, lần này bằng cách lọc ra những cuốn sách được xuất bản trước năm 2015 (tức là chúng ta không muốn bất kỳ cuốn sách nào xuất hiện trong tập kết quả của chúng ta đã được xuất bản)trước năm 2015). Chúng tôi sử dụng mệnh đề bộ lọc cho mục đích này và bất kỳ kết quả nào không khớp với tiêu chí bộ lọc sẽ bị loại bỏ. Truy vấn sau đây thêm mệnh đề bộ lọc với mệnh đề release\_date.

**Liệt kê 2.23 Alọcđiều khoản không ảnh hưởng đến tính liên quan**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

**52 CPHẦN2*Bắt đầu***

"bool": {

"phải": [{"match": {"author": "Joshua"}}],

"must\_not":[{"range":{"amazon\_rating":{"lt":4.7}}}],

"nên": [{"match": {"tags": "Phần mềm"}}],

**"bộ lọc":[{"phạm vi":{"ngày phát hành":{"gte": "2015-01-01"}}}]}**

}

} **điều khoản lọc với**

**một truy vấn phạm vi**

Truy vấn này chỉ trả về một cuốn sách, Effective Java, vì đây là cuốn sách duy nhất trong chỉ mục của chúng tôi khớp với cả ba mệnh đề trong truy vấn bool. Nếu bạn chạy truy vấn này trong Kibana, hãy chú ý đến điểm số không thay đổi trong đầu ra. Mệnh đề bộ lọc không ảnh hưởng đến điểm số: nó chạy trong ngữ cảnh bộ lọc, nghĩa là điểm số không bị thay đổi (thông tin thêm về ngữ cảnh trong chương 8).

Cuối cùng, chúng tôi cũng muốn tìm những cuốn sách phiên bản thứ ba của Joshua. Để làm điều này, chúng tôi cập nhật

lọcđiều khoản có mộtthuật ngữtruy vấn.

**Liệt kê 2.24booltruy vấn với bổ sunglọcđiều khoản**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{"match": {"author": "Joshua"}}],

"must\_not":[{"range":{"amazon\_rating":{"lt":4.7}}}],

"nên": [{"match": {"tags": "Phần mềm"}}], "bộ lọc":[

{"phạm vi":{"ngày phát hành":{"gte": "2015-01-01"}}},

**{"term": {"edition": 3}}**

]}

}

}

**truy vấn thuật ngữ trong mệnh đề lọc**

Truy vấn này là sự kết hợp của các truy vấn toàn văn bản và cấp thuật ngữ, tất cả đều hoạt động đồng bộ để đáp ứng các yêu cầu phức tạp của chúng tôi. Truy vấn bool là công cụ tìm kiếm Swiss Army knife. Chúng tôi dành nhiều thời gian để thảo luận về nhiều truy vấn hợp chất, tùy chọn và mẹo khác nhau trong chương 11, nhằm nâng cao kiến ​​thức của chúng tôi về bộ công cụ này.

Cho đến nay, chúng tôi đã bắt đầu bằng cách triển khai một vài tìm kiếm. Một khía cạnh khác của tìm kiếm là phân tích. Elasticsearch giúp đưa ra số liệu thống kê và tổng hợp bao gồm biểu diễn trực quan của dữ liệu bằng biểu đồ thanh, bản đồ nhiệt, bản đồ, đám mây thẻ và nhiều hơn nữa.

Trong khi tìm kiếm giúp chúng ta tìm kim trong đống cỏ khô, thì tổng hợp thu nhỏ lại để thiết lập bản tóm tắt dữ liệu của chúng ta, như tổng số lỗi máy chủ trong giờ qua, doanh số bán sách trung bình trong quý thứ ba, phim được phân loại theo tổng thu nhập, v.v. Chúng ta sẽ xem xét tổng hợp và các chức năng khác trong chương 13, nhưng ở đây chúng ta sẽ xem xét một số ví dụ cấp cao.

#### Tổng hợp

Cho đến thời điểm này, chúng tôi đã tìm kiếm các tài liệu từ một tập hợp tài liệu nhất định. Mặt khác của đồng xu là phân tích. Phân tích cho phép các tổ chức tìm ra thông tin chi tiết

***2.6 Tổng hợp* 53**

vào dữ liệu bằng cách xem xét bức tranh toàn cảnh và phân tích dữ liệu từ cấp độ rất cao để rút ra kết luận về nó. Chúng tôi sử dụng API tổng hợp để cung cấp phân tích trong Elasticsearch.

Tổng hợp được chia thành ba loại:

* *Tổng hợp số liệu*—Các tập hợp đơn giản nhưtổng hợp,phút,tối đa, Vàtrung bình. Chúng cung cấp giá trị tổng hợp trên một tập hợp dữ liệu tài liệu.
* *Tổng hợp thùng*—Các tập hợp thu thập dữ liệu vào các “thùng”, được phân chia theo các khoảng thời gian như ngày, nhóm tuổi, v.v. Những tập hợp này giúp chúng ta xây dựng biểu đồ hình chữ nhật, biểu đồ hình tròn và các hình ảnh trực quan khác.
* *Tổng hợp đường ống*—Các tập hợp hoạt động dựa trên kết quả từ các tập hợp khác.

Chúng tôi sẽ xem xét một số ví dụ về tổng hợp số liệu và nhóm dữ liệu trong các trang tiếp theo, nhưng chúng tôi sẽ dành nội dung chi tiết (bao gồm cả cách làm việc với tổng hợp đường ống) cho đến chương 13.

Tương tự như tìm kiếm, chúng ta sử dụng điểm cuối \_search cho các phép tổng hợp. Tuy nhiên, chúng ta sử dụng một đối tượng mới có tên là aggs (viết tắt của aggregations) trong yêu cầu thay cho đối tượng truy vấn mà chúng ta đã sử dụng cho đến nay. Để tiết lộ sức mạnh thực sự của Elasticsearch, chúng ta có thể kết hợp cả tìm kiếm và phép tổng hợp trong một truy vấn duy nhất.

Để chứng minh tổng hợp hiệu quả, chúng ta cần chuyển đổi và lập chỉ mục một tập dữ liệu mới: Dữ liệu liên quan đến COVID cho 10 quốc gia. Tệp covid-26march2021.txt để lập chỉ mục dữ liệu bằng API \_bulk có sẵn trong các tệp của cuốn sách; sau đây là một đoạn trích của tệp đó.

**Liệt kê 2.25 Lập chỉ mục hàng loạt dữ liệu liên quan đến COVID**

BÀI VIẾT COVID/\_BULK

{"chỉ mục":{}}

{"quốc gia":"Hoa Kỳ","ngày":"2021-03-26","tử vong":561142,"đã phục hồi":23275268}

{"chỉ mục":{}}

{"quốc gia":"Brazil","ngày":"2021-03-26","tử vong":307326,"đã phục hồi":10824095}

...

API \_bulk lập chỉ mục 10 tài liệu của chúng tôi thành một chỉ mục covid mới được tạo. Vì chúng tôi không quan tâm đến ID tài liệu, chúng tôi để hệ thống tạo ID ngẫu nhiên cho từng tài liệu—do đó, tên chỉ mục và ID trống trong hành động chỉ mục của API:

{"index":{}}). Bây giờ chúng ta đã có một tập hợp tài liệu trong chỉ mục covid, hãy thực hiện một số tác vụ tổng hợp cơ bản, bắt đầu bằng tổng hợp số liệu.

* + 1. Số liệu

Tổng hợp số liệu là tổng hợp đơn giản mà chúng ta thường sử dụng trong cuộc sống hàng ngày: ví dụ, chiều cao trung bình của học sinh trong một lớp là bao nhiêu? Giao dịch phòng ngừa rủi ro tối thiểu là bao nhiêu? Tổng thu nhập của một bộ phim là bao nhiêu? Elasticsearch cung cấp khá nhiều số liệu như vậy, hầu hết đều có thể tự giải thích.

Trước khi xem xét các số liệu thực tế, chúng ta hãy nhanh chóng xem qua cú pháp cho các phép tổng hợp. Các truy vấn tổng hợp được viết bằng cú pháp DSL truy vấn giống như tìm kiếm

**54 CPHẦN2*Bắt đầu***

truy vấn; hình 2.16 cho thấy mộtVí dụ. Điểm đáng chú ý là aggs là đối tượng cấp gốc, với số liệu avg (trung bình) được xác định trên trường amazon\_rating. Sau khi truy vấn này được thực thi, nó sẽ trả về kết quả tổng hợp cho người dùng. Chúng ta hãy cùng bắt đầu thực thi một số tổng hợp số liệu.

**Chúng tôi sử dụng giống nhau\_tìm kiếm**

**điểm cuối vì tổng hợp.**

**aggs(viết tắt củatổng hợp) là tín hiệu cho loại hoạt động mà chúng ta muốn thực hiện.**

NHẬN sách/\_tìm kiếm

{

"aggs": { "xếp hạng trung bình": {

"trung bình": {

**Tên tùy chỉnh do người dùng xác định (đánh giá trung bìnhtrong trường hợp này) được đưa vào trường đầu ra**

**Chúng tôi muốn tính toán giá trị trung bình (trung bình) hệ mét.**

}

}

"trường": "amazon\_rating"

}

}

**Tìm tổng hợp trên**

**Đánh giá của amazoncánh đồng.**

**Hình 2.16 Truy vấn cú pháp DSL để tìm tổng hợp xếp hạng trung bình**

**FXEM TỔNG SỐ BỆNH NHÂN NGUY HIỂM****(TỔNG SỐ LIỆU)**

Quay trở lại dữ liệu COVID của chúng ta, giả sử chúng ta muốn tìm tổng số ca quan trọngbệnh nhân trên khắp 10 quốc gia. Đối với điều đó, chúng tôi sử dụng số liệu tổng sau.

**Liệt kê 2.26 Lấy tổng số bệnh nhân nguy kịch**

NHẬN covid/\_search

{

"aggs": {

**Truy vấn tổng hợp**

**sử dụng aggssự vật Tên do người dùng xác định của**

"bệnh nhân nguy kịch": { "tổng": {

"trường": "quan trọng"

**yêu cầu tổng hợp**

**Số liệu tổng hợp sẽ cộng tổng số bệnh nhân nguy kịch.**

}

} **Trường mà trên đó**

} **tổng hợpđược áp dụng**

}

Với đoạn mã này, chúng ta tạo một truy vấn kiểu tổng hợp; aggs là tín hiệu cho Elastic-search rằng đây là một tác vụ tổng hợp. Số liệu tổng là loại tổng hợp mà chúng ta định thực hiện. Ở đây, chúng ta yêu cầu công cụ tìm tổng số bệnh nhân nguy kịch bằng cách cộng số lượng của từng quốc gia. Phản hồi sẽ như thế này:

"tổng hợp" : { "bệnh nhân nguy kịch" : {

"giá trị" : 44045.0

}

}

***2.6 Tổng hợp* 55**

Tổng số bệnh nhân nguy kịch được trả về dưới tên báo cáo của chúng tôi (critical\_patients) trong phản hồi. Lưu ý rằng phản hồi bao gồm tất cả các tài liệuments trả về, nếu chúng ta không yêu cầu rõ ràng để ngăn chặn chúng. Chúng ta có thể đặt size=0 ở cấp độ gốc để ngăn phản hồi chứa các tài liệu nguồn gốc:

NHẬN covid/\_search

{

**"kích thước": 0,**

"aggs": {

...

}

}

Bây giờ chúng ta đã biết cách tổng hợp hoạt động đối với một số liệu tổng, hãy cùng xem xét một số số liệu khác.

**TRONG****HÁT CÁC SỐ LIỆU KHÁC**

Tương tự như vậy, nếu chúng ta muốn tìm số ca tử vong cao nhất trên tất cả các quốc gia trong dữ liệu COVID của mình, chúng ta có thể sử dụng tổng hợp tối đa.

**Liệt kê 2.27 Sử dụngtối đamét**

NHẬN covid/\_search

{

"kích thước": 0,

"aggs": { "số\_chết\_tối\_đa": {

"tối đa": {

"field": "cái chết"

}

}

}

}

Truy vấn này trả về số ca tử vong cao nhất trong số 10 quốc gia trong tập dữ liệu của chúng tôi:

"tổng hợp" : { "max\_deaths" : {

"giá trị" : 561142.0

}

}

Tương tự như vậy, chúng ta có thể tìm ra số liệu tối thiểu (min), trung bình (avg) và các số liệu khác. Nhưng cómột hàm thống kê trả về tất cả các số liệu cơ bản này cùng nhau: số liệu thống kê.

**Liệt kê 2.28 Trả về tất cả các số liệu thống kê vớithống kêmét**

NHẬN covid/\_search

{

"kích thước": 0,

"aggs": {

"tất cả\_thống\_số\_số\_trị": {

**56 CPHẦN2*Bắt đầu***

"thống kê": {

"field": "cái chết"

}

}

}

}

**Truy vấn thống kê trả về tất cả năm số liệu cốt lõi cùng một lúc.**

Cái nàythống kêtruy vấn trả vềđếm,trung bình,tối đa,phút, Vàtổng hợp, tất cả trong một lần. Đây là phản hồi từ ban đầu của chúng tôithống kêtruy vấn:

"tổng hợp" : { "all\_stats" : {

"đếm" : 10,

"tối thiểu" : 30772.0,

"tối đa" : 561142.0,

"trung bình": 163689,1,

"tổng" : 1636891.0

}

}

**GHI CHÚ**Nếu bạn tò mò, hãy hoán đổi truy vấn stats với truy vấn extended\_stats và kiểm tra kết quả. Bạn sẽ thấy nhiều số liệu thống kê hơn, bao gồm phương sai, độ lệch chuẩn và các số liệu khác. Tôi sẽ để bạn tự thử nghiệm (mã có sẵn trong các tệp của sách).

Bây giờ chúng ta đã biết tổng hợp số liệu là gì, hãy cùng xem qua một loại tổng hợp khác: tổng hợp theo nhóm.

* + 1. Xôtổng hợp

Tổng hợp nhóm (hoặc đơn giản là phân nhóm) phân tách dữ liệu thành nhiều nhóm hoặc nhóm khác nhau. Ví dụ, chúng ta có thể thêm các nhóm như thế này vào nhóm của mình: khảo sát nhóm người lớn theo nhóm tuổi (20–30, 31–40, 41–50), phim theo xếp hạng đánh giá hoặc số lượng nhà mới xây dựng mỗi tháng.

Elasticsearch cung cấp ít nhất hai chục tập hợp ngay khi xuất xưởng, mỗi tập hợp có chiến lược phân nhóm riêng. Hơn nữa, chúng ta có thể lồng các tập hợp vào các nhóm chính. Hãy cùng xem một vài tập hợp phân nhóm đang hoạt động.

**HXÔ ISTOGRAM**

Tổng hợp phân nhóm biểu đồ histogram tạo ra danh sách các nhóm biểu diễn giá trị số bằng cách duyệt qua tất cả các tài liệu. Ví dụ, nếu chúng ta muốn phân loại các quốc gia theo số bệnh nhân nguy kịch trong các nhóm 2.500, chúng ta có thể viết truy vấn sau.

**Liệt kê 2.29 quốc gia theo bệnh nhân nguy kịch trong nhóm2.500**

NHẬN covid/\_search

{

"kích thước": 0,

"aggs": { "critical\_patients\_as\_histogram": {

**Tên báo cáo do người dùng xác định**

***2.6 Tổng hợp* 57**



**Xô**

"histogram": { "field": "critical", "interval": 2500

**Loại tổng hợp nhóm: biểu đồ histogram, trong trường hợp này**

**khoảng thời gian** }

}

}

}

**Trường mà tổng hợp được áp dụng**

Câu trả lời sẽ tương tự như sau, trong đó mỗi nhóm có một khóa và một giá trị:

"tổng hợp" : { "bệnh nhân quan trọng dưới dạng biểu đồ" : {

"thùng" : [{

"key" : 0.0,

"doc\_count" : 4

},

{

"chìa khóa" : 2500.0,

"doc\_count" : 3

},

{

"chìa khóa" : 5000.0,

"doc\_count" : 0

},

{

"chìa khóa" : 7500.0,

"doc\_count" : 3

}]

}

}

Nhóm đầu tiên có bốn tài liệu (quốc gia) với số lượng bệnh nhân nguy kịch lên đến 2.500. Nhóm thứ hai có ba quốc gia với số lượng bệnh nhân nguy kịch từ 2.500 đến 5.000, v.v.

**R****XÔ ANGE**

Tổng hợp nhóm phạm vi xác định một tập hợp các nhóm dựa trên các phạm vi được xác định trước. Ví dụ, giả sử chúng ta muốn phân loại số thương vong do COVID theo quốc gia (thương vong lên đến 60.000, 60.000–70.000, 70.000–80.000 và 80.000–120.000). Chúng ta có thể

xác định các phạm vi đó như được hiển thị ở đây.

**Liệt kê 2.30 Thương vong trong phạm vi tùy chỉnh bằng cách sử dụngphạm vixô**

NHẬN covid/\_search

{

"kích thước": 0,

"aggs": { "range\_countries":



{

"phạm vi": {

"field": "cái chết", "phạm vi": [

{"đến": 60000},

**phạm vi xô**

**tổng hợp Trường mà trên đó để**

**áp dụng tổng hợp**

**Xác định các phạm vi tùy chỉnh dưới dạng một mảng**

{"từ": 60000,"đến": 70000},

**58 CPHẦN2*Bắt đầu***

{"từ": 70000,"đến": 80000},

{"từ": 80000,"đến": 120000}

]

}

}

}

}

Chúng tôi định nghĩa một loại bucket tổng hợp phạm vi với một tập hợp các phạm vi tùy chỉnh. Sau khi truy vấn được thực thi, các bucket kết quả sẽ hiển thị các khóa với các phạm vi bucket tùy chỉnh và số lượng tài liệu cho mỗi phạm vi:

"tổng hợp" : { "range\_countries" : {

"thùng" : [{

"key" : "\*-60000.0", "to" : 60000.0,

**"doc\_count" : 1**

},{

"khóa": "60000.0-70000.0",

"từ" : 60000.0,

"đến" : 70000.0,

**"doc\_count" : 0**

},{

"khóa": "70000.0-80000.0",

"từ" : 70000.0,

"đến" : 80000.0,

**"doc\_count" : 2**

},{

"khóa": "80000.0-120000.0",

"từ" : 80000.0,

"đến" : 120000.0,

**"doc\_count" : 3**

}]

}

}

Kết quả cho thấy một quốc gia có tới 60.000 thương vong, ba quốc gia có số thương vong từ 80.000 đến 12.0000, v.v.

Chúng ta có thể thực hiện một tập hợp phong phú các phép tổng hợp trên dữ liệu của mình bằng cách sử dụng các hàm thống kê có sẵn. Chúng ta sẽ xem xét các hàm này trong chương 13; cho đến lúc đó, hãy kiên nhẫn.

Chúng ta mới chỉ khám phá được bề nổi của những gì Elasticsearch cung cấp; chúng ta sẽ khám phá nhiều chức năng hơn nữa trong các chương tiếp theo. Bây giờ, đã đến lúc kết thúc. Trong chương tiếp theo, chúng ta tìm hiểu kiến ​​trúc của Elasticsearch, cơ chế tìm kiếm, các bộ phận chuyển động của nó và nhiều hơn nữa.

#### Bản tóm tắt

* Elasticsearch cung cấp một bộ API tài liệu để lập chỉ mục dữ liệu và bảng điều khiển Dev Tools của Kibana hỗ trợ viết các truy vấn lập chỉ mục để duy trì dữ liệu.
* Để lấy một tài liệu bằng cách sử dụng một API tài liệu duy nhất, chúng tôi phát hành mộtLẤYlệnh trên chỉ mục có ID (NHẬN <index\_name>/\_doc/<ID>).

***Bản tóm tắt* 59**

* + Để lấy nhiều tài liệu, nếu các mã định danh tài liệu có sẵn, chúng ta có thể sử dụngidtruy vấn.
  + Elasticsearch cung cấp nhiều API tìm kiếm, bao gồm các truy vấn cơ bản và nâng cao.
  + Truy vấn toàn văn tìm kiếm trong dữ liệu phi cấu trúc để tìm tài liệu có liên quan.
  + Truy vấn thuật ngữ tìm kiếm thông qua dữ liệu có cấu trúc như số và ngày tháng để tìm tài liệu phù hợp.
  + Truy vấn hợp chất cho phép chúng ta biên dịch các truy vấn lá và tạo ra một tập hợp các truy vấn nâng cao hơn.booltruy vấn, một trong những truy vấn hợp chất, cung cấp một cơ chế để tạo truy vấn nâng cao với nhiều mệnh đề (ví dụ:phải,phải\_không,nên, Vàlọc). Các mệnh đề này giúp chúng ta tạo ra các truy vấn phức tạp.
  + Trong khi tìm kiếm tìm kiếm các tài liệu phù hợp dựa trên các tiêu chí nhất định, phân tích cho phép chúng ta tổng hợp dữ liệu bằng cách cung cấp các hàm thống kê.
  + Tổng hợp số liệu lấy các tổng hợp chung nhưtối đa,phút,tổng hợp, Vàtrung bình
  + Tổng hợp nhóm phân loại tài liệu thành nhiều nhóm khác nhau (nhóm) dựa trên các tiêu chí nhất định.

*Ngành kiến ​​​​trúc*

***Chương này bao gồm***

* Kiến trúc cấp cao và các khối xây dựng của Elasticsearch
* Cơ chế tìm kiếm và lập chỉ mục
* Hiểu cách hoạt động của chỉ số đảo ngược
* Thuật toán liên quan và tương đồng
* Thuật toán định tuyến

Trong chương 2, chúng ta đã chơi với các tính năng cơ bản của Elasticsearch: chúng ta lập chỉ mục tài liệu, thực hiện truy vấn tìm kiếm, hướng dẫn qua các hàm phân tích, v.v. Chúng ta đã chơi với máy chủ một thời gian ngắn mà không biết nhiều về nội bộ của nó. Tin tốt là chúng ta không cần phải đổ mồ hôi để bắt đầu với Elasticsearch. Nó dễ sử dụng ở mức độ đơn giản nhưng cần thời gian để thành thạo.

Tất nhiên, Elasticsearch, giống như bất kỳ công cụ tìm kiếm nào khác, đòi hỏi phải đào sâu để trở thành bậc thầy về công nghệ. Điều đó nói rằng, sản phẩm được thiết kế để hoạt động ngay lập tức với các API và công cụ trực quan, và chúng ta có thể sử dụng phần mềm mà không cần nhiều kiến ​​thức tiên quyết. Trước khi chúng ta bị cuốn theo những thứ dễ sử dụng, thực hành

**60**

***3.1 Một cấp độ caoTổng quan* 61**

các khía cạnh của Elasticsearch, về lâu dài, việc hiểu được kiến ​​trúc cấp cao, hoạt động bên trong của máy chủ và sự phân đôi của các bộ phận chuyển động của nó sẽ có lợi cho chúng ta.

Việc nắm bắt máy chủ được khuyến khích mạnh mẽ đối với các kỹ sư muốn sử dụng Elasticsearch hiệu quả và hiệu suất cao. Chúng ta có thể cần gỡ lỗi lý do tại sao kết quả truy vấn không như mong đợi. Chúng ta có thể được giao nhiệm vụ tìm ra lý do khiến hiệu suất giảm sút phát sinh do sự gia tăng theo cấp số nhân của dữ liệu được lập chỉ mục. Cụm có thể trở nên không ổn định do sự cố bộ nhớ, dẫn đến cuộc gọi sản xuất lúc 2:00 sáng. Hoặc các yêu cầu kinh doanh có thể yêu cầu một trình phân tích ngôn ngữ tùy chỉnh để tích hợp ứng dụng với ngôn ngữ của quốc gia khác.

Là kỹ sư, chúng ta được kỳ vọng có thể tinh chỉnh các truy vấn hoặc điều chỉnh các tính năng quản trị hoặc tạo ra một trang trại đa cụm Elasticsearch cho nhu cầu kinh doanh. Và để có được kiến ​​thức như vậy, chúng ta phải nắm vững công nghệ bằng cách hiểu cách thức hoạt động bên trong của nó—các bu lông và ổ trục—đó chính là nội dung của chương này.

Trong chương này, chúng ta sẽ nói về các khối xây dựng của Elasticsearch và hiểu rõ hơn về cách thức hoạt động của quy trình tìm kiếm và lập chỉ mục đằng sau hậu trường. Chúng ta sẽ tìm hiểu về các nguyên tắc cơ bản làm nền tảng cho công cụ tìm kiếm, chẳng hạn như chỉ mục đảo ngược, tính liên quan và phân tích văn bản. Cuối cùng, chúng ta sẽ khám phá bản chất phân cụm và phân tán của máy chủ Elasticsearch. Trước tiên, hãy cùng xem cách hoạt động của công cụ Elasticsearch ở cấp độ cao.

#### 3.1 Tổng quan cấp cao

Elasticsearch là một ứng dụng phía máy chủ có khả năng chạy trên mọi thứ từ PC cá nhân đến một trang trại máy tính phục vụ hàng gigabyte, terabyte hoặc petabyte dữ liệu. Nó được phát triển bằng ngôn ngữ lập trình Java với Apache Lucene bên trong. Apache Lucene, một thư viện tìm kiếm toàn văn hiệu suất cao được phát triển bằng Java,

được biết đến rộng rãi với các tính năng tìm kiếm và lập chỉ mục mạnh mẽ. Tuy nhiên, Lucene không phải là một ứng dụng hoàn chỉnh mà chúng ta có thể dễ dàng tải xuống, cài đặt và làm việc. Vì đây là một thư viện, chúng ta được kỳ vọng sẽ tích hợp các ứng dụng với nó thông qua giao diện lập trình. Elasticsearch thực hiện chính xác điều đó: nó gói Lucene như một thư viện tìm kiếm toàn văn cốt lõi của nó, xây dựng một ứng dụng phía máy chủ phân tán và có thể mở rộng. Elasticsearch đã xây dựng một ứng dụng không phụ thuộc vào ngôn ngữ lập trình với Lucene là trung tâm để phục vụ tìm kiếm toàn văn.

Tuy nhiên, Elasticsearch không chỉ là một công cụ tìm kiếm toàn văn. Nó đã phát triển thành một công cụ tìm kiếm phổ biến với các tổng hợp và phân tích phục vụ cho nhiều trường hợp sử dụng khác nhau (giám sát ứng dụng, phân tích dữ liệu nhật ký, chức năng tìm kiếm ứng dụng web, ghi lại sự kiện bảo mật, học máy, v.v.). Đây là một công cụ tìm kiếm hiện đại, có khả năng mở rộng, hiệu suất cao với mục tiêu chính là tính khả dụng cao, khả năng chịu lỗi và tốc độ.

Giống như một tài khoản ngân hàng vô dụng nếu không có tiền trong đó, Elasticsearch chẳng là gì nếu không có dữ liệu: dữ liệu đưa vào Elasticsearch và dữ liệu đưa ra. Hãy dành thời gian xem xét cách Elasticsearch xử lý dữ liệu này.

**62 CPHẦN3*Ngành kiến ​​​​trúc***

* + 1. Dữ liệu trong

Elasticsearch cần dữ liệu trước khi có thể cung cấp câu trả lời cho các truy vấn. Dữ liệu có thể được lập chỉ mục vào Elasticsearch từ nhiều nguồn và theo nhiều cách khác nhau: trích xuất từ ​​cơ sở dữ liệu, sao chép từ hệ thống tệp, tải từ các hệ thống khác (bao gồm cả hệ thống phát trực tuyến thời gian thực), v.v.

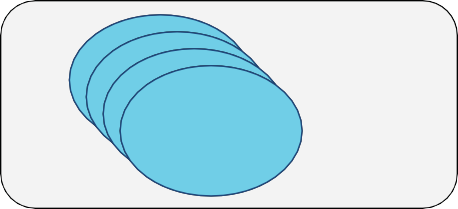
Hình 3.1 cho thấy dữ liệu được đưa vào Elasticsearch thông qua ba nguồn dữ liệu:

* + - * *Cơ sở dữ liệu*—Các ứng dụng thường lưu trữ dữ liệu trong cơ sở dữ liệu như một hệ thống bản ghi có thẩm quyền. Chúng ta có thể chuẩn bị Elasticsearch bằng dữ liệu được lấy từ cơ sở dữ liệu theo lô hoặc gần thời gian thực. Vì hình dạng dữ liệu trong cơ sở dữ liệu có thể không chính xác như Elasticsearch mong đợi, chúng ta có thể sử dụng công cụ trích xuất, chuyển đổi, tải (ETL) (như Logstash—một công cụ xử lý dữ liệu từ bộ Elastic) để chuyển đổi và làm giàu dữ liệu trước khi lập chỉ mục vào Elasticsearch.
      * *Lưu trữ tập tin*—Các ứng dụng, máy ảo (VM), cơ sở dữ liệu và các hệ thống khác tạo ra hàng tấn dữ liệu nhật ký và số liệu. Dữ liệu này rất quan trọng để phân tích lỗi, gỡ lỗi ứng dụng, theo dõi yêu cầu của người dùng và kiểm tra. Dữ liệu có thể được lưu trữ trong bộ lưu trữ phần cứng vật lý hoặc các vị trí đám mây như AWS S3 hoặc Azure files/blob. Dữ liệu này phải được đưa vào Elasticsearch để tìm kiếm, phân tích và lưu trữ. Chúng ta có thể sử dụng các công cụ như Filebeat (một công cụ chuyển dữ liệu với mục đích duy nhất là đưa dữ liệu từ các tệp vào các đích như Elasticsearch) hoặc Logstash để đổ dữ liệu vào Elasticsearch cho mục đích tìm kiếm, gỡ lỗi và phân tích.



Yêu cầu truy vấn ??

Phản hồi truy vấn



Tài liệu

**Nút A** (JSON)

**Cụm**

Kibana

**Khách hàng đưa ra truy vấn sẽ nhận được kết quả tìm kiếm từ máy chủ.**

Phát trực tiếp (thời gian thực)

Dịch vụ lập chỉ mục

Dữ liệu

Tài liệu



**Dữ liệu được công cụ phân tích và lưu trữ trong quá trình lập chỉ mục.**

**Dịch vụ lập chỉ mục tùy chỉnh đảm bảo dữ liệu được đưa vào Elasticsearch từ nhiều nguồn khác nhau.**

**Dữ liệu chảy vào Elasticsearch từ nhiều nguồn.**

Cơ sở dữ liệu

Lưu trữ tập tin

Ứng dụng

**Hình 3.1 Chuẩn bị Elasticsearchvới dữ liệu**

***3.1 Một cấp độ caoTổng quan* 63**

* + - * + *Ứng dụng*—Một số ứng dụng như eBay, Twitter, các trang web vận chuyển (tàu hỏa và máy bay) và các công cụ định giá/báo giá phát ra các sự kiện dưới dạng luồng dữ liệu, thường là theo thời gian thực. Chúng ta có thể sử dụng các thành phần Elastic Stack như Logstash hoặc xây dựng các thư viện phát trực tuyến theo thời gian thực nội bộ để xuất bản dữ liệu lên Elasticsearch; sau đó, dữ liệu này được sử dụng cho chức năng tìm kiếm.

Elasticsearch mong đợi dữ liệu đến từ bộ lưu trữ liên tục hoặc theo thời gian thực được phân tích và xử lý nội bộ. Quy trình phân tích dưới dạng tìm kiếm và phân tích giúp truy xuất dữ liệu hiệu quả. Trong thiết lập thông thường, các tổ chức sử dụng các công cụ ETL để vận chuyển dữ liệu vào Elasticsearch. Dữ liệu được thu thập có thể tìm kiếm được sau khi đi qua kho lưu trữ Elasticsearch.

* + 1. Đang xử lý dữ liệu

Đơn vị thông tin cơ bản trong Elasticsearch được biểu diễn bằng một tài liệu JSON. Ví dụ, chúng ta có thể tạo một bài báo dưới dạng tài liệu JSON cho một tạp chí, như danh sách sau đây cho thấy.

**Liệt kê 3.1 Bài viết tin tức tiêu biểu được biểu diễn dưới dạng tài liệu JSON**

{

"title":"Làm việc từ xa có phải là chuẩn mực mới không?", "author":"John Doe",

"synopsis":"Covid đã thay đổi cuộc sống. Nó đã thay đổi cách chúng ta làm việc..", "publish\_date":"2021-01-01",

"số\_từ":3500

}

**CTHU THẬP DỮ LIỆU**

Cũng giống như chúng ta lưu trữ dữ liệu trong cơ sở dữ liệu quan hệ, chúng ta cần phải lưu trữ (lưu trữ) dữ liệu trong Elasticsearch. Elasticsearch lưu trữ tất cả dữ liệu của mình trên đĩa bằng các kỹ thuật được tối ưu hóa và nén. Ngoài dữ liệu kinh doanh của chúng tôi, Elasticsearch còn có dữ liệu riêng liên quan đến trạng thái cụm, thông tin xuyên quốc gia, v.v. cần được lưu trữ.

Nếu bạn cài đặt Elasticsearch bằng tệp nhị phân (phiên bản nén), nó có một thư mục có tên là data trong thư mục cài đặt. Các thư mục data có thể được gắn kết dưới dạng hệ thống tệp mạng, được định nghĩa là đường dẫn gắn kết hoặc được khai báo bằng biến path.data. Bất kể cơ chế cài đặt nào, Elasticsearch đều lưu trữ dữ liệu trên hệ thống tệp được tối ưu hóa.

Đi sâu hơn một chút vào các thư mục dữ liệu, Elasticsearch phân loại dữ liệu theo từng nút và từng loại dữ liệu. Mỗi nút có một thư mục chuyên dụng với một tập hợp các thùng chứa dữ liệu liên quan. Elasticsearch tạo ra một tập hợp các thùng (gọi là chỉ mục trong thuật ngữ Elasticsearch) dựa trên từng loại dữ liệu. Nói một cách đơn giản, chỉ mục là một tập hợp các tài liệu hợp lý. Đó là một thùng chứa dữ liệu kinh doanh của chúng tôi.

Ví dụ, các bài báo có thể được lưu trữ trong một nhóm có tên là news và trades trong một chỉ mục có tên là trades, và phim có thể được chia thành các chỉ mục theo thể loại như classic\_movies, comedy\_movies, horror\_movies, v.v. Tất cả các tài liệu thuộc một loại cụ thể (ví dụ, cars, trades hoặc students) đều nằm trong chỉ mục riêng của chúng. Chúng ta có thể tạo bao nhiêu chỉ mục tùy ý, dựa trên các yêu cầu của mình (lưu ý đến giới hạn bộ nhớ/kích thước).

**64 CPHẦN3*Ngành kiến ​​​​trúc***

**Một chỉ mục trong Elasticsearch giống như một bảng trong cơ sở dữ liệu**

Trong cơ sở dữ liệu quan hệ, chúng ta định nghĩa một bảng để lưu trữ hồ sơ của mình. Mở rộng khái niệm này, một chỉ mục trong Elasticsearch tương đương với một bảng trong cơ sở dữ liệu; Elasticsearch cho phép chúng ta tạo một chỉ mục không có lược đồ (có nghĩa là chúng ta không cần tạo lược đồ dựa trên mô hình dữ liệu của mình trước, không giống như trong cơ sở dữ liệu, nơi một bảng không thể tồn tại nếu không có lược đồ), giúp lập chỉ mục văn bản dạng tự do.

MỘT*chỉ số*là một bộ sưu tập hợp lý các tài liệu được lưu trữ theo từng mảnh.*Mảnh vỡ*đang chạy các phiên bản Apache Lucene. (Hiểu được cơ chế này là một điều khó khăn và có vẻ quá sức—hãy kiên nhẫn, vì chúng tôi sẽ sớm đề cập đến các phân đoạn và các bộ phận chuyển động khác.)

**T****LOẠI DỮ LIỆU**

Dữ liệu có nhiều loại khác nhau: ngày tháng, số, chuỗi, Boolean, địa chỉ IP, vị trí, v.v. Elasticsearch cho phép chúng ta lập chỉ mục tài liệu bằng cách hỗ trợ các loại dữ liệu phong phú. Một quy trình được gọi là ánh xạ chuyển đổi các loại dữ liệu JSON thành các loại dữ liệu Elasticsearch phù hợp. Ánh xạ sử dụng định nghĩa lược đồ để cho Elasticsearch biết cách xử lý dữ liệu của chúng tatrường. Ví dụ, trường tiêu đề và tóm tắt trong tài liệu bài viết tin tức trong danh sách 3.1 là trường văn bản, trong khi published\_date là trường ngày và number\_of\_words là trường số.

Trong khi lập chỉ mục dữ liệu, Elasticsearch phân tích dữ liệu đến theo từng trường. Nó phân tích từng trường bằng các thuật toán nâng cao dựa trên các định nghĩa ánh xạ. Sau đó, nó lưu trữ các trường này trong các cấu trúc dữ liệu hiệu quả sao cho dữ liệu ở dạng có thể tìm kiếm và phân tích cho mục đích của chúng ta. Các trường toàn văn trải qua một quy trình bổ sung gọi là phân tích văn bản, đây chính là trái tim và linh hồn của một công cụ tìm kiếm hiện đại như Elasticsearch. Phân tích văn bản (được thảo luận trong phần tiếp theo) là một bước quan trọng để đưa dữ liệu thô của chúng ta vào dạng cho phép Elasticsearch truy xuất dữ liệu hiệu quả trong khi hỗ trợ vô số truy vấn.

**MỘTPHÂN TÍCH DỮ LIỆU**

Dữ liệu được biểu diễn dưới dạng văn bản được phân tích trong giai đoạn phân tích văn bản. Văn bản được chia nhỏTôiKhông có gì từ(gọi điện *các mã thông báo*) sử dụng Một bộ của quy tắc. Về cơ bản, hai các quá trình xảy ra trong quá trình phân tích: mã hóa và chuẩn hóa.

*Mã hóa*là quá trình chia văn bản thành các mã thông báo dựa trên một tập hợp các quy tắc. Như thể hiện trong bảng 3.1, văn bản trong trường tóm tắt được chia thành các từ riêng lẻ(mã thông báo) được phân tách bằng dấu cách (điều này được thực hiện bởi một trình phân tích mã thông báo tiêu chuẩn—một thành phần phần mềm tích hợp chia văn bản thành các mã thông báo dựa trên các quy tắc được xác định trước).

**Bảng 3.1 Không được mã hóa so với được mã hóaTóm tắtcánh đồng**

|  |  |
| --- | --- |
| Chuỗi không được mã hóa | Peter Piper đã hái một giỏ ớt ngâm chua! |
| (trước khi mã hóa) |  |
| Được mã hóaSợi dây | [peter,piper,hái,một,món,ớt,ngâm,ngâm] |
| (sau khi mã hóa) | Lưu ý: các mã thông báo được viết thường và dấu gạch nối và dấu chấm than |
|  | đã bị xóa bỏ. |

* 1. ***Một cấp độ caoTổng quan* 65**

Ví dụ, nếu chúng ta tìm kiếm Peter and Peppers, chúng ta mong đợi tài liệu khớp với tiêu chí tìm kiếm của chúng ta. Sẽ khó khớp với tiêu chí tìm kiếm nếu văn bản được đưa vào nguyên trạng mà không được chuyển đổi thành mã thông báo.

Trong khi chúng tôi đã phân tách các từ dựa trên khoảng trắng trong ví dụ ở bảng 3.1, một số biến thể tokenizer có sẵn trong Elasticsearch có thể giúp chúng tôi trích xuất các mã thông báo dựa trên các chữ số, chữ cái không phải chữ cái, từ dừng, v.v.; chúng tôi không giới hạn ở khoảng trắng. Ví dụ, giả sử chúng tôi muốn tìm kiếm dữ liệu của mình theo những cách khác. Tiếp tục ví dụ tương tự, người dùng có thể sử dụng bất kỳ văn bản tìm kiếm nào sau đây:

* “Peter Piper là ai?”
* “Peter đã chọn gì?”
* “Peter đã hái ớt ngâm chưa?”
* “Ớt muối Peter”
* “Dưa chua ớt” (tìm kiếm của chúng tôi sẽ trả về tài liệu này theo truy vấn này)

Trong quá trình phân tách, Elasticsearch lưu trữ các mã thông báo (các từ riêng lẻ) nhưng không cải thiện, làm giàu hoặc chuyển đổi chúng. Nghĩa là, các mã thông báo được lưu trữ nguyên trạng. Nếu chúng ta tìm kiếm "ớt ngâm", có thể chúng ta sẽ nhận được tài liệu Peter Piper này, mặc dù điểm liên quan (cho chúng ta biết kết quả tìm kiếm khớp với tiêu chí truy vấn chặt chẽ như thế nào) có thể hơi thấp. Tuy nhiên, tìm kiếm "ớt ớt" có thể không trả về kết quả nào (tên gọi khác của ớt là capsicum). Trừ khi chúng ta có các mã thông báo đã làm giàu, nếu không sẽ không có cách dễ dàng nào để trả lời tìm kiếm cho văn bản như vậy—và đó là lý do tại sao chúng ta có một quy trình khác được gọi là chuẩn hóa hoạt động trên các mã thông báo này.

*Chuẩn hóa*giúp xây dựng trải nghiệm người dùng phong phú bằng cách tạo dữ liệu bổ sung xung quanh các mã thông báo. Đây là quá trình giảm (gốc) các mã thông báo thành các từ gốc hoặc tạo từ đồng nghĩa cho chúng. Ví dụ:

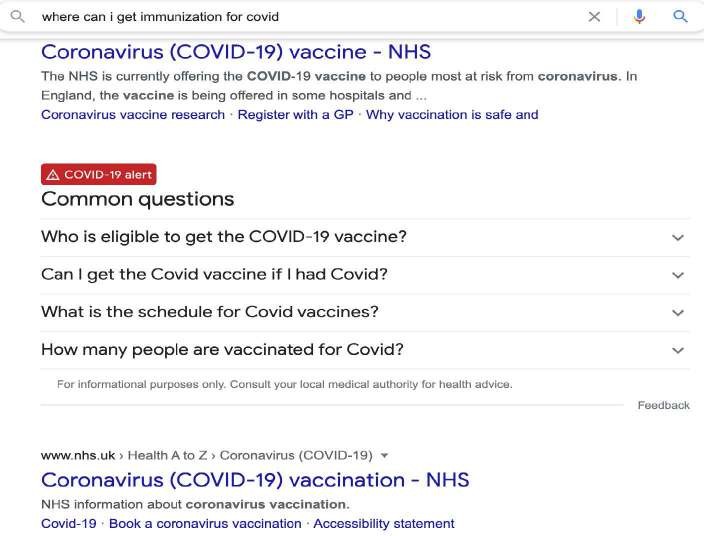
* Cácớt chuôngcó thể dùng từ gốc để tạo ra các từ thay thế như capsicum.
* CácNgười thổi sáotoken có thể được tạo ra để sản xuất Bagpiper.

Chuẩn hóa cũng có thể giúp xây dựng danh sách các từ đồng nghĩa cho các mã thông báo, một lần nữa làm phong phú thêm trải nghiệm tìm kiếm của người dùng. Ví dụ, danh sách các từ đồng nghĩa cho công việc có thể bao gồm working, office work, employment hoặc job; và authoring, authored và authored by đều liên quan đến author. Cùng với các mã thông báo, các từ gốc, từ đồng nghĩa, v.v. được lưu trữ trong một cấu trúc dữ liệu nâng cao được gọi là chỉ mục đảo ngược. Hãy lấy ví dụ về một từ duy nhất, vaccine. Từ này có thể được phân tích và lưu trữ trong chỉ mục đảo ngược so với một số từ gốc hoặc syn-các từ như tiêm chủng, đã tiêm vắc-xin, miễn dịch, miễn dịch, tiêm chủng, tiêm nhắc lại, v.v.

ra.

Ví dụ, khi người dùng tìm kiếm về tiêm chủng, các tài liệu có từ khóa vắc-xin hoặc từ khóa liên quan có thể xuất hiện vì những từ khóa này có liên quan với nhau. Nếu bạn tìm kiếm “tôi có thể tiêm chủng phòng ngừa covid ở đâu” trên Google, khả năng cao là kết quả của bạn sẽ liên quan đến vắc-xin. Khi tôi thử tìm kiếm này, Google trả về kết quả mong đợi (xem hình 3.2).

**66 CPHẦN3*Ngành kiến ​​​​trúc***



**Hình 3.2 Kết quả tìm kiếm (như mong đợi)từ Google**

Chúng ta tìm hiểu về giai đoạn phân tích này trong chương 7. Sau khi dữ liệu đã được phân tích, nó được gửi đến một nút dữ liệu cụ thể để duy trì. Thông thường, phân tích văn bản, hoạt động duy trì, sao chép dữ liệu, v.v. được thực hiện trong một phần giây, do đó dữ liệu đã sẵn sàng để sử dụng trong vòng chưa đầy một giây sau khi được lập chỉ mục (thỏa thuận mức dịch vụ [SLA] mà Elasticsearch bảo đảm). Bước tiếp theo sau khi xử lý dữ liệu trong Elasticsearch là truy xuất dữ liệu dưới dạng tìm kiếm và phân tích, chúng ta sẽ thảo luận về điều này trong phần tiếp theo.

* + 1. Dữ liệu ra

Sau khi dữ liệu được thu thập, phân tích và lưu trữ trong Elasticsearch, dữ liệu sẽ được truy xuất thông qua các truy vấn tìm kiếm và phân tích. Search sẽ tìm dữ liệu khớp với một truy vấn cụ thể, trong khi ana-lytics sẽ thu thập dữ liệu để tạo thành số liệu thống kê tóm tắt. Search không chỉ tìm kiếm các từ khớp chính xác mà còn tìm kiếm các từ gốc, từ đồng nghĩa, lỗi chính tả, v.v. Các tính năng nâng cao của truy vấn toàn văn là những gì khiến các công cụ tìm kiếm hiện đại như Elasticsearch trở thành lựa chọn ưu tiên cho tìm kiếm.

Khi một truy vấn tìm kiếm được đưa ra, nếu trường là trường toàn văn, nó sẽ trải qua một giai đoạn phân tích tương tự như giai đoạn được thực hiện khi trường được lập chỉ mục. Nghĩa là, truy vấn được mã hóa và chuẩn hóa bằng cách sử dụng cùng các trình phân tích liên kết với trường đó. Các mã thông báo tương ứng được tìm kiếm và khớp trong chỉ mục đảo ngược và kết quả dựa trên các kết quả khớp được chuyển tiếp trở lại máy khách. Ví dụ, nếu một trường được đặt bằng tiếng Pháp

* 1. ***Tòa nhàkhối* 67**

bộ phân tích, cùng một bộ phân tích được sử dụng trong giai đoạn tìm kiếm. Điều này đảm bảo rằng các từ được lập chỉ mục và chèn vào các chỉ mục đảo ngược cũng được khớp trong khi tìm kiếm.

Như trong bất kỳ ứng dụng nào, một loạt các khối xây dựng tạo nên máy chủ Elasticsearch. Chúng tôi thảo luận về chúng trong phần tiếp theo và chúng tôi làm việc rộng rãi với chúng trong suốt cuốn sách này.

#### Các khối xây dựng

Trong chương trước, chúng tôi đã lập chỉ mục các tài liệu mẫu và thực hiện tìm kiếm trên đó. Tuy nhiên, chúng tôi đã đẩy cuộc thảo luận về các thành phần như chỉ mục, tài liệu, phân đoạn và bản sao sang thời điểm sau—và thời điểm đó đã đến. Các thành phần này tạo nên Elasticsearch và là các khối xây dựng của công cụ tìm kiếm. Hãy cùng xem xét chúng chi tiết và tìm hiểu ý nghĩa của chúng.

* + 1. Tài liệu

Tài liệu là đơn vị thông tin cơ bản được Elasticsearch lập chỉ mục để lưu trữ. Các trường riêng lẻ của mỗi tài liệu được phân tích để tìm kiếm và phân tích nhanh hơn. Elasticsearch mong đợi các tài liệu ở định dạng JSON. JSON là một định dạng dữ liệu đơn giản mà con người có thể đọc được đã trở nên phổ biến trong những năm gần đây. Nó biểu diễn dữ liệu dưới dạng cặp khóa-giá trị: ví dụ {"name":"John Doe"}. Khi chúng ta giao tiếp với Elasticsearch qua các API RESTful, chúng ta gửi các truy vấn đến Elasticsearch dưới dạng các đối tượng JSON. Đến lượt mình, Elasticsearch sẽ tuần tự hóa các tài liệu JSON này và lưu trữ chúng trong kho tài liệu phân tán của nó sau khi chúng được phân tích.

**P****DỮ LIỆU TÀI LIỆU ARSING**

Dữ liệu của chúng tôi, được biểu diễn dưới dạng tài liệu JSON, được Elasticsearch phân tích trong quá trình lập chỉ mục. Ví dụ, hình 3.3 biểu diễn tài liệu JSON cho đối tượng sinh viên.

**Cặp tên-giá trị**

**Đối tượng bên trongĐịa chỉgói một cái khácđối tượng bên trong:vị trí.**

{

"title":"John Doe", "date\_of\_birth":"1972-14-03", "age":23,

"địa chỉ": { "đường phố": "Johndoe Land", "mã bưu chính": "LON1DN", "vị trí":{

"vĩ độ": "51.5",

"dài": "0,12",

}

}

}

**Thông tin văn bản được cung cấp dưới dạng JSONsợi dâyloại (được đặt trong dấu ngoặc kép) được chuyển đổi thành dữ liệu thích hợp**

**kiểu (chữ) trong Elasticsearch.**

**Thông tin ngày tháng dưới dạng chuỗi JSON được chuyển đổi thành Elasticsearchngàykiểu.**

**Vị trí là được cung cấp dưới dạng vĩ độ và kinh độ, được liên kết vớiđiểm địa lýnhập vào Elasticsearch.**

**Hình 3.3 Một tài liệu của sinh viên được thể hiện ở định dạng JSON**

**68 CPHẦN3*Ngành kiến ​​​​trúc***

Chúng tôi mô tả các thuộc tính của học sinh bằng các cặp tên-giá trị. Các trường tên luôn là chuỗi trong dấu ngoặc kép và các giá trị tuân theo các kiểu dữ liệu của JSON (int, string, boolean, v.v.). Các đối tượng bên trong có thể được biểu diễn dưới dạng các đối tượng lồng nhau, mà JSON hỗ trợ, chẳng hạn như trường địa chỉ cho tài liệu học sinh. Lưu ý rằng JSON không có kiểu ngày được xác định, nhưng thông lệ của ngành là cung cấp dữ liệu ngày và giờ dưới dạng chuỗi (tốt nhất là theo định dạng ISO 8601: yyyy-MM-dd hoặc với thành phần thời gian, yyyy-MM-ddThh:mm:ss). Elasticsearch phân tích thông tin ngày thành chuỗi để trích xuất dữ liệu dựa trên định nghĩa lược đồ.

Elasticsearch sử dụng trình phân tích cú pháp JSON để giải mã dữ liệu thành các loại phù hợp dựa trên các quy tắc ánh xạ có sẵn trên chỉ mục. Mỗi chỉ mục có một tập hợp các quy tắc ánh xạ mà Elasticsearch áp dụng khi một tài liệu đang được lập chỉ mục hoặc một truy vấn tìm kiếm được thực hiện. Chúng ta tìm hiểu thêm về ánh xạ trong chương 4.

Elasticsearch đọc các giá trị từ tài liệu JSON và chuyển đổi chúng thành các kiểu dữ liệu cụ thể để phân tích và lưu trữ. Cũng giống như JSON hỗ trợ lồng dữ liệu (một đối tượng cấp cao nhất bao gồm một đối tượng cấp tiếp theo được gói lại với một đối tượng khác bên dưới), Elasticsearch cũng hỗ trợ các cấu trúc dữ liệu lồng nhau.

**R****PHÂN TÍCH CƠ SỞ DỮ LIỆU ELATIONAL**

Nếu bạn có một số hiểu biết về cơ sở dữ liệu quan hệ, phép loại suy được thể hiện trong hình

3.4 có thể giúp ích. Hãy nhớ rằng, Elasticsearch có thể được sử dụng như một máy chủ lưu trữ (mặc dù tôi không đề xuất sử dụng dịch vụ lưu trữ chính—chúng ta sẽ thảo luận về lý do sau) giống như bất kỳ cơ sở dữ liệu nào. So sánh và đối chiếu Elasticsearch với cơ sở dữ liệu là công bằng. Tài liệu JSON từ hình 3.3 tương đương với một bản ghi trong bảng của cơ sở dữ liệu quan hệ (hình 3.4).



**Biểu diễn JSON của dữ liệu sinh viên trong**

**Elasticsearch (dữ liệu không được chuẩn hóa)**

**Dữ liệu sinh viên được biểu diễn dưới dạng dữ liệu quan hệ trong một**

**cơ sở dữ liệu (dữ liệu được chuẩn hóa)**

{

"title":"John Doe", "ngày\_sinh

":"1972-14-03",

"tuổi":23,

"Địa chỉ":{

//..

}

}

**Các bản ghi được chia thành hai bảng riêng biệt và được nối với nhau bằng khóa ngoại giữa hai bảng này.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | HỌC SINHBÀN | |
| NHẬN DẠNG | TIÊU ĐỀ | NGÀY\_Sinh | TUỔI | | ĐỊA CHỈ\_ID |
| 1 | John  con nai | 1972-14-03 | 23 | | 123456 |
|  |  |  |  | |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | | BẢNG ĐỊA CHỈ | |
| ĐỊA CHỈ\_ID | ĐỊA CHỈ\_DÒNG1 | | MÃ BƯU ĐIỆN |
| 123456 | 34,Johndoe LD1DNN  Đất đai, London | | |
|  |  | |  |

**Hình 3.4 Một tài liệu JSON so với cấu trúc bảng cơ sở dữ liệu quan hệ**

***3.2 Tòa nhàkhối* 69**

Dữ liệu khá giống nhau ngoại trừ cách nó được hình thành, định dạng và lưu trữ trong các cấu trúc này: cấu trúc cơ sở dữ liệu là quan hệ, trong khi cấu trúc trong Elasticsearch là phi chuẩn hóa và không quan hệ. Một tài liệu trong Elasticsearch bao gồm thông tin độc lập không có quan hệ.

Như bạn có thể thấy ở phía bên phải trong hình 3.4, các bảng STUDENT và ADDRESS có liên quan với nhau thông qua khóa ngoại ADDRESS\_ID, xuất hiện trong cả hai bảng. Elasticsearch không có khái niệm về mối quan hệ (hoặc, chúng ta có thể nói, một phạm vi giới hạn các mối quan hệ được hỗ trợ), do đó toàn bộ tài liệu của sinh viên được lưu trữ trong một chỉ mục duy nhất. Các đối tượng bên trong (trường địa chỉ trong hình 3.4) nằm trong cùng một chỉ mục với các trường chính.

**GHI CHÚ**Dữ liệu trong Elasticsearch được phi chuẩn hóa để hỗ trợ tìm kiếm và truy xuất nhanh chóng, không giống như trong cơ sở dữ liệu quan hệ, nơi dữ liệu được chuẩn hóa ở nhiều dạng khác nhau. Mặc dù bạn có thể tạo mối quan hệ cha-con ở một mức độ nào đó trong Elasticsearch, nhưng việc làm như vậy có thể dẫn đến tình trạng tắc nghẽn và suy giảm hiệu suất. Nếu dữ liệu của bạn được mong đợi là quan hệ, Elasticsearch có thể không phải là giải pháp phù hợp.

Cũng giống như chúng ta có thể chèn nhiều bản ghi vào một bảng, chúng ta có thể lập chỉ mục nhiều tài liệu JSON vào Elasticsearch. Tuy nhiên, không giống như trong cơ sở dữ liệu quan hệ, nơi chúng ta phải tạo sơ đồ bảng trước, Elasticsearch (giống như các cơ sở dữ liệu NoSQL khác) cho phép bạn chèn tài liệu mà không cần sơ đồ được xác định trước. Tính năng không có sơ đồ này hữu ích trong quá trình thử nghiệm và phát triển nhưng cũng có thể là vấn đề trong môi trường sản xuất.

**D****HOẠT ĐỘNG OCUMENTGiao diện lập trình ứng dụng (API)S**

Theo tiêu chuẩn trên Elasticsearch, việc lập chỉ mục tài liệu được mong đợi sẽ sử dụng các API tài liệu được xác định rõ ràng. Có hai loại API tài liệu: loại hoạt động trên một tài liệu duy nhất và loại hoạt động trên nhiều tài liệu cùng một lúc (lô). Chúng ta có thể lập chỉ mục hoặc truy xuất từng tài liệu một bằng cách sử dụng API tài liệu đơn hoặc nhóm chúng lại bằng cách sử dụng API nhiều tài liệu (được hiển thị dưới dạng API RESTful qua HTTP). Chúng được mô tả ngắn gọn ở đây:

* *API tài liệu đơn*—Thực hiện các hành động trên từng tài liệu, từng cái một. Giống như các hoạt động liên quan đến CRUD (tạo, đọc, cập nhật và xóa), các API này cho phép chúng ta lấy, lập chỉ mục, xóa và cập nhật tài liệu.
* *API nhiều tài liệu*—Làm việc với nhiều tài liệu cùng lúc. Chúng cho phép chúng ta xóa và cập nhật nhiều tài liệu bằng một truy vấn duy nhất, lập chỉ mục hàng loạt và lập chỉ mục lại dữ liệu từ chỉ mục nguồn sang chỉ mục đích.

Mỗi API có cách sử dụng cụ thể, chúng tôi sẽ thảo luận chi tiết trong chương 5.

Khi chúng ta đang nói về chủ đề tài liệu, một điều thường nảy sinh: các loại tài liệu. Mặc dù các loại tài liệu đã lỗi thời và bị xóa khỏi phiên bản 8.x, chúng có thể xuất hiện trên radar của bạn khi làm việc với Elasticsearch và khiến bạn bối rối, đặc biệt là với các phiên bản cũ hơn (phiên bản 5.x trở xuống) của Elasticsearch. Vì vậy, hãy dành một chút thời gian để hiểu chúng là gì và trạng thái hiện tại của chúng trước khi tìm hiểu về chỉ mục.

**70 CPHẦN3*Ngành kiến ​​​​trúc***

**Loại bỏ các loại**



Dữ liệu chúng tôi lưu trữ có hình dạng cụ thể: tài liệu phim có các thuộc tính liên quan đến phim, dữ liệu cho ô tô có các thuộc tính liên quan đến ô tô và tài liệu nhân viên có dữ liệu liên quan đến bối cảnh việc làm và kinh doanh. Chúng tôi lập chỉ mục các tài liệu JSON này vào các nhóm hoặc bộ sưu tập tương ứng: tài liệu phim bao gồm dữ liệu phim cần được lưu trong chỉ mục có tênphim, ví dụ. Do đó, chúng tôi lập chỉ mục một tài liệu có loạiBộ phimvào mộtphimchỉ số,Xe hơivào mộtxe ô tôchỉ mục, v.v.Nghĩa là, một tài liệu phim mà chúng ta lập chỉ mục vào Elasticsearch có một loại được gọi làBộ phim. Tương tự như vậy, tất cả các giấy tờ xe đều thuộcXe hơiloại, tài liệu nhân viên theoNgười lao độngloại, v.v.

Trước phiên bản 5.*x*, Elasticsearch cho phép người dùng lập chỉ mục nhiều loạicủa các tài liệu trong một chỉ mục duy nhất. Nghĩa là, mộtxe hơichỉ mục có thể bao gồm các loại nhưXe ô tô,Hiệu suất-Xe ô tô,Các mặt hàng xe hơi,Bán xe,Đại lýShowRooms,Xe đã qua sử dụng, hoặc thậm chíKhách hàng,Đơn hàngv.v. Mặc dù nghe có vẻ là một kế hoạch hay để tập hợp tất cả các mẫu xe liên quan đến ô tô ở một nơi, nhưng nó lại có một hạn chế.

Trong cơ sở dữ liệu, các cột trong một bảng là độc lập với nhau. Thật không may, Elasticsearch không như vậy. Các trường tài liệu có thể có nhiều loại khác nhau, nhưng chúng tồn tại trong cùng một chỉ mục. Điều đó có nghĩa là một trường trong một loại cóchữkiểu dữ liệu có thể-không có loại khác, chẳng hạn nhưngày, trong một chỉ mục khác. Điều này là do cách Lucene duy trì các kiểu trường trong một chỉ mục. Vì Lucene quản lý các trường ở cấp độ chỉ mục nên không có sự linh hoạt để khai báo hai trường có kiểu dữ liệu khác nhau trong cùng một chỉ mục.

Một loại duy nhất cho mỗi chỉ mục đã được giới thiệu bắt đầu từ phiên bản 6.0, vì vậy, ví dụ,xe ô tôchỉ số dự kiến ​​sẽ giữ nguyênxe hơitài liệu. Khi lập chỉ mục mộtxe hơitài liệument, tên chỉ mục phải được theo sau bởi loại: ví dụ,ĐẶT xe/xe/ 1chỉ số axe hơitài liệu có ID1và đưa nó vào mộtxe ô tômục lục.

Tuy nhiên, các API đã được nâng cấp bắt đầu từ phiên bản 7.0.0: loại tài liệu được thay thế bằng loại tài liệu mặc định\_tài liệu, hiện là một vật cố định cố định trongURL như một điểm cuối. Do đó, URL làĐẶT xe/\_doc/1. Như hình minh họa, Elasticsearch cho phép chúng ta sử dụng các kiểu dữ liệu, nhưng nó sẽ đưa ra cảnh báo (như hiển thị bên phải trong hình) nếu phiên bản của chúng ta trước 8.x, khuyên chúng ta nên sử dụng điểm cuối không có kiểu dữ liệu.

**URL bao gồm loại (xe hơi) của tài liệu (đã lỗi thời và bị xóa trong phiên bản 8.0).**

#! [loại bỏ kiểu] Việc chỉ định kiểu trong yêu cầu chỉ mục tài liệu đã lỗi thời, hãy sử dụng các điểm cuối không có kiểu thay thế (/{index}/\_doc/{id},

/{index}/\_doc hoặc /{index}/\_create/{id}).

ĐẶT xe/xe/1

{

"hãng sản xuất":"Toyota", "mẫu":"Avensis"

}

**> V7.0: Kiểu rõ ràng được thay thế bằng điểm cuối có tên\_tài liệu**

**(không phải loại tài liệu).**

ĐẶT xe/\_doc/1

{

...

}

{

"\_index" : "xe hơi",

"\_type" : "xe hơi",

"\_id" : "1",

"\_version" : 1, "result" : "đã tạo", "\_shards" : {

"tổng cộng" : 2,

"thành công" : 1,

"thất bại" : 0

},

"\_seq\_no" : 0,

"\_primary\_term" : 1

}

**Trong khi sử dụng loại**

**được phép lên đến phiên bản**

**7.x (bạn sẽ nhận được cảnh báo như hiển thị ở đây), bạn nên loại bỏ hoàn toàn kiểu này.**

**Cảnh báo sử dụng**

**lập chỉ mục tài liệu đã đánh máy trước phiên bản 8**

***3.2 Tòa nhàkhối* 71**

Luôn mô hình hóa dữ liệu của bạn sao cho mỗi chỉ mục có một hình dạng dữ liệu cụ thể. Khi chúng ta lập chỉ mục một tài liệu, chúng ta tạo ra một ánh xạ một-một giữa tài liệu và chỉ mục. Nghĩa là, một chỉ mục chỉ có thể có một loại tài liệu. Chúng tôi tuân theo nguyên tắc này trong suốt cuốn sách.

Bây giờ chúng ta biết dữ liệu của mình được trình bày dưới dạng tài liệu JSON để lưu trữ trong Elastic-search, bước hợp lý tiếp theo là tìm ra nơi lưu trữ các tài liệu này. Giống như một bảng trong cơ sở dữ liệu lưu trữ các bản ghi, một thùng đặc biệt (bộ sưu tập) được gọi là*chỉ số*giữ tất cả các tài liệu có hình dạng cụ thể trong Elasticsearch. Chi tiết có trong phần tiếp theo.

* + 1. Chỉ số

Chúng ta cần một container để lưu trữ tài liệu của mình trong kho lưu trữ. Đối với điều này, Elasticsearch tạo một chỉ mục dưới dạng một tập hợp hợp lý các tài liệu. Cũng giống như chúng ta lưu trữ tài liệu giấy trong tủ hồ sơ, Elasticsearch lưu trữ tài liệu dữ liệu trong một chỉ mục, ngoại trừ chỉ mục không phải là nơi lưu trữ vật lý; đó là một nhóm hợp lý. Nó bao gồm (hoặc được sao lưu bởi)*mảnh vỡ*Hình 3.5 hiển thị chỉ mục cars bao gồm ba phân đoạn trên ba nút (mỗi nút là một phiên bản của Elasticsearch), mỗi nút có một phân đoạn.

**Mỗi mảnh vỡđược tạo thành từ hai bản sao (bản sao).**



Dữ liệu trong

xe hơi

**MỘTxe hơitài liệu đang được lập chỉ mục.**

xe ô tô

chỉ số

Nút A

Nút B

**Có bản sao của các mảnh nút khác**

**trên nút hiện tại.**

Mảnh vỡ 1

Bản sao 2

Bản sao 3

**Mỗi nút là một phiên bản đang chạy của Elasticsearch.**

Bản sao 3

Mảnh vỡ 2

Bản sao 1



xe hơi

Bản sao 1

**Dựa trên định tuyếnthuật toán, mộtxe hơitài liệu có thể đi**

**đến Shard 3.**

Nút C

Mảnh vỡ 3

**Hình 3.5 Một chỉ mục được thiết kế với ba mảnh và hai bản sao cho mỗi mảnh**

Bản sao 2

Ngoài việc có ba mảnh, chỉ mục được khai báo có hai bản sao cho mỗi mảnh, cả hai đều được lưu trữ trên các nút khác. Điều đó dẫn đến một câu hỏi: mảnh là gì?

*Mảnh vỡ*là các trường hợp vật lý của Apache Lucene, những con ngựa thồ đằng sau hậu trường trong việc đưa dữ liệu của chúng ta vào và ra khỏi bộ lưu trữ. Nói cách khác, các phân đoạn đảm nhiệm việc lưu trữ và truy xuất dữ liệu vật lý của chúng ta.

**72 CPHẦN3*Ngành kiến ​​​​trúc***

**GHI CHÚ**Bắt đầu từ phiên bản 7.x, bất kỳ chỉ mục nào được tạo ra đều được thiết lập để được sao lưu bởi một phân đoạn duy nhất và một bản sao theo mặc định. Các phân đoạn và bản sao của chỉ mục có thể được định cấu hình và tùy chỉnh dựa trên nhu cầu về kích thước dữ liệu. Trong khi các bản sao có thể được thay đổi kích thước, số lượng phân đoạn không thể được sửa đổi trên một chỉ mục trực tiếp.

Shard có thể được phân loại thành primary và replica. Primary shard lưu trữ tài liệu, trong khi replica shard (hay đơn giản là replica), như tên gọi của nó, là bản sao của primary shard. Chúng ta có thể có một hoặc nhiều bản sao cho mỗi shard. Chúng ta cũng có thể không có repli-cas, nhưng thiết kế như vậy không được khuyến khích cho môi trường sản xuất—trong môi trường sản xuất thực tế, chúng ta tạo nhiều bản sao cho mỗi shard. Replica lưu trữ bản sao dữ liệu, tăng tính dự phòng của hệ thống và do đó giúp tăng tốc truy vấn tìm kiếm. Chúng ta tìm hiểu thêm về shard và replica trong phần tiếp theo.

Một chỉ mục có thể chứa bất kỳ số lượng tài liệu nào (nhưng như chúng ta đã thảo luận trước đó, chỉ có một loại tài liệu), vì vậy, bạn nên tìm một kích thước tối ưu cho nhu cầu của mình (chúng ta sẽ thảo luận về kích thước phân mảnh trong chương 14). Mỗi chỉ mục có thể tồn tại trên một nút duy nhất hoặc được phân phối trên nhiều nút trong cụm. Sử dụng thiết kế trong hình 3.6 (một chỉ mục với một phân mảnh và hai bản sao), mọi tài liệu chúng ta lập chỉ mục sẽ được lưu trữ dưới dạng ba bản sao: một bản trong một phân mảnh và hai bản sao trên các bản sao.

**GHI CHÚ**Phải thực hiện việc định cỡ chỉ mục tối ưu trước khi thiết lập chỉ mục. Kích thước phụ thuộc vào dữ liệu chúng ta có thể phải phục vụ ngày hôm nay cũng như kích thước ước tính cho tương lai. Bản sao tạo ra các bản sao dữ liệu bổ sung, vì vậy hãy đảm bảo bạn tạo dung lượng khi cân nhắc đến điều này. Chúng ta sẽ xem xét việc định cỡ chỉ mục trong chương 14.

Mỗi chỉ mục đều có các thuộc tính như ánh xạ, cài đặt và bí danh. Ánh xạ là quá trình xác định các định nghĩa lược đồ, trong khi cài đặt cho phép chúng ta định cấu hình các phân đoạn và bản sao. Bí danh là tên thay thế được đặt cho một chỉ mục duy nhất hoặc một tập hợp các chỉ mục. Một số cài đặt, chẳng hạn như số lượng bản sao, có thể được đặt động; nhưng các thuộc tính khác, như số lượng phân đoạn, không thể thay đổi khi chỉ mục đang hoạt động. Lý tưởng nhất là chúng ta nên tạo các mẫu để định cấu hình bất kỳ chỉ mục mới nào.

Chúng tôi sử dụng API RESTful để làm việc trên các chỉ mục: tạo và xóa chỉ mục, thay đổi cài đặt của chúng, đóng và mở chúng, lập chỉ mục lại dữ liệu và các hoạt động khác. Chúng tôi sẽ làm việc thông qua các API này một cách rộng rãi trong chương 6.

* + 1. Luồng dữ liệu

Các chỉ mục (như movies, movie\_reviews, v.v.) lưu trữ và thu thập dữ liệu. Theo thời gian, chúng có thể trở nên rất lớn khi dữ liệu được tích lũy và lưu trữ nhiều hơn. Việc thêm các nút có thể làm giảm vấn đề này bằng cách phân phối các mảnh trên nhiều nút hơn. Kỳ vọng là dữ liệu như vậy không cần phải được chuyển sang các chỉ mục mới hơn theo định kỳ (hàng giờ, hàng ngày hoặc hàng tháng). Ghi nhớ điều này, chúng ta hãy xem xét một loại dữ liệu khác: dữ liệu chuỗi thời gian.

Như tên gọi cho thấy, dữ liệu chuỗi thời gian nhạy cảm với thời gian và phụ thuộc vào thời gian. Lấy ví dụ về nhật ký được tạo bởi máy chủ web Apache, được hiển thị trong hình 3.6.

***3.2 Tòa nhàkhối* 73**



**Hình 3.6 Mẫu tệp nhật ký máy chủ web Apache**

Nhật ký được ghi liên tục vào tệp nhật ký của ngày hiện tại. Mỗi câu lệnh nhật ký có dấu thời gian liên quan. Vào nửa đêm, tệp được sao lưu bằng dấu ngày và một tệp mới được tạo cho ngày mới. Khung nhật ký tự động khởi tạo quá trình chuyển đổi trong quá trình chuyển đổi ngày.

Nếu chúng ta muốn giữ dữ liệu nhật ký trong Elasticsearch, chúng ta cần phải suy nghĩ lại về chiến lược lập chỉ mục dữ liệu thay đổi/chuyển đổi định kỳ. Chúng ta có thể viết một tập lệnh chuyển đổi chỉ mục chuyển đổi các chỉ mục vào mỗi nửa đêm hằng ngày—nhưng không chỉ chuyển đổi dữ liệu. Ví dụ, chúng ta cũng cần chuyển hướng các yêu cầu tìm kiếm đến một chỉ mục mẹ duy nhất thay vì nhiều chỉ mục chuyển đổi. Nghĩa là, chúng ta không muốn đưa ra các truy vấn như GET index1,index2,index3/\_search, chỉ định các chỉ mục riêng lẻ; thay vào đó, chúng ta muốn gọi GET myalias/search, trả về dữ liệu từ tất cả các chỉ mục cơ bản. Chúng ta tạo một bí danh cho mục đích này trong chương 6.

**SỰ ĐỊNH NGHĨA** Bí danh là tên thay thế cho một chỉ mục duy nhất hoặc một tập hợp nhiều chỉ mục. Cách lý tưởng để tìm kiếm trên nhiều chỉ mục là tạo một bí danh trỏ đến chúng. Khi chúng ta tìm kiếm trên một bí danh, về cơ bản chúng ta đang tìm kiếm trên tất cả các chỉ mục được quản lý bởi bí danh này.

Điều này dẫn chúng ta đến một khái niệm quan trọng được gọi là luồng dữ liệu: một cơ chế chỉ mục để lưu trữ dữ liệu chuỗi thời gian. Điều này được thảo luận trong phần tiếp theo.

**TLÀM-****DỮ LIỆU SERIES**

Luồng dữ liệu chứa dữ liệu chuỗi thời gian trong Elasticsearch—chúng cho phép chúng ta lưu trữ dữ liệu trong nhiều chỉ mục nhưng cho phép truy cập như một tài nguyên duy nhất cho các truy vấn liên quan đến tìm kiếm và phân tích. Như đã thảo luận trước đó, dữ liệu được gắn thẻ theo trục ngày hoặc giờ, chẳng hạn như nhật ký, sự kiện ô tô tự động, dự báo thời tiết hàng ngày, mức độ ô nhiễm trong một thành phố, doanh số bán hàng Black Friday theo giờ, v.v., dự kiến ​​sẽ được lưu trữ trong các chỉ mục theo thời gian. Ở cấp độ cao, các chỉ mục này được gọi là luồng dữ liệu. Đằng sau hậu trường, mỗi luồng dữ liệu có một tập hợp các chỉ mục cho từng điểm thời gian. Các chỉ mục này được Elasticsearch tự động tạo và được ẩn.

**74 CPHẦN3*Ngành kiến ​​​​trúc***

Hình 3.7 cho thấy một luồng dữ liệu mẫu cho nhật ký đơn hàng thương mại điện tử được tạo và ghi lại hàng ngày. Nó cũng cho thấy luồng dữ liệu đơn hàng bao gồm các chỉ mục ẩn được tạo tự động theo ngày. Bản thân luồng dữ liệu không gì khác hơn là một bí danh cho các chỉ mục ẩn theo chuỗi thời gian (lăn) ẩn đằng sau hậu trường. Trong khi các yêu cầu tìm kiếm/đọc trải dài trên tất cả các chỉ mục ẩn hỗ trợ của luồng dữ liệu, các yêu cầu lập chỉ mục chỉ được chuyển hướng đến chỉ mục mới (hiện tại).

**Tất cả các truy vấn tìm kiếm và lập chỉ mục đều được hướng dẫn**

**đến dữ liệu đơn lẻnguồn tài nguyên luồng.**

order-logs-data-stream

chỉ mục/tìm kiếm

**Dữ liệu nhật ký đơn hàngstream đóng vai trò là bí danh để sao lưu các chỉ mục.**

**Tự động tạo các chỉ mục ẩn (chuyển đổi khi cần thiết)**

.order.logs.25-07-2021

.order.logs.25-07-2021.1

.order.logs.26-07-2021

.order.logs.24-07-2021

.order.logs.23-07-2021

**Một chỉ mục mới được tạo ra khi**

**ngày tháng trôi qua. Viết các truy vấn đi đến chỉ mục này.**

**Hình 3.7 Luồng dữ liệu bao gồm các chỉ mục ẩn được tạo tự động.**

Luồng dữ liệu được tạo bằng cách sử dụng mẫu lập chỉ mục phù hợp. Mẫu là bản thiết kế bao gồm các thiết lập và giá trị cấu hình được sử dụng để tạo tài nguyên như chỉ mục. Chỉ mục được tạo từ mẫu sẽ kế thừa các thiết lập được xác định trong mẫu. Chúng ta sẽ xem xét việc phát triển luồng dữ liệu bằng mẫu lập chỉ mục trong chương 6.

Trong vài phần trước, chúng tôi đã giải thích ngắn gọn rằng các chỉ mục và luồng dữ liệu được phân phối trên các phân đoạn và bản sao. Hãy cùng tìm hiểu sâu hơn về các thành phần này.

* + 1. Mảnh vỡ và bản sao

Shards là các thành phần phần mềm lưu trữ dữ liệu, tạo cấu trúc dữ liệu hỗ trợ (như inverted index), quản lý truy vấn và phân tích dữ liệu trong Elasticsearch. Chúng là các phiên bản của Apache Lucene được phân bổ cho một index trong quá trình tạo index. Trong quá trình lập chỉ mục, một tài liệu sẽ đi qua shard. Shards tạo các phân đoạn tệp không thể thay đổi để lưu trữ tài liệu trong một hệ thống tệp bền vững.

Lucene là một công cụ hiệu suất cao để lập chỉ mục tài liệu một cách hiệu quả. Có rất nhiều thứ diễn ra đằng sau hậu trường khi lập chỉ mục một tài liệu và Lucene thực hiện điều này rất hiệu quả. Ví dụ, các tài liệu ban đầu được sao chép vào bộ đệm trong bộ nhớ trên phân đoạn và sau đó được ghi vào các phân đoạn có thể ghi trước khi được hợp nhất và hoàn thiện thành

***3.2 Tòa nhàkhối* 75**

hệ thống lưu trữ tập tin cơ bản. Hình 3.8 cho thấy hoạt động bên trong của công cụ Lucene trong quá trình lập chỉ mục.

**Một trường hợp Lucene**



Tài liệu (DJSoOcuNm) ent

(DTiếng ViệtoOcuNm) ent

(JDSoOcNa) thì(JSON)

**Phân đoạn**

Mảnh vỡ

Chỉ mục Elasticsearch

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**



**Một mảnh Lucene bao gồm nhiều phân đoạn.**

**Lucene hợp nhất mọi thứ**

**Phân đoạn**

**ba phân đoạn thành một phân đoạn mới.**

**Hình 3.8 Cơ chế lập chỉ mục tài liệu của Lucene**

Các mảnh được phân phối trên toàn cụm để có sẵn và chuyển đổi dự phòng. Mặt khác, các bản sao cho phép dự phòng trong hệ thống. Khi một chỉ mục đang hoạt động, các mảnh không thể được di dời, vì làm như vậy sẽ làm mất hiệu lực dữ liệu hiện có trong chỉ mục.

Là bản sao trùng lặp của các mảnh, các mảnh bản sao cung cấp tính dự phòng và tính khả dụng cao trong một ứng dụng. Bằng cách phục vụ các yêu cầu đọc, các bản sao cho phép phân phối tải đọc trong thời gian cao điểm. Một bản sao không được đặt cùng vị trí trên cùng một nút với mảnh tương ứng, vì điều đó sẽ phá vỡ mục đích của tính dự phòng. Ví dụ, nếu nút bị sập, bạn sẽ mất tất cả dữ liệu từ mảnh và bản sao của nó nếu chúng được đặt cùng vị trí. Do đó, các mảnh và bản sao tương ứng của chúng được phân phối trên các nút khác nhau trong cụm.

**ĐỊNH NGHĨA**Một cụm là một tập hợp các nút. Một nút là một thể hiện của máy chủ Elasticsearch. Ví dụ, khi chúng ta khởi động máy chủ Elasticsearch trên máy của mình, về cơ bản chúng ta đang tạo một nút. Nút này tham gia vào một cụm theo mặc định. Vì cụm chỉ có nút này, nên nó được gọi là cụm một nút. Nếu chúng ta khởi động nhiều thể hiện của máy chủ hơn, chúng sẽ tham gia vào cụm này, với điều kiện là các thiết lập là chính xác.

**76 CPHẦN3*Ngành kiến ​​​​trúc***

**DPHÂN PHỐI CÁC MẢNH VẬT VÀ BẢN SAO**

Mỗi mảnh được mong đợi sẽ chứa một lượng dữ liệu nhất định. Như chúng ta đã biết, dữ liệu được phân bổ trên nhiều mảnh trên nhiều nút. Hãy xem các mảnh được phân phối như thế nào khi chúng ta bắt đầu các nút mới hoặc giảm đi khi chúng ta mất các nút.

Giả địnhchúng tôi đã tạo một chỉ mục virus\_mutations để lưu trữ dữ liệu đột biến virus COVID. Theo chiến lược của chúng tôi, chỉ mục này sẽ được cung cấp ba mảnh. (Chương 6 đề cập đến cơ chế tạo chỉ mục với một số lượng mảnh và bản sao nhất định; hiện tại, chúng ta hãy tiếp tục thảo luận về cách phân phối các mảnh.)

**Tình trạng sức khỏe của một cụm**

Elasticsearch sử dụng hệ thống đèn giao thông đơn giản dựa trên các chỉ báo để cho chúng ta biết tình trạng của cụm tại bất kỳ thời điểm nào. Một cụm có ba trạng thái có thể xảy ra sau đây:

* *MÀU ĐỎ*—Các mảnh vỡ vẫn chưa được chỉ định, vì vậy không phải tất cả dữ liệu đều có sẵn để truy vấn. Điều này thường xảy ra khi cụm đang khởi động, trong thời gian đó các mảnh vỡ ở trạng thái tạm thời.
* *MÀU VÀNG*—Bản sao chưa được chỉ định, nhưng tất cả các mảnh đã được chỉ định và đang hoạt động. Điều này có thể xảy ra khi các nút lưu trữ bản sao bị sập hoặc mới xuất hiện.
* *MÀU XANH LÁ*—Đây là trạng thái hạnh phúc khi tất cả các mảnh vỡ và

bản sao được giaoVà **Chỉ ra tình trạng sức khỏe của các mảnh vỡ bằng đèn giao thông**

phục vụ nhưhy vọng. **bảng tín hiệu**

Elasticsearch cung cấp API cụm để lấy thông tin liên quan đến cụm, bao gồm cả tình trạng cụm. Chúng ta có thể sử dụngNHẬN \_cluster/healthđiểm cuối để lấy số liệu sức khỏecator; hình tiếp theo cho thấy kết quả của cuộc gọi như vậy.

{

NHẬN \_cluster/health

"cluster\_name" : "elasticsearch",

"trạng thái": "đỏ", "hết thời gian": sai, "số lượng nút": 1,

"số\_nút\_dữ\_liệu" : 1,

"active\_primary\_shards" : 25,

"active\_shards" : 25,

"unassigned\_shards" : 24

...

}

**Lấy trạng thái của cụm bằng cách gọi điểm cuối tình trạng của cụm**

Không phải tất cả các phân đoạn đều được chỉ định và sẵn sàng (cụm đang được chuẩn bị).

Các mảnh đã được phân công và sẵn sàng, nhưng các bản sao chưa được phân công và sẵn sàng.

Các mảnh vỡ và bản sao đều đã được phân công và sẵn sàng.

**MÀU XANH LÁ**

**MÀU VÀNG**

**MÀU ĐỎ**

***3.2 Tòa nhàkhối* 77**

Chúng ta cũng có thể sử dụng lệnh gọi API \_cat (API văn bản nhỏ gọn và căn chỉnh) để lấy tình trạng của cụm. Chúng ta có thể gọi API này trực tiếp trong trình duyệt thay vì trong Kibana nếu cần.Ví dụ, gọi GET localhost:9200/\_cat/health để lấy thông tin về tình trạng của cụm bằng cách gọi lệnh trên máy chủ (localhost là nơi URL Elasticsearch đang chạy trên cổng 9200).

Khi chúng ta khởi động nút đầu tiên (Nút A), không phải tất cả các phân đoạn đều được tạo cho chỉ mục. Điều này thường xảy ra khi máy chủ mới khởi động. Elasticsearch đánh dấu trạng thái cụm này là ĐỎ, cho biết hệ thống không khỏe mạnh (xem hình 3.9).

**Nút A có hai phân đoạn chính đã sẵn sàng,**

**và mảnh thứ ba đang được tạo ra.**

**Mảnh 3 chưa có**

**vẫn chưa được thực hiện.**

Nút A

Mục lục:đột biến virus

Mảnh vỡ 3

Mảnh vỡ 2

Mảnh vỡ 1

Chưa có phân đoạn nào được chỉ định. Trạng thái cụm là ĐỎ.

**Hình 3.9 Công cụ chưa sẵn sàng và có trạng thái ĐỎ vì các phân đoạn chưa được khởi tạo đầy đủ.**

Khi Node A xuất hiện, dựa trên các thiết lập, ba mảnh được tạo trên nút này cho chỉ mục virus\_mutations (xem hình 3.10). Node A tham gia vào một cụm nút đơn mới được tạo theo mặc định. Các hoạt động lập chỉ mục và tìm kiếm có thể bắt đầu ngay lập tức, vì cả ba mảnh đều được tạo thành công. Tuy nhiên, chưa có bản sao nào được tạo. Bản sao, như bạn đã biết, là bản sao dữ liệu và được sử dụng để sao lưu. Việc tạo chúng trên cùng một nút sẽ không phải là điều đúng đắn (nếu nút này bị sập, bản sao cũng sẽ bị mất).

**Nút A có ba phân đoạn chính,**

**tất cả đã được khởi tạo và sẵn sàng.**

**Tất cả các mảnh vỡ đã sẵn sàng**

**để hành động.**

Nút A

Mục lục:đột biến virus

Mảnh vỡ 3

Mảnh vỡ 2

Mảnh vỡ 1

Tất cả các mảnh đã sẵn sàng, nhưng bản sao vẫn chưa được chỉ định. Trạng thái cụm là VÀNG.

**Hình 3.10 Một nút đơn với ba phân đoạn tham gia vào một cụm nút đơn**

**78 CPHẦN3*Ngành kiến ​​​​trúc***

Vì các bản sao vẫn chưa được khởi tạo nên có khả năng cao là chúng ta sẽ mất dữ liệu nếu có bất kỳ điều gì xảy ra với Node A. Do rủi ro này, trạng thái hoạt động của cụm được đặt thành VÀNG.

Chúng tôi biết rằng tất cả các mảnh vỡ đều nằm trên một nút duy nhất và nếu nút này bị sập vì bất kỳ lý do gì, chúng tôi sẽ mất mọi thứ. Để tránh mất dữ liệu, chúng tôi quyết định khởi động một nút thứ hai để tham gia cụm hiện có. Sau khi nút mới (Nút B) được tạo và thêm vào cụm, Elasticsearch phân phối ba mảnh vỡ ban đầu như sau:

**1** Mảnh 2 và Mảnh 3 bị xóa khỏi Nút A.

**2** Shard 2 và Shard 3 được thêm vào Node B.

Động thái này phân phối dữ liệu của chúng ta trên toàn cụm bằng cách điều chỉnh các phân đoạn, như hình 3.11 minh họa.



**Các mảnh vỡ được cân bằng**

**khi Nút B được thêm vào.**

**Mảnh 2 và Mảnh 3 là**

**được chuyển đến Nút B.**

Nút A

Mục lục:đột biến virus

Mảnh vỡ2Mảnh 3

Mảnh vỡ 1

**Hình 3.11 Các mảnh vỡ được cân bằngvào nút mới, nhưng bản sao vẫn chưa được chỉ định (trạng thái VÀNG).**

Nút B

Mục lục:đột biến virus

Tất cả các mảnh vỡ đã được phân bổ, nhưng bản sao vẫn chưa được phân bổ

đã khởi tạo. Trạng thái cụm là VÀNG.

Mảnh vỡ 3

Mảnh vỡ 2

**Nút B đã được khởi động. Các mảnh đã được cân bằng.**

**Nút B có hai nút mớicác mảnh được phân bổ.**

Ngay khi các phân đoạn được phân phối sau khi chúng ta thêm Nút B, trạng thái cụm sẽ chuyển sang MÀU VÀNG.

**GHI CHÚ** Elasticsearch cho biết tình trạng của cụm thông qua điểm cuối: GET

\_cluster/health. Điểm cuối này sẽ lấy thông tin chi tiết về cụm, bao gồm tên cụm, trạng thái của cụm, số lượng phân đoạn và bản sao, v.v.

Khi nút mới hoạt động, ngoài việc phân phối các mảnh, các bản sao được khởi tạo. Một bản sao của mỗi mảnh được tạo ra và dữ liệu được sao chép vào các bản sao này từ các mảnh tương ứng. Như đã đề cập trước đó, các bản sao không nằm trên cùng một nút với mảnh chính. Bản sao 1 là bản sao của Mảnh 1, nhưng nó được tạo ra và có sẵn trên Nút B. Tương tự như vậy, Bản sao 2 và Bản sao 3 là bản sao của Mảnh 2 và Mảnh 3, tương ứng, nằm trên Nút B, nhưng các bản sao có sẵn trên Nút A (hình 3.12).

***3.2 Tòa nhàkhối* 79**

**Bản sao của Shard 2 và Shard 3, tồn tại trên Node B**



Nút A

Mục lục:đột biến virus

Mảnh vỡ 1

Bản sao 2

Bản sao 3

**Hình 3.12 Ngày vui vẻ! Tất cả các mảnh vỡ và bản sao đã được phân bổ.**

Nút B

Mục lục:đột biến virus

Tất cả các phân đoạn được phân bổ và các bản sao được khởi tạo.

Trạng thái của cụm là XANH.

Mảnh vỡ 3

Mảnh vỡ 2

Bản sao 1

**Bản sao của Shard 1, tồn tại trên Node A**

**Nút B có hai nút mớicác mảnh được phân bổ.**

Các mảnh chính và bản sao đều đã được chỉ định và sẵn sàng. Trạng thái cụm hiện là XANH LÁ.

**RMẢNH CÂN BẰNG EBALANCE**

Luôn có nguy cơ lỗi phần cứng. Trong ví dụ của chúng tôi, điều gì xảy ra nếu Node A bị sập? Nếu Node A biến mất, Elasticsearch sẽ điều chỉnh lại các mảnh bằng cách thăng cấp Replica 1 lên Shard 1, vì Replica 1 là bản sao của Shard 1 (xem Hình 3.13).

**Nút A bị sập, khiến Shard 1 và Bản sao 2 và 3 không khả dụng.**



Nút A

Mục lục:đột biến virus

**Tai nạn**

Mảnh vỡ 1

Bản sao 2

Bản sao 3

**Bản sao 1 được thăng cấp thành Mảnh 1.**

Nút B

Mục lục:đột biến virus

Mảnh vỡ 2

Mảnh vỡ 3

**Không còn bản sao nữatồn tại trên Node B**

Bản sao 1

Tất cả các mảnh đã được chỉ định, nhưng bản sao không khả dụng. Trạng thái cụm là VÀNG.

**Nút B có**

**ba mảnh vỡ và không có bản sao.**

Mảnh vỡ 1

**Hình 3.13 Bản sao sẽ bị mất (hoặc được chuyển thành phân đoạn) khi một nút gặp sự cố.**

**80 CPHẦN3*Ngành kiến ​​​​trúc***

Bây giờ Node B là node duy nhất trong cụm và nó có ba mảnh. Vì các bản sao không còn tồn tại nữa nên trạng thái của cụm được đặt thành VÀNG. Khi DevOps đưa Node A lên, các mảnh sẽ được cân bằng lại và chỉ định lại, và các bản sao sẽ được khởi tạo, do đó hệ thống sẽ cố gắng đạt được trạng thái XANH LÁ cây lành mạnh. Lưu ý rằng Elasticsearch quản lý những thảm họa như vậy và có thể tồn tại với các tài nguyên tối thiểu nếu cần, để tránh thời gian chết và nó thực hiện điều đó ở chế độ ẩn mà không cần chúng ta phải lo lắng về bất kỳ vấn đề đau đầu nào trong số những vấn đề vận hành đó.

**S****KÍCH THƯỚC CỨNG**

Một câu hỏi thường gặp là làm thế nào để định cỡ các mảnh vỡ. Không có câu trả lời nào phù hợp với tất cả mọi người. Để có được kết quả xác đáng, chúng ta phải thực hiện các thử nghiệm định cỡ (với sự thẩm định cẩn thận) dựa trên các yêu cầu dữ liệu hiện tại và nhu cầu trong tương lai của tổ chức. Thực hành tốt nhất của ngành là định cỡ một mảnh vỡ riêng lẻ không quá 50 GB, nhưng tôi đã thấy các mảnh vỡ có kích thước lên tới 400 GB. Các chỉ mục của GitHub được trải rộng trên 128 mảnh vỡ, mỗi mảnh có kích thước 120 GB. Tôi khuyên bạn nên tạo các mảnh vỡ có kích thước từ 25 GB đến 40 GB, lưu ý đến bộ nhớ heap của nút. Nếu chúng tabiết rằng một chỉ mục phim có thể chứa tới 500 GB dữ liệu, tốt nhất là phân phối dữ liệu này thành 10 đến 20 phân đoạn được phân bổ trên nhiều nút.

Có một tham số nữa cần cân nhắc khi định cỡ các mảnh: bộ nhớ heap. Như chúng ta đã biết, các nút có tài nguyên tính toán hạn chế, chẳng hạn như bộ nhớ và dung lượng đĩa. Mỗi phiên bản Elasticsearch có thể được điều chỉnh để sử dụng bộ nhớ heap dựa trên bộ nhớ khả dụng. Tôi khuyên bạn nên lưu trữ tối đa 20 mảnh trên mỗi gigabyte bộ nhớ heap. Theo mặc định, Elasticsearch được khởi tạo với bộ nhớ 1 GB, nhưng có thể thay đổi cài đặt bằng cách chỉnh sửa tệp jvm.options trong thư mục cấu hình của quá trình cài đặt. Điều chỉnh các thuộc tính Xms và Xmx của JVM để đặt bộ nhớ heap dựa trên tính khả dụng và nhu cầu của bạn.

Điểm mấu chốt là các mảnh lưu trữ dữ liệu của chúng ta, vì vậy chúng ta phải thực hiện công việc ban đầu để có được kích thước phù hợp. Kích thước phụ thuộc vào lượng dữ liệu mà chỉ mục lưu trữ (bao gồm các yêu cầu trong tương lai) và lượng bộ nhớ heap mà chúng ta có thể phân bổ cho một nút. Mọi tổ chức phải có chiến lược cho các mảnh trước khi đưa dữ liệu lên tàu. Điều bắt buộc là phải cân bằng các yêu cầu về dữ liệu và số lượng mảnh tối ưu.

**Không thể sửa đổi các mảnh vỡ trên một chỉ mục trực tiếp**

Số lượng mảnh không thể thay đổi sau khi chỉ mục được tạo và đang hoạt động. Khi chúng ta tạo chỉ mục, Elasticsearch liên kết một mảnh duy nhất với một bản sao duy nhất theo mặc định (trước phiên bản 7, mặc định là năm mảnh và một bản sao). Trong khi số lượng mảnh được ghi trên đá, số lượng bản sao có thể thay đổi bằng API cài đặt chỉ mục trong suốt vòng đời của chỉ mục. Các tài liệu được lưu trữ trong một mảnh cụ thể theo thuật toán định tuyến:

shard\_number = hash(document\_id) % số\_phân\_mảnh\_chính

Thuật toán phụ thuộc trực tiếp vào số lượng mảnh, do đó, việc thay đổi số lượng đó trong quá trình chạy trực tiếp sẽ sửa đổi vị trí của tài liệu hiện tại và làm hỏng nó. Điều này sẽ làm ảnh hưởng đến quá trình lập chỉ mục và truy xuất đảo ngược. Tuy nhiên, vẫn có cách giải quyết: lập chỉ mục lại. Bằng cách lập chỉ mục lại, chúng ta có thể thay đổi cài đặt mảnh nếu cần. Chúng ta sẽ tìm hiểu chi tiết về cơ chế lập chỉ mục lại trong chương 5.

***3.2 Tòa nhàkhối* 81**

Chúng ta sẽ tìm hiểu về kích thước phân đoạn trong chương 14.Các mảnh và bản sao tạo thành các nút và các nút tạo thành các cụm. Chúng ta hãy tìm hiểu về chúng trong phần tiếp theo.

* + 1. Các nút và cụm

Khi bạn khởi chạy Elasticsearch, nó sẽ khởi động một phiên bản duy nhất được gọi là một nút. Mỗi nút lưu trữ một tập hợp các phân đoạn và bản sao (các phiên bản của Apache Lucene). Chỉ mục, một tập hợp logic để lưu trữ dữ liệu của chúng tôi, được tạo trên các phân đoạn và bản sao này. Hình 3.14 cho thấy một nút duy nhất tạo thành một cụm.

**Một nút tạo thành một cụm nút đơn**

**Một nút là một phiên bản đang chạy của Elasticsearch.**

Mảnh vỡ 3

**Nút**

Tài liệu (DJSoOcuNm) ent

Mục lục

xe ô tô

(DJoScOuNm)khoa học

(JSON)

**Máy chủ mong đợi các tài liệu định dạng JSON làm dữ liệu đầu vào.**

**Các tài liệu được lưu trữ dưới dạng từng mảnh.**

Mảnh vỡ 2

Mảnh vỡ 1

**Cụm nút đơn**

**Hình 3.14 Elasticsearch một nútcụm**



Mặc dù bạn có thể khởi động nhiều nút trên cùng một máy—do đó tạo ra một cụm nhiều nút—nhưng không nên làm như vậy. Các bản sao sẽ không bao giờ được tạo trên cùng một máy nơi các phân đoạn tương ứng của chúng tồn tại, dẫn đến tình huống không có bản sao lưu dữ liệu của chúng ta. Điều này khiến cụm ở trạng thái VÀNG không lành mạnh. Để khởi động một nút khác trên cùng một máy, tất cả những gì chúng ta cần làm là đảm bảo dữ liệu và đường dẫn nhật ký khác nhau (hãy xem thanh bên “Nút bổ sung trên máy cá nhân” để biết chi tiết).

**STiếng Anh-****CỤM NÚT**

Khi bạn khởi động một nút lần đầu tiên, Elasticsearch sẽ tạo một cụm mới, thường được gọi là cụm một nút. Mặc dù chúng ta có thể sử dụng cụm một nút cho mục đích phát triển, nhưng nó không phải là thiết lập cấp sản xuất. Một môi trường sản xuất thông thường có một nhóm các nút dữ liệu tạo thành các cụm (một hoặc nhiều) dựa trên các yêu cầu tìm kiếm dữ liệu và ứng dụng. Nếu chúng ta khởi động một nút khác trong cùng một mạng, nút mới được khởi tạo sẽ tham gia vào cụm hiện có, với điều kiện cụm

Thuộc tính .name trỏ đến cụm nút đơn. Chúng tôi thảo luận về thiết lập cấp sản xuất trong chương 14.

**GHI CHÚ**Một tệp thuộc tính có tên elasticsearch.yml (trong thư mục <INSTALL\_DIR>/con-fig) đề cập đến một thuộc tính có tên cluster.name, thuộc tính này quyết định tên của cụm của chúng ta. Đây là một thuộc tính quan trọng vì tất cả các nút có cùng tên sẽ kết hợp với nhau và tạo thành một cụm. Ví dụ, nếu chúng ta mong đợi một cụm 100 nút

**82 CPHẦN3*Ngành kiến ​​​​trúc***

cụm, tất cả 100 nút phải có cùng thuộc tính cluster.name. Tên cụm mặc định cho máy chủ có sẵn là elasticsearch, nhưng cách tốt nhất là cấu hình tên duy nhất cho cụm.

Cụm có thể được tăng lên bằng cách mở rộng quy mô (mở rộng theo chiều dọc) hoặc mở rộng ra (mở rộng theo chiều ngang). Khi các nút bổ sung được khởi động, chúng có thể tham gia cùng một cụm nhưcác nút hiện có miễn là thuộc tính cluster.name giống nhau. Do đó, một nhóm các nút có thể tạo thành một cụm nhiều nút, như thể hiện trong hình 3.15.

cụm.tên=

hành động

**Nút**

**B**

**Nút**

**MỘT**

**Tai nhiều nút**



cụm.tên=

hành động

**Nút**

**MỘT**

**Cụm nút đơn**

**Nút**

**C**

**là**

**Ví dụ về Elasticsearch tạo thành một cụm nút đơn với**

**cụm.tên = es\_in\_action**

**Các nút bổ sung tham gia cụm hiện có với**

**cụm.tên = es\_in\_action**

**để hình thành một cụm nhiều nút.**

**Hình 3.15 Sự hình thành cụm từ một nút đơn đến cụm nhiều nút**

Việc thêm nhiều nút hơn vào cụm không chỉ tạo ra sự dự phòng và làm cho hệ thống có khả năng chịu lỗi mà còn mang lại lợi ích hiệu suất rất lớn. Khi chúng ta thêm nhiều nút hơn, chúng ta tạo ra nhiều không gian hơn cho các bản sao. Hãy nhớ rằng số lượng phân đoạn không thể thay đổi khi chỉ mục đang hoạt động. Vậy, lợi ích của việc thêm các phân đoạn vào các nút là gì? Thông thường, dữ liệu được lập chỉ mục lại từ một chỉ mục hiện có thành một chỉ mục mới. Chỉ mục mới này được cấu hình với một số lượng phân đoạn mới, có tính đến các nút mới bổ sung.

**Quản lý không gian đĩa của nútkhôn ngoan**

Để cải thiện hiệu suất đọc, chúng ta có thể thêm bản sao, nhưng chúng đi kèm với bộ nhớ lớn hơn.ory và yêu cầu không gian đĩa. Mặc dù không phải là bất thườngKhi làm việc với Elasticsearch, chúng ta phải xem xét các yêu cầu về kích thước dữ liệu để tạo các cụm chứa terabyte hoặc thậm chí petabyte.

***3.2 Tòa nhàkhối* 83**

Ví dụ, nếu chúng ta có một*ba mảnh và 15 bản sao cho mỗi mảnh*chiến lược, với mỗiVới kích thước shard là 50 GB, chúng ta phải đảm bảo rằng tất cả 15 bản sao đều có đủ dung lượng không chỉ để lưu trữ tài liệu trên đĩa mà còn cho bộ nhớ heap:

* Bộ nhớ mảnh vỡ: 3✕50 GB/phân mảnh = 150 GB
* Bộ nhớ bản sao cho mỗi phân đoạn: 15✕50 GB/bản sao = 750 GB/mỗi phân đoạn
* (Bộ nhớ sao chép cho 3 mảnh = 3✕750GB = 2250GB)
* Tổng bộ nhớ cho cả phân đoạn và bản sao trên một nút nhất định = 150 GB + 750 GB = 900 GB
* (Tổng cộng cho 20 nút = 18 TB)

Cần tới 18 TB cho một chỉ mục với*ba mảnh và 15 bản sao cho mỗi mảnh*chiến lược. Ngoài không gian đĩa ban đầu này, chúng ta cần thêm không gian đĩa để chạy máy chủ trơn tru. Vì vậy, chúng ta phải cân nhắc kỹ lưỡng các yêu cầu về dung lượng. Chúng ta thảo luận về cách thiết lập cụm với bộ nhớ và không gian đĩa trong chương 14 và 15.

**Tôi****ĐẶC BIỆT-ĐA CỤM NÚT**

Chúng ta đã xem xét sơ qua một nút như một thể hiện của máy chủ Elasticsearch. Như đã nêu trước đó, khi chúng ta khởi động ứng dụng Elasticsearch, về cơ bản chúng ta đang khởi tạo một nút. Theo mặc định, nút này tham gia vào một cụm một nút. Chúng ta có thể tạo một cụm gồm bất kỳ số lượng nút nào dựa trên yêu cầu dữ liệu của mình. Chúng ta cũng có thể tạo nhiều cụm, như thể hiện trong hình 3.16, nhưng việc thực hiện như vậy phụ thuộc vào các trường hợp sử dụng của tổ chức.

**Một nút duy nhất trên một máy tạo thành cụm nút đơn**



Nút A



**Nhiều nút trong một trang trại**

**máy tính tạo thành một cụm**

Nút

MỘT

Nút

B

Nút

C

**Nhiều cụmvới nhiều nút tạo thành một cụm đa**

lasticsearch.guru

**Cụm dữ liệu kinh doanh**

**Cụm dữ liệu hoạt động**



VÀ

Nút

C

Nút

B

Nút

MỘT

Nút

C

Nút

B

Nút

MỘT

**Hình 3.16 Cấu hình cụm khác nhau**

**84 CPHẦN3*Ngành kiến ​​​​trúc***

Một tổ chức có thể có nhiều loại dữ liệu khác nhau, chẳng hạn như dữ liệu liên quan đến doanh nghiệp (hóa đơn, đơn đặt hàng, thông tin khách hàng, v.v.) và thông tin hoạt động (nhật ký máy chủ web và máy chủ cơ sở dữ liệu, nhật ký ứng dụng, số liệu ứng dụng, v.v.). Việc gom tất cả các loại dữ liệu vào một cụm duy nhất không phải là điều bất thường, nhưng có thể không phải là phương pháp hay nhất. Có thể là một chiến lược tốt hơn khi tạo nhiều cụm cho các dạng dữ liệu khác nhau, với các cấu hình tùy chỉnh cho từng cụm. Ví dụ: một cụm dữ liệu quan trọng đối với doanh nghiệp có thể đang chạy trên một cụm tại chỗ với các tùy chọn bộ nhớ và không gian đĩa lớn hơn được hình thành, trong khi một cụm dữ liệu giám sát ứng dụng có thiết lập hơi khác một chút.

**Nút bổ sung trên máy cá nhân**

Nếu bạn đang chạy một nút trên máy tính xách tay cá nhân của bạnhoặc PC, bạn có thể khởi tạo một nút bổ sung từ cùng một thư mục cài đặt. Chạy lại tập lệnh shell bằng cách truyền hai tham số bổ sung:đường dẫn.dữ liệuVàđường dẫn.logstùy chọn, trỏ tới các thư mục tương ứng, như được hiển thị ở đây:

$>cd <thư\_mục\_cài\_đặt>/bin

$>./elasticsearch -Epath.data=../data2 -Epath.logs=../log2

Lệnh này sẽ tạo ra một nút bổ sung với các thư mục dữ liệu và nhật ký đã cho. Nếu bạnbây giờ phát hànhNHẬN \_cat/nodeslệnh, một nút thứ hai sẽ xuất hiện trong danh sách.

Mỗi nút cần phải làm việc với các trách nhiệm cụ thể tại một số thời điểm. Một số có thể cần phải lo lắng về các hoạt động liên quan đến dữ liệu như lập chỉ mục và lưu trữ đệm, trong khi những nút khác có thể được yêu cầu phối hợp các yêu cầu và phản hồi của máy khách. Một số tác vụ cũng có thể bao gồm giao tiếp giữa các nút và quản lý cấp cụm. Để cho phép các trách nhiệm như vậy, Elasticsearch có một tập hợp các vai trò cho các nút để thực hiện một tập hợp các trách nhiệm cụ thể khi được giao cho vai trò đó. Chúng ta sẽ thảo luận về các vai trò của nút trong phần tiếp theo.

**NVAI TRÒ CỦA ODE**

Mỗi nút đóng nhiều vai trò, từ điều phối viên đến quản lý dữ liệu rồi trở thành chủ. Các nhà phát triển Elasticsearch liên tục mày mò với các nút, vì vậy thỉnh thoảng bạn có thể thấy các vai trò mới xuất hiện. Bảng 3.2 liệt kê các vai trò của nút mà chúng ta có thể phân loại.

**Bảng 3.2 Vai trò và trách nhiệm của nút**

|  |  |
| --- | --- |
| **Vai trò** | **Sự miêu tả** |
| Nút chính | Trách nhiệm chính của nó là quản lý cụm. |
| Nút dữ liệu | Chịu trách nhiệm lưu trữ và truy xuất tài liệu. |
| Tiêu thụnút | Chịu trách nhiệm chuyển đổi dữ liệu thông qua quá trình thu thập dữ liệu trước khi lập chỉ mục. |
| Nút học máy | Xử lý các công việc và yêu cầu về máy học. |
| Biến đổinút | Xử lý các yêu cầu chuyển đổi. |
| Nút phối hợp | Đây là vai trò mặc định. Nó xử lý các yêu cầu đến từ khách hàng. |

* 1. ***Tòa nhàkhối* 85**

Vai trò của các nút như sau:

* *Nút chính—*Tham gia vào các hoạt động cấp cao như tạo và xóa chỉ mục, hoạt động của nút và các công việc liên quan đến quản trị khác để quản lý cụm. Các hoạt động quản trị này là các quy trình nhẹ; do đó, một nút chính là đủ cho toàn bộ cụm. Nếu nút chính này gặp sự cố, cụm sẽ bầu một trong các nút khác làm nút chính, do đó, nút chính sẽ được chuyển giao. Các nút chính không tham gia vào các hoạt động CRUD của tài liệu, nhưng nút chính biết vị trí của các tài liệu.
* *Nút dữ liệu—*Nơi diễn ra các hoạt động lập chỉ mục, tìm kiếm, xóa và các hoạt động liên quan đến tài liệu khác. Các nút này lưu trữ các tài liệu được lập chỉ mục. Khi nhận được yêu cầu lập chỉ mục, các nút dữ liệu sẽ nhảy vào hoạt động để lưu tài liệu vào chỉ mục của nó bằng cách gọi một trình ghi trên phân đoạn Lucene. Như bạn có thể hình dung, chúng thường xuyên giao tiếp với đĩa trong các hoạt động CRUD; do đó, chúng là các hoạt động sử dụng nhiều bộ nhớ và I/O đĩa.

**GHI CHÚ**Các biến thể cụ thể của vai trò nút dữ liệu được sử dụng khi chúng tôi triển khai nhiềutriển khai theo từng tầng:dữ liệu nóng,dữ liệu\_lạnh,dữ liệu\_ấm, Vàdữ liệu\_đông\_lạivai trò. Chúng ta sẽ xem xét chúng trong chương 14.

* *Nút nhập—*Xử lý các hoạt động nhập dữ liệu như chuyển đổi và làm giàu trước khi lập chỉ mục bắt đầu. Các tài liệu được nhập thông qua hoạt động đường ống (ví dụ: xử lý tài liệu Word hoặc PDF) có thể được đưa qua quá trình xử lý bổ sung trước khi được lập chỉ mục.
* *Nút học máy—*Như tên gọi cho thấy, thực hiện các thuật toán học máy và phát hiện các bất thường. Đây là một phần của giấy phép thương mại, vì vậy bạn phải mua giấy phép X-Pack để kích hoạt các khả năng học máy.
* *Chuyển đổi nút*—Bổ sung mới nhất vào danh sách. Một nút chuyển đổi được sử dụng để tổng hợp các bản tóm tắt dữ liệu. Nó được yêu cầu để thực hiện các lệnh gọi API chuyển đổi để tạo (chuyển đổi) các chỉ mục mới được xoay vòng dựa trên các chỉ mục hiện có.
* *Nút phối hợp*—Trong khi vai trò được người dùng gán cho một nút theo mục đích (hoặc theo mặc định), có một vai trò đặc biệt mà tất cả các nút đều đảm nhận, bất kể sự can thiệp của người dùng: nút điều phối (hoặc điều phối viên). Như tên gọi của nó, điều phối viên sẽ xử lý các yêu cầu của khách hàng từ đầu đến cuối. Khi một yêu cầu được thực hiện với Elasticsearch, một trong những nút này sẽ tiếp nhận yêu cầu và đội mũ của điều phối viên. Sau khi chấp nhận các yêu cầu, điều phối viên sẽ yêu cầu các nút khác trong cụm xử lý yêu cầu. Nó chờ phản hồi trước khi thu thập và đối chiếu kết quả và gửi lại cho khách hàng. Về cơ bản, nó hoạt động như một trình quản lý công việc, phân phối các yêu cầu đến cho các nút thích hợp và phản hồi cho khách hàng.

**CVAI TRÒ ĐẶC TRƯNG**

Khi chúng ta khởi động Elasticsearch ở chế độ phát triển, nút theo mặc định được đặt bằngvai trò master, data và ingest (và mỗi nút theo mặc định là một điều phối viên—không có cờ đặc biệt nào để bật hoặc tắt một nút điều phối). Chúng ta có thể cấu hình các

**86 CPHẦN3*Ngành kiến ​​​​trúc***

vai trò dựa trên nhu cầu của chúng tôi: ví dụ, trong một cụm gồm 20 nút, chúng tôi có thể kích hoạt 3 nút làm nút chính, 15 nút làm nút dữ liệu và 2 nút làm nút tiếp nhận.

Để cấu hình một vai trò trênmột nút, tất cả những gì chúng ta cần làm là tinh chỉnh thiết lập node.roles trong tệp cấu hình elasticsearch.yml. Thiết lập này bao gồm một danh sách các vai trò: ví dụ, thiết lập node.roles: [master] cho phép nút làm nút chính. Có thể thiết lập nhiều vai trò nút, như trong ví dụ sau:

node.roles: [master, dữ liệu, nhập, ml]

**Nút này có bốn vai trò: chủ, dữ liệu, tiếp nhận và học máy.**

Hãy nhớ rằng, chúng tôi đã đề cập rằng vai trò điều phối viên là vai trò mặc định được cung cấp cho tất cả các nút. Mặc dù chúng tôi thiết lập bốn vai trò trong ví dụ (master, data, ingest và ml),nút này vẫn kế thừa vai trò điều phối viên.

Chúng ta có thể chỉ định cụ thể vai trò điều phối cho một nút bằng cách bỏ qua bất kỳ giá trị node.roles nào. Trong đoạn mã sau, chúng ta chỉ định cho nút vai trò điều phối, nghĩa là nút này không tham gia vào bất kỳ hoạt động nào khác ngoài việc điều phối các yêu cầu:

nút.vai trò: [ ]

**Để mảng vai trò trống sẽ thiết lập nút làm điều phối viên.**

Có một lợi ích khi cho phép các nút làm điều phối viên chuyên dụng: chúng hoạt động như bộ cân bằng tải, xử lý các yêu cầu và đối chiếu các tập kết quả. Tuy nhiên, rủi ro lớn hơn lợi ích nếu chúng ta cho phép nhiều nút chỉ làm điều phối viên.

Trong thời gian trước của chúng tôithảo luận, tôi đã đề cập rằng Elasticsearch lưu trữ các trường văn bản đầy đủ được phân tích trong một cấu trúc dữ liệu nâng cao được gọi là chỉ mục đảo ngược. Nếu có một cấu trúc dữ liệu mà bất kỳ công cụ tìm kiếm nào (không chỉ Elasticsearch) phụ thuộc nhiều vào, thì đó chính là chỉ mục đảo ngược. Đã đến lúc xem xét hoạt động bên trong của chỉ mục đảo ngược, để củng cố sự hiểu biết của chúng ta về quy trình phân tích văn bản, lưu trữ và truy xuất.

#### Chỉ số đảo ngược

Nếu bạn nhìn vào mặt sau của bất kỳ cuốn sách nào, thông thường bạn sẽ thấy một mục lục ánh xạ các từ khóa đến các trang có chứa chúng. Đây không gì khác ngoài một biểu diễn vật lý của một mục lục đảo ngược.

Xin chào THẾ GIỚIXin chào bạn

**Chỉ số đảo ngược**

|  |  |
| --- | --- |
| **Mã thông báo** | **Số tài liệu** |
| Xin chào | 1, 2 |
| thế giới | 1 |
| cái chết | 2 |

Tương tự như việc điều hướng đến các trang dựa trên số trang liên quan đến từ khóa bạn đang tìm kiếm trong một cuốn sách, Elasticsearch tham khảo chỉ mục đảo ngược cho các từ tìm kiếm và liên kết tài liệu. Khi công cụ tìm thấy các mã định danh tài liệu cho các từ tìm kiếm này, nó sẽ trả về toàn bộ tài liệu bằng cách truy vấn máy chủ để trả về cho máy khách. Elasticsearch sử dụng một cấu trúc dữ liệu được gọi là chỉ mục đảo ngược cho

**Hình 3.17 Dữ liệu chỉ số đảo ngượckết cấu**

mỗi trường toàn văn trong giai đoạn lập chỉ mục, như hình 3.17 minh họa.

* 1. ***đảo ngượcchỉ số* 87**

Ở cấp độ cao, chỉ mục đảo ngược là một cấu trúc dữ liệu giống như từ điển, nhưng có cả từ và danh sách tài liệu có chứa từ. Chỉ mục đảo ngược này là chìa khóa để truy xuất tài liệu nhanh hơn trong giai đoạn tìm kiếm toàn văn. Đối với mỗi tài liệu bao gồm các trường toàn văn, máy chủ sẽ tạo một chỉ mục đảo ngược tương ứng.

**GHI CHÚ**Cây khối k chiều (BKD) là cấu trúc dữ liệu đặc biệt được sử dụng để lưu trữ các trường không phải văn bản như số và hình dạng địa lý.

Chúng ta đã học một chút lý thuyết về chỉ mục đảo ngược, nhưng bây giờ hãy xem một ví dụ đơn giản về cách chúng hoạt động. Giả sử chúng ta có hai tài liệu với một trường văn bản, lời chào:

//Tài liệu 1

{

"lời chào":"Xin chào, THẾ GIỚI"

}

//Tài liệu 2

{

"lời chào":"Xin chào, bạn"

}

Trong Elasticsearch, quá trình phân tích là một chức năng phức tạp được thực hiện bởi một mô-đun phân tích. Mô-đun phân tích được tạo thành từ các bộ lọc ký tự, một bộ phân tích mã thông báo và các bộ lọc mã thông báo. Khi tài liệu đầu tiên được lập chỉ mục, như trong trường lời chào (một trường văn bản), một chỉ mục đảo ngược được tạo ra. Mỗi trường toàn văn được sao lưu bằng một chỉ mục đảo ngược. Giá trị của lời chào “Hello, World” được phân tích, phân tích mã thông báo và chuẩn hóa thành hai từ—hello và world—vào cuối quá trình. Nhưng có một vài bước ở giữa.

Hãy xem xét toàn bộ quá trình (hình 3.18). Dòng đầu vào <h2>Hello WORLD</h2> được loại bỏ các ký tự không mong muốn như đánh dấu HTML. Dữ liệu đã được dọn dẹp được chia thành các mã thông báo (nhiều khả năng là các từ riêng lẻ) dựa trên khoảng trắng, do đó hình thành

Dảiđánh dấu HTML (html\_dảibộ lọc ký tự)

<h2>Xin chào THẾ GIỚI</h2>

Phân biệt các từ (khoảng trắngbộ phân tích dữ liệu)

Xin chào THẾ GIỚI

Chữ thườngcác mã thông báo (chữ thường

|  |  |
| --- | --- |
| [Xin chào] [THẾ GIỚI] | bộ lọc mã thông báo) |
|  |
|  | |

[xin chào] [thế giới]

**Hình 3.18 Quy trình phân tích văn bản trong đó Elasticsearch xử lý văn bản**

**88 CPHẦN3*Ngành kiến ​​​​trúc***

Xin chào THẾ GIỚI. Cuối cùng, các bộ lọc mã thông báo được áp dụng để câu có thể được chuyển đổiinto tokens: [hello] [world]. Theo mặc định, Elasticsearch sử dụng một trình phân tích chuẩn để viết thường các token, như trường hợp ở đây. Lưu ý rằng dấu câu (dấu phẩy) cũng đã bị xóa trong quá trình này.

Sau các bước này, một chỉ mục đảo ngược được tạo cho trường này. Nó chủ yếu được sử dụng trong các chức năng tìm kiếm toàn văn bản. Về bản chất, nó là một bản đồ băm với các từ là khóa trỏ đến các tài liệu có các từ này. Nó bao gồm một tập hợp các từ duy nhất và tần suất các từ đó xuất hiện trên tất cả các tài liệu trong chỉ mục.

Hãy xem lại ví dụ của chúng ta. Vì tài liệu 1 (“Hello WORLD”) đã được lập chỉ mục và phân tích, nên một chỉ mục đảo ngược đã được tạo ra với các mã thông báo (các từ riêng lẻ) và các tài liệu mà chúng xuất hiện; xem bảng 3.3.

**Bảng 3.3 Từ được mã hóa cho tài liệu “Xin chào, Thế giới”**

|  |  |  |
| --- | --- | --- |
| **Từ** | **Tính thường xuyên** | **ID tài liệu** |
| Xin chào | 1 | 1 |
| thế giới | 1 | 1 |

Các từ hello và world được thêm vào chỉ mục đảo ngược cùng với ID tài liệu nơi tìm thấy những từ này (tất nhiên là ID tài liệu 1). Chúng tôi cũng lưu ý tần suất của các từ trên tất cả các tài liệu trong chỉ mục đảo ngược này.

Khi tài liệu thứ hai (“Xin chào, bạn”) được lập chỉ mục, cấu trúc dữ liệu sẽ được cập nhật (bảng 3.4).

**Bảng 3.4 Từ được mã hóa cho tài liệu “Xin chào, bạn”**

|  |  |  |
| --- | --- | --- |
| **Từ** | **Tính thường xuyên** | **ID tài liệu** |
| Xin chào | 2 | 1,2 |
| thế giới | 1 | 1 |
| cái chết | 1 | 2 |

Khi cập nhật chỉ mục đảo ngược cho từ hello, ID tài liệu của tài liệu thứ hai được thêm vào và tần suất của từ tăng lên. Tất cả các mã thông báo từ tài liệu đến được khớp với các khóa trong chỉ mục đảo ngược trước khi được thêm vào cấu trúc dữ liệu (như mate trong ví dụ này) dưới dạng bản ghi mới nếu mã thông báo được nhìn thấy lần đầu tiên.

Bây giờ chỉ mục đảo ngược đã được tạo từ các tài liệu này, khi tìm kiếm hello xuất hiện, Elasticsearch sẽ tham khảo chỉ mục đảo ngược này trước. Chỉ mục đảo ngược chỉ ra rằng từ hello có trong ID tài liệu 1 và 2, do đó các tài liệu có liên quan sẽ được lấy và trả về cho máy khách.

* 1. ***Sự liên quan* 89**

Chúng tôi đã đơn giản hóa quá mức chỉ mục đảo ngược ở đây, nhưng không sao cả, vì mục tiêu của chúng tôi là có được sự hiểu biết cơ bản về cấu trúc dữ liệu. Ví dụ, nếu bây giờ chúng tôi tìm kiếm hello mate, cả hai tài liệu 1 và 2 đều được trả về, nhưng tài liệu 2 có thể có điểm liên quan cao hơn tài liệu 1 vì nội dung của tài liệu 2 khớp với truy vấn.

Trong khi chỉ mục đảo ngược được tối ưu hóa để truy xuất thông tin nhanh hơn, nó làm tăng độ phức tạp cho quá trình phân tích và đòi hỏi nhiều không gian hơn. Chỉ mục đảo ngược tăng lên khi hoạt động lập chỉ mục tăng lên, do đó tiêu tốn tài nguyên tính toán và không gian heap. Chúng ta sẽ không làm việc trực tiếp với cấu trúc dữ liệu, nhưng việc hiểu những điều cơ bản sẽ giúp ích.

Chỉ số đảo ngược cũng giúp suy ra điểm liên quan; nó cung cấp tần suất của các thuật ngữ, là một trong những thành phần để tính điểm liên quan. Chúng tôi đã sử dụng thuật ngữ liên quan trong một thời gian và đã đến lúc hiểu thuật ngữ này là gì và các thuật toán mà công cụ tìm kiếm như Elasticsearch sử dụng để tìm kiếm kết quả liên quan cho người dùng. Chúng tôi thảo luận về các khái niệm liên quan đến tính liên quan trong phần tiếp theo.

#### Sự liên quan

Các công cụ tìm kiếm hiện đại không chỉ trả về kết quả dựa trên tiêu chí truy vấn mà còn phân tích và trả về kết quả có liên quan nhất. Nếu bạn là nhà phát triển hoặc kỹ sư DevOps, rất có thể bạn đã sử dụng Stack Overflow để tìm kiếm câu trả lời cho các vấn đề kỹ thuật. Tìm kiếm trên Stack Overflow không bao giờ làm tôi thất vọng (ít nhất là trong hầu hết các trường hợp). Kết quả tôi nhận được cho một truy vấn gần giống với những gì tôi đang tìm kiếm. Các kết quả được sắp xếp theo thứ tự hợp lý, từ các kết quả có liên quan cao ở trên cùng đến các kết quả ít liên quan nhất ở dưới cùng. Rất khó có khả năng kết quả của một truy vấn không đáp ứng được yêu cầu của bạn—nhưng bạn có thể không quay lại Stack Overflow nếu các kết quả nằm rải rác khắp nơi.

* + 1. Sự liên quan điểm số

Stack Overflow áp dụng một bộ thuật toán liên quan để tìm kiếm và sắp xếp kết quả trả về cho người dùng. Tương tự như vậy, Elasticsearch trả về kết quả cho các truy vấn toàn văn được sắp xếp theo điểm liên quan.

*Sự liên quan*là số dấu phẩy động dương xác định thứ hạng của kết quả tìm kiếm. Elasticsearch sử dụng thuật toán liên quan Best Match 25 (BM25) theo mặc định để chấm điểm kết quả trả về để khách hàng có thể mong đợi kết quả liên quan. Đây là phiên bản nâng cao của thuật toán tương tự tần suất thuật ngữ/tần suất tài liệu nghịch đảo (TF/IDF) đã sử dụng trước đó (xem phần tiếp theo để biết thêm về thuật toán liên quan).

Nếu chúng ta đang tìm kiếm, ví dụ, Java trong tiêu đề của một cuốn sách, một tài liệu có nhiều hơn một lần xuất hiện từ Java trong tiêu đề sẽ có liên quan hơn các tài liệu có một hoặc không có lần xuất hiện nào trong tiêu đề. Hãy xem kết quả ví dụ trong hình 3.19, được Elasticsearch truy xuất cho truy vấn tìm kiếm từ khóa Java trong tiêu đề (ví dụ đầy đủ được cung cấp trong kho lưu trữ của cuốn sách).

**90 CPHẦN3*Ngành kiến ​​​​trúc***

NHẬN sách/\_tìm kiếm

{

"\_source": "tiêu đề", "truy vấn": {

"cuộc thi đấu": {

"tiêu đề": "Java"

}

}

}

"lượt truy cập" : {

…

**"điểm\_tối\_đa" : 0.33537668,**

"lượt truy cập" : [{

**"\_score" : 0.33537668,**

"\_source" : { "title" : "Có hiệu lực

Java" }

},

{

**"\_score" : 0.30060259,**

"\_source" : { "title" : "Đầu tiên

Java" }

},

{

**"\_score" : 0,18531466,**

"\_source" : { "title" : "Kiểm thử theo hướng: TDD và TDD chấp nhận cho các nhà phát triển Java"}

}…]

}

**Hình 3.19 Kết quả có liên quan cho Java trong tìm kiếm tiêu đề**

Kết quả đầu tiên cho thấy điểm liên quan cao hơn (0,33537668) so với kết quả thứ hai và thứ ba. Tất cả các tiêu đề đều bao gồm từ Java, do đó để tính điểm liên quan, Elasticsearch sử dụng thuật toán chuẩn độ dài trường: từ tìm kiếm trong tiêu đề đầu tiên (Java hiệu quả, chứa hai từ) có liên quan hơn từ tìm kiếm trong tiêu đề thứ hai.ond title (Head First Java, gồm ba từ). max\_score là điểm cao nhất trong tất cả các điểm có sẵn, thường là điểm của tài liệu đầu tiên.

Điểm liên quan được tạo ra dựa trên các thuật toán tương đồng được sử dụng. Trong ví dụ này, Elasticsearch mặc định áp dụng thuật toán BM25 để xác định điểm. Thuật toán này phụ thuộc vào tần suất thuật ngữ, tần suất tài liệu nghịch đảo và độ dài trường, nhưng Elasticsearch cho phép linh hoạt hơn nhiều: nó cung cấp một số thuật toán ngoài BM25. Các thuật toán này được đóng gói trong một mô-đun có tên là similarity, xếp hạng các tài liệu khớp. Chúng ta hãy xem một số thuật toán tương đồng này.

* + 1. Thuật toán liên quan (tương tự)

Elasticsearch sử dụng một số thuật toán liên quan, mặc định là BM25. BM25 và TF-IDF đều là các thuật toán liên quan được sử dụng trong Elasticsearch để chấm điểm và xếp hạng tài liệu. Chúng khác nhau ở cách tính trọng số thuật ngữ và điểm tài liệu.

* + - * *TF-IDF (Tần suất thuật ngữ-Tần suất tài liệu nghịch đảo).*Thuật toán TF-IDF là một lược đồ trọng số truyền thống gán trọng số cho một thuật ngữ trong một tài liệu dựa trên tần suất thuật ngữ (TF) và tần suất tài liệu nghịch đảo (IDF). TF là số lần một thuật ngữ xuất hiện trong một tài liệu, trong khi IDF là thước đo mức độ phổ biến hoặc hiếm gặp của một thuật ngữ trên toàn bộ tập tài liệu. Các thuật ngữ

***3.4 Sự liên quan* 91**

xuất hiện thường xuyên hơn trong một tài liệu cụ thể và ít thường xuyên hơn trong toàn bộ bộ sưu tập nên được coi là có liên quan hơn theo thuật toán TF-IDF.

* + - * + *Thuật toán BM25 (Best Matching 25).*Thuật toán liên quan BM25 là một cải tiến so với thuật toán TF-IDF. Thuật toán này giới thiệu hai sửa đổi quan trọng đối với thuật toán cơ sở ở chỗ nó sử dụng một hàm phi tuyến tính cho tần suất thuật ngữ để ngăn các thuật ngữ lặp lại nhiều nhận được điểm số quá cao. Thuật toán này cũng sử dụng một hệ số chuẩn hóa độ dài tài liệu để chống lại sự thiên vị đối với các tài liệu dài hơn. Trong thuật toán TF-IDF, các tài liệu dài hơn có nhiều khả năng có tần suất thuật ngữ cao hơn, điều này có nghĩa là chúng có thể nhận được tín dụng không đáng có với điểm số cao hơn. Nhiệm vụ của BM25 là tránh sự thiên vị như vậy.

Vì vậy, cả BM25 và TF-IDF đều là thuật toán liên quan được sử dụng trong Elasticsearch. BM25 được coi là cải tiến so với TF-IDF do tính năng bão hòa tần suất thuật ngữ và chuẩn hóa độ dài tài liệu. Do đó, BM25 được kỳ vọng sẽ chính xác hơn và trả về kết quả tìm kiếm có liên quan.

Elasticsearch cung cấp một mô-đun có tên là similarity cho phép chúng ta áp dụng thuật toán phù hợp nhất nếu thuật toán mặc định không phù hợp với yêu cầu của chúng ta.

Thuật toán tương tự được áp dụng cho mỗi trường bằng cách sử dụng API ánh xạ. Vì Elastic-search linh hoạt nên nó cũng cho phép tùy chỉnh các thuật toán dựa trên yêu cầu của chúng tôi. (Đây là một tính năng nâng cao, vì vậy thật không may là chúng tôi không thảo luận nhiều về nó trong cuốn sách này.) Bảng 3.5 liệt kê các thuật toán có sẵn.

**Bảng 3.5 Thuật toán tương tự của Elasticsearch**

|  |  |  |
| --- | --- | --- |
| **Thuật toán tương tự** | **Kiểu** | **Sự miêu tả** |
| Okapi BM25 (mặc định) | BM25 | Một thuật toán TF/IDF nâng cao xem xét độ dài trường ngoài tần suất thuật ngữ và tài liệu |
| Phân kỳ từ ngẫu nhiên (DFR) | DFR | Sử dụng khuôn khổ DFR do các tác giả của nó là Amati và Rijsbergen phát triển, nhằm mục đích cải thiện tính liên quan của tìm kiếm bằng cách đo lường sự khác biệt giữa thuật ngữ thực tế và phân phối ngẫu nhiên dự kiến. Các thuật ngữ xuất hiện thường xuyên hơn trong các tài liệu có liên quan so với phân phối ngẫu nhiên được gán trọng số cao hơn khi xếp hạng kết quả tìm kiếm. |
| Phân kỳ từ Độc lập (DFI) | DFI | Một mô hình cụ thể của họ DFR đo lường sự phân kỳ của phân phối tần suất thuật ngữ thực tế so với phân phối độc lập. DFI hướng đến việc gán điểm cao hơn cho các tài liệu bằng cách so sánh tần suất thuật ngữ quan sát được với tần suất mong đợi trong tần suất thuật ngữ ngẫu nhiên, không tương quan. |
| LM Dirichlet | LMDirichlet | Tính toán sự liên quancủa các tài liệu dựa trên xác suất tạo ra các thuật ngữ truy vấn từ mô hình ngôn ngữ của tài liệu |
| LM Jelinek-Mercer | L.M. Jelinek- Mercer | Cung cấp kết quả tìm kiếm được cải thiệnsự liên quan so với các mô hình không tính đến sự thưa thớt dữ liệu |
| Bị thương |  | Tạo một tập lệnh thủ công |
| Sự tương đồng Boolean | Boolean | Không xem xét các yếu tố xếp hạng trừ khi các tiêu chí truy vấn được đáp ứng |

**92 CPHẦN3*Ngành kiến ​​​​trúc***

Trong phần tiếp theo, chúng tôi sẽ tóm tắt lại thuật toán BM25, đây là thuật toán TF/IDF cải tiến thế hệ tiếp theo.

**TANH TATHE****CHE PHỦBM25THUẬT TOÁN**

Có ba yếu tố chính liên quan đến việc liên kết điểm liên quan với kết quả: tần suất thuật ngữ (TF), tần suất tài liệu nghịch đảo (IDF) và chuẩn độ dài trường. Hãy cùng xem xét các yếu tố này một cách ngắn gọn và tìm hiểu cách chúng ảnh hưởng đến tính liên quan.

*Tần suất thuật ngữ*biểu thị số lần từ tìm kiếm xuất hiện trong trường của tài liệu hiện tại. Nếu chúng ta tìm kiếm một từ trong trường tiêu đề, số lần từ đó xuất hiện được biểu thị bằng biến tần suất thuật ngữ. Tần suất càng cao, điểm càng cao.

Ví dụ, chúng ta đang tìm kiếm từ Java trong trường tiêu đề trên ba tài liệu. Khi lập chỉ mục, chúng ta đã tạo chỉ mục đảo ngược với thông tin tương tự: từ, số lần từ đó xuất hiện trong trường đó (trong một tài liệu) và ID tài liệu. Chúng ta có thể tạo một bảng với dữ liệu này, như bảng 3.6 cho thấy.

**Bảng 3.6 Tần suất thuật ngữ cho một từ khóa tìm kiếm**

|  |  |  |
| --- | --- | --- |
| **Tiêu đề** | **Tính thường xuyên** | **ID tài liệu** |
| *Làm chủ Java:* | 3 | 25 |
| *Học Java Core và Java Enterprise với các ví dụ* |  |  |
| *Java hiệu quả* | 1 | 13 |
| *Đầu tiên Java* | 1 | 39 |

*Java*xuất hiện ba lần trong tài liệu có ID 25 và một lần trong hai tài liệu khác. Vì từ tìm kiếm xuất hiện thường xuyên hơn trong tài liệu đầu tiên (ID 25), nên hợp lý khi coi tài liệu đó là tài liệu yêu thích của chúng tôi. Hãy nhớ rằng, tần suất càng cao thì mức độ liên quan càng lớn.

Mặc dù con số này có vẻ là dấu hiệu khá tốt về tài liệu có liên quan nhất trong kết quả tìm kiếm của chúng tôi, nhưng thường là không đủ. Một yếu tố khác, tần suất tài liệu ngược, khi kết hợp với TF, tạo ra điểm số được cải thiện.

Số lần từ tìm kiếm xuất hiện trên toàn bộ tập hợp tài liệu (tức là trên toàn bộ chỉ mục) là tần suất tài liệu. Nếu tần suất tài liệu của một từ cao, chúng ta có thể suy ra rằng từ tìm kiếm là phổ biến trên toàn bộ chỉ mục. Nếu từ xuất hiện nhiều lần trên tất cả các tài liệu trong một chỉ mục, thì đó là một thuật ngữ phổ biến và do đó, không liên quan lắm.

Các từ xuất hiện thường xuyên không có ý nghĩa: các từ như a, an, the, it, v.v. là phổ biến trong ngôn ngữ tự nhiên và do đó có thể bị bỏ qua. Đảo ngược tần suất tài liệu (tần suất tài liệu nghịch đảo) cung cấp ý nghĩa cao hơn cho các từ không phổ biến trên toàn bộ chỉ mục. Do đó, tần suất tài liệu càng cao thì mức độ liên quan càng thấp và ngược lại. Bảng 3.7 cho thấy mối quan hệ giữa tần suất từ ​​và mức độ liên quan.

***3.4 Sự liên quan* 93**

**Bảng 3.7 Mối quan hệ giữa tần suất từ ​​và mức độ liên quan**

|  |  |
| --- | --- |
| **Tần suất từ** | **Sự liên quan** |
| Tần suất cao hơn  Tần suất tài liệu cao hơn | Sự liên quan cao hơn  Độ liên quan thấp hơn |

**Dừng từ**

Những từ như*cái, một, nó, một, nhưng, nếu, vì,*Và*Và*, được gọi là*dừng từ*và có thể được loại bỏ bằng cách sử dụng plugin dừng bộ lọc. Mặc địnhtiêu chuẩnmáy phân tích không cótừ dừngtham số được kích hoạt (từ dừngbộ lọc được thiết lập thành\_không có\_theo mặc định), vì vậynhững từ này được phân tích. Tuy nhiên, nếu yêu cầu của chúng ta là bỏ qua những từ này, chúng ta có thể kích hoạt bộ lọc dừng từ bằng cách thêm tham sốtừ dừngđặt thành

\_Tiếng Anh\_, như được hiển thị ở đây:

PUT index\_with\_stopwords

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "standard\_with\_stopwords\_enabled": {

"loại": "chuẩn",

**"từ dừng": "\_tiếng Anh\_"**

}

}

}

}

}

Chúng ta sẽ tìm hiểu về cách tùy chỉnh trình phân tích trong chương 7.

Cho đến phiên bản 5.0, Elasticsearch đã sử dụng hàm tương tự TF-IDF để tính điểm và xếp hạng kết quả. Hàm TF-IDF đã bị loại bỏ để chuyển sang hàm BM25. Thuật toán TF-IDF không xem xét độ dài của trường, làm lệch điểm liên quan. Ví dụ, bạn nghĩ tài liệu nào trong số các tài liệu này liên quan hơn đến tiêu chí tìm kiếm?

* Một trường có 100 từ, bao gồm 5 lần xuất hiện của một từ tìm kiếm
* Một trường có 10 từ, bao gồm 3 lần xuất hiện của từ tìm kiếm

Về mặt logic, có thể thấy rõ ràng rằng tài liệu thứ hai có liên quan nhất vì nó có nhiều từ tìm kiếm hơn với độ dài ngắn hơn. Elasticsearch đã cải thiện thuật toán tương tự của mình bằng cách tăng cường TF-IDF với một tham số bổ sung: độ dài trường.

Chuẩn độ dài trường cung cấp điểm dựa trên độ dài của trường: từ tìm kiếm xuất hiện nhiều lần trong một trường ngắn có liên quan hơn. Ví dụ, từ Java xuất hiện một lần trong một bản tóm tắt dài có thể không chỉ ra kết quả hữu ích. Trên

**94 CPHẦN3*Ngành kiến ​​​​trúc***

Mặt khác, như thể hiện trong bảng 3.8, cùng một từ xuất hiện hai lần hoặc nhiều hơn trong trường tiêu đề (với ít từ hơn) cho biết cuốn sách nói về ngôn ngữ lập trình Java.

**Bảng 3.8 So sánh các lĩnh vực khác nhau để thu thập sự tương đồng**

|  |  |  |  |
| --- | --- | --- | --- |
| **Từ** | **Chiều dài trường** | **Tính thường xuyên** | **Liên quan?** |
| *Java* | Tóm tắt trường có độ dài 100 từ | 1 | KHÔNG |
| *Java* | Trường tiêu đề có độ dài 5 từ | 2 | Đúng |

Trong hầu hết các trường hợp, thuật toán BM25 là đủ. Tuy nhiên, nếu chúng ta cần hoán đổi BM25 với một thuật toán khác, chúng ta có thể thực hiện bằng cách cấu hình nó bằng API lập chỉ mục. Hãy cùng xem xét cơ chế cấu hình thuật toán khi cần.

**CHÌNH DẠNG****THUẬT TOÁN TƯƠNG TỰ**

Elasticsearch cho phép chúng ta cắm thêm các thuật toán tương tự khác nếu BM25 mặc định không phù hợp với yêu cầu của chúng ta. Có hai thuật toán tương tự được cung cấp sẵnkhông cần tùy chỉnh thêm: BM25 và boolean. Chúng ta có thể thiết lập thuật toán tương tự cho từng trường khi tạo định nghĩa lược đồ bằng API cài đặt chỉ mục, như thể hiện trong hình 3.20.

**GHI CHÚ**Làm việc với các thuật toán tương tự là một chủ đề nâng cao. Mặc dù tôi khuyên bạn nên đọc phần này, bạn có thể bỏ qua và xem lại khi bạn muốn biết thêm.

PUT index\_with\_different\_similarities

{

"ánh xạ": { "thuộc tính": {

"tiêu đề":{

"loại": "văn bản",

**"tương tự": "BM25"**

},

"tác giả":{ "loại": "văn bản",

**"tương tự": "boolean"**

}

**Tạo một chỉ mục với hai**

**cánh đồng**

**Cáctiêu đềtrường được xác định bằng một**

**BM25(mặc định) sự tương đồng rõ ràng.**

**Cáctác giảtrường được xác định bằng mộtBooleanhàm tương tự.**

}

}

}

**Hình 3.20 Thiết lập các trường có chức năng tương tự khác nhau**

Trong hình, một chỉ sốindex\_with\_different\_similaritiesđang được phát triển với một lược đồ có hai trường:tiêu đềVàtác giả. Điểm quan trọng là các chi tiết cụ thể

* 1. ***Sự liên quan* 95**

sự kết hợp của hai thuật toán khác nhau được gắn vào hai trường này một cách độc lập: tiêu đề được liên kết với thuật toán BM25, trong khi văn bản được thiết lập bằng boolean.

Mỗi hàm tương tự có các tham số bổ sung và chúng ta có thể thay đổi chúng để phản ánh kết quả tìm kiếm chính xác. Ví dụ, mặc dù hàm BM25 được đặt theo mặc định với các tham số tối ưu, chúng ta có thể dễ dàng sửa đổi hàm bằng API cài đặt chỉ mục. Chúng ta có thể thay đổi hai tham số trong BM25 nếu cần: k1 và b, được mô tả trong bảng 3.9.

**Bảng 3.9 Các tham số hàm tương tự BM25 có sẵn**

|  |  |  |
| --- | --- | --- |
| **Tài sản** | **Giá trị mặc định** | **Sự miêu tả** |
| k1 | 1.2 | Thuật ngữ phi tuyến tínhtần số bão hòa biến |
| b | 0,75 | Hệ số chuẩn hóa TF dựa trên độ dài của tài liệu |

Hãy xem một ví dụ. Hình 3.21 hiển thị một chỉ mục có hàm tương tự tùy chỉnh, trong đó hàm BM25 cốt lõi được sửa đổi bằng các thiết lập riêng của chúng tôi cho k1 và b.

ĐẶT my\_bm25\_index

{

"cài đặt": {

"chỉ số":{ "tương tự":{

"custom\_BM25":{ "loại":"BM25",

"k1":"1.1",

"b": "0,85"

**Cấu hình chỉ mục với các tham số BM25 cụ thể**

**Tạo một hàm tương tự tùy chỉnh với thuật toán BM25 đã sửa đổi**

}

} **Đặtk1Vàbgiá trị**

} **dựa trên yêu cầu của chúng tôi**

}

}

**Hình 3.21 Thiết lập các tham số tùy chỉnh trên hàm tương tự BM25**

Ở đây, chúng ta đang tạo một kiểu tương tự tùy chỉnh—một phiên bản tinh chỉnh của BM25, có thể được sử dụng lại ở nơi khác. (Nó giống như một hàm kiểu dữ liệu, được xác định trước và sẵn sàng để gắn vào các thuộc tính.) Sau khi tạo xong hàm tương tự này, chúng ta có thể sử dụng nó khi thiết lập một trường, như thể hiện trong hình 3.22.

**96 CPHẦN3*Ngành kiến ​​​​trúc***

PUT sách/\_bản đồ

{

"thuộc tính":{ "tóm tắt":{

"loại":"văn bản", "tương tự":"custom\_BM25"

}

}

}

**Tạo một chỉ mục với các trường và kiểu của chúng**

**CácTóm tắttrường hiện được tạo bằng thuật toán tương tự BM25 đã sửa đổi.**

**Hình 3.22 Tạo một trường trong chỉ mục vớimột hàm tương tự BM25 tùy chỉnh**

Chúng tôi tạo một định nghĩa ánh xạ, gán hàm tương tự tùy chỉnh (custom\_BM25) của chúng tôi cho một trường tóm tắt. Khi xếp hạng kết quả dựa trên trường này, Elasticsearch sẽ xem xét hàm tương tự tùy chỉnh được cung cấp để áp dụng điểm số.

**Thuật toán tương tự là quái vật**

Truy xuất thông tin là một chủ đề rộng lớn và phức tạp. Các thuật toán liên quan đến việc chấm điểmvà kết quả xếp hạng là nâng cao và phức tạp. Trong khi Elasticsearch cung cấp một số thuật toán tương tự như plug-and-play, bạn có thể cần hiểu sâu hơn khi làm việc với chúng. Sau khi bạn đã tinh chỉnh các tham số cấu hình của các hàm chấm điểm này, hãy đảm bảo rằng bạn đã kiểm tra và thử mọi kết hợp có thể.

Bạn có thể tự hỏi làm thế nào Elasticsearch có thể truy xuất tài liệu trong một phần giây. Làm thế nào nó biết được tài liệu nằm ở đâu trong nhiều phân đoạn này? Chìa khóa là thuật toán định tuyến, được thảo luận tiếp theo.

#### Thuật toán định tuyến

Mỗi tài liệu đều có một nơi lưu trữ cố định: nghĩa là nó phải thuộc về một phân đoạn chính cụ thể. Elasticsearch sử dụng thuật toán định tuyến để phân phối tài liệu đến phân đoạn cơ sở khi lập chỉ mục.

*Lộ trình*là một quá trình phân bổ một nơi lưu trữ cho một tài liệu vào một mảnh nhất định, với mỗi tài liệu được lưu trữ trong một và chỉ một mảnh chính. Việc truy xuất cùng một tài liệu rất dễ dàng, vì cùng một hàm định tuyến được sử dụng để tìm mảnh mà tài liệu đó thuộc về.

Thuật toán định tuyến là một công thức đơn giản mà Elasticsearch sử dụng để suy ra phân đoạn cho một tài liệu trong quá trình lập chỉ mục hoặc tìm kiếm:

shard\_number = hash(id) % số\_shards

Đầu ra của hàm định tuyến là số phân đoạn. Nó được tính bằng cách băm ID của tài liệu và tìm phần còn lại của băm khi chia (sử dụng toán tử mod- ulo) cho số phân đoạn. Hàm băm mong đợi một ID duy nhất, thường là ID tài liệu hoặc ID tùy chỉnh do người dùng cung cấp. Các tài liệu được phân phối đều, do đó không có khả năng một trong các phân đoạn bị quá tải.

* 1. ***Tỷ lệ* 97**

Lưu ý rằng công thức phụ thuộc trực tiếp vào biến number\_of\_shards. Điều đó có nghĩa là chúng ta không thể thay đổi số lượng shards sau khi một chỉ mục được tạo. Nếu chúng ta có thể thay đổi các thiết lập (ví dụ, thay đổi số lượng shards từ hai thành bốn), thì hàm định tuyến sẽ bị hỏng đối với các bản ghi hiện có và dữ liệu sẽ không được tìm thấy. Đây là lý do tại sao Elasticsearch không cho phép chúng ta thay đổi các shards sau khi một chỉ mục được thiết lập.

Nếu chúng ta không lường trước được sự tăng trưởng dữ liệu và các phân đoạn bị cạn kiệt do dữ liệu tăng đột biến thì sao? Không phải là mất hết. Có một cách giải quyết: lập chỉ mục lại dữ liệu của chúng ta. Việc lập chỉ mục lại hiệu quả tạo ra một chỉ mục mới với các thiết lập phù hợp và sao chép dữ liệu từ chỉ mục cũ sang chỉ mục mới.

**Bản sao có thể được thay đổi trên một chỉ mục hoạt động**

Trong khi số lượng mảnh không thể thay đổi khi chỉ mục đang hoạt động, chúng ta có thể thay đổi số lượng bản sao nếu cần. Hãy nhớ rằng, hàm định tuyến là hàm của số lượng mảnh chính, không phải bản sao. Nếu chúng ta cần thay đổi số mảnh, chúng ta phải đóng các chỉ mục (các chỉ mục đóng chặn tất cả các hoạt động đọc và ghi), thay đổi số mảnh và mở lại các chỉ mục. Ngoài ra, chúng ta có thể tạo một chỉ mục mới với một tập hợp các mảnh mới và lập chỉ mục lại dữ liệu từ chỉ mục cũ sang chỉ mục mới.

Một trong những mục tiêu chính của Elasticsearch là khả năng mở rộng của engine. Trong phần tiếp theo, chúng ta sẽ xem xét khả năng mở rộng của Elasticsearch ở cấp độ cao: cách mở rộng, cách mở rộng theo chiều dọc và chiều ngang hoạt động như thế nào và quy trình lập chỉ mục lại. Chúng ta không đi sâu vào các giải pháp mở rộng ở đây vì chúng ta sẽ quay lại thảo luận này trong chương 14.

#### Tỷ lệ

Khi Shay Banon và nhóm của ông viết lại Elasticsearch từ đầu, một trong những mục tiêu của họ là đảm bảo máy chủ có thể mở rộng quy mô một cách dễ dàng. Nghĩa là, nếu dữ liệu tăng hoặc tải truy vấn tăng, việc thêm các nút bổ sung sẽ giải quyết được vấn đề. Tất nhiên, vẫn còn những giải pháp khác, chẳng hạn như mở rộng theo chiều dọc, điều chỉnh hiệu suất, v.v. Có hai trường phái chính về mở rộng quy mô, dựa trên nhu cầu: mở rộng theo chiều ngang và mở rộng theo chiều dọc. Elasticsearch hỗ trợ cả hai.

* + 1. Mở rộng quy mô (mở rộng theo chiều dọc)

Trong kịch bản mở rộng theo chiều dọc, chúng tôi không mua thêm VM từ nhà cung cấp đám mây của mình mà thay vào đó là thêm tài nguyên điện toán, chẳng hạn như bộ nhớ bổ sung, CPU và I/O, vào các máy hiện có. Ví dụ, chúng tôi có thể tăng lõi CPU, bộ nhớ gấp đôi, v.v.

Ngoài ra còn có một cách khác để tăng sức mạnh của cụm. Vì bây giờ chúng ta có đủ chỗ, chúng ta có thể cài đặt thêm các nút trên máy, do đó tạo ra nhiều nút trong một máy duy nhất (fat).

Hãy nhớ rằng việc mở rộng quy mô đòi hỏi phải tắt cụm, do đó ứng dụng của chúng ta có thể gặp thời gian chết trừ khi chúng ta dựa vào mô hình khôi phục sau thảm họa (DR) truyền thống. Điều này đưa hệ thống thứ cấp hoặc sao lưu của chúng ta, phục vụ khách hàng trong khi hệ thống chính đang được bảo trì.

**98 CPHẦN3*Ngành kiến ​​​​trúc***

Tuy nhiên, kiểu mở rộng này có nguy cơ tiềm ẩn. Nếu toàn bộ máy bị sập trong trường hợp khẩn cấp hoặc do lỗi phần cứng, dữ liệu có thể bị mất vì tất cả các nút lưu trữ dữ liệu đều nằm trên cùng một máy. Tất nhiên, chúng tôi có bản sao lưu, do đó có thể khôi phục, nhưng rất tốn kém. Các bản sao lưu của chúng tôi được lưu trữ trên cùng một máy, mặc dù là các nút khác nhau, điều này gây ra rắc rối.

* + 1. Thu nhỏ (thu nhỏ theo chiều ngang)

Ngoài ra, chúng ta có thể mở rộng môi trường của mình (theo chiều ngang). Thay vì gắn thêm RAM và bộ nhớ vào các máy hiện có, chúng ta có thể thêm một số máy mới (có thể là VM có ít năng lượng tài nguyên hơn các máy lớn được sử dụng để mở rộng theo chiều dọc) để tạo thành một trang trại mở rộng theo chiều ngang.

Các VM mới này được khởi động như các nút mới, do đó tham gia vào cụm Elasticsearch hiện có. Ngay khi các nút mới tham gia vào cụm, Elasticsearch, là một công cụ kiến ​​trúc phân tán, sẽ phân phối dữ liệu ngay lập tức cho chúng. Việc tạo VM rất dễ dàng, đặc biệt là khi sử dụng các công cụ cơ sở hạ tầng dưới dạng mã (IaC) hiện đại như Terraform, Ansible, Chef, v.v., vì vậy cách tiếp cận này có xu hướng được nhiều nhóm kỹ sư DevOps ưa chuộng.

Chúng ta hãy kết thúc chương này tại đây. Đây là hướng dẫn để bạn hiểu được những điều cơ bản của Elasticsearch: các bộ phận chuyển động, khối cấp thấp và khái niệm tìm kiếm. Chúng tôi sẽ nói thêm về các khái niệm và nguyên tắc cơ bản, cùng với các ví dụ, trong hai chương tiếp theo, vì vậy hãy chú ý theo dõi.

#### Bản tóm tắt

* + - * Elasticsearch mong đợi dữ liệu được đưa vào để lập chỉ mục. Nguồn dữ liệu có thể bao gồm các tệp đơn giản, cơ sở dữ liệu, luồng trực tiếp, Twitter, v.v.
      * Trong quá trình lập chỉ mục, dữ liệu trải qua giai đoạn phân tích nghiêm ngặt, trong đó các cấu trúc dữ liệu nâng cao như chỉ mục đảo ngược được tạo ra.
      * Dữ liệu được truy xuất hoặc tìm kiếm thông qua API tìm kiếm (cùng với API tài liệu để truy xuất từng tài liệu).
      * Dữ liệu đến phải được gói gọn trong một tài liệu JSON. Vì tài liệu JSON là thực thể lưu trữ dữ liệu cơ bản nên nó được lưu trữ trong các phân đoạn và bản sao.
      * Các mảnh và bản sao là các phiên bản Apache Lucene có trách nhiệm lưu trữ, truy xuất và phân phối tài liệu.
      * Khi chúng ta khởi động ứng dụng Elasticsearch, nó khởi động như một ứng dụng một nút, một cụm. Việc thêm các nút sẽ mở rộng cụm, biến nó thành cụm nhiều nút.
      * Để truy xuất thông tin và lưu trữ dữ liệu nhanh hơn, Elasticsearch cung cấp các cấu trúc dữ liệu tiên tiến như chỉ mục đảo ngược cho dữ liệu có cấu trúc (như thông tin văn bản) và cây BKD cho dữ liệu phi cấu trúc (như ngày tháng và số).
      * Độ liên quan là điểm số dấu phẩy động dương được đính kèm vào kết quả tài liệu được truy xuất. Nó xác định mức độ phù hợp của tài liệu với tiêu chí tìm kiếm.

***Bản tóm tắt* 99**

* + - * + Elasticsearch sử dụng thuật toán so sánh độ liên quan hoặc độ tương đồng Okapi Best Match (BM) 25, một phiên bản nâng cao của thuật toán so sánh độ tương đồng tần suất thuật ngữ/tần suất tài liệu nghịch đảo.
        + Bạn có thể mở rộng Elasticsearch lên hoặc ra, dựa trên yêu cầu và tài nguyên có sẵn của bạn. Mở rộng quy mô sẽ tăng cường các máy hiện có (thêm bộ nhớ, CPU, RAM, v.v.), trong khi mở rộng quy mô sẽ tạo ra nhiều máy ảo (VM) hơn, cho phép chúng tham gia cụm và chia sẻ tải.

*Bản đồ*

***Chương này bao gồm***

* Kiểu dữ liệu trường
* Ngầm địnhvà lập bản đồ rõ ràng
* Các kiểu dữ liệu cốt lõi
* Kiểu dữ liệu nâng cao
* API để tạo/truy cập ánh xạ

Dữ liệu giống như cầu vồng—nó có đủ loại “màu sắc”. Dữ liệu kinh doanh có nhiều hình dạng và dạng thức khác nhau được biểu diễn dưới dạng thông tin văn bản, ngày tháng, số, đối tượng bên trong, Boolean, vị trí địa lý, địa chỉ IP, v.v. Trong Elasticsearch, chúng tôi lập mô hình và lập chỉ mục dữ liệu dưới dạng tài liệu JSON, trong đó mỗi tài liệu bao gồm một số trường và mỗi trường chứa một loại dữ liệu nhất định. Ví dụ, tài liệu phim có thể bao gồm tiêu đề và tóm tắt được biểu diễn dưới dạng dữ liệu văn bản, ngày phát hành dưới dạng ngày và tổng thu nhập dưới dạng dữ liệu dấu phẩy động.

Trong các chương trước, khi chúng ta lập chỉ mục các tài liệu mẫu, chúng ta không bận tâm đến các kiểu dữ liệu của các trường. Elasticsearch đã suy ra các kiểu này một cách ngầm định bằng cách xem xét từng trường và loại thông tin trong đó. Elasticsearch đã tạo ra một lược đồ mà không cần chúng ta phải thực hiện bất kỳ công việc nào trước, không giống như trong cơ sở dữ liệu quan hệ.

**100**

***4.1 Tổng quan vềlập bản đồ* 101**

Bắt buộc phải có lược đồ bảng được định nghĩa và phát triển trong cơ sở dữ liệu trước khi truy xuất hoặc lưu trữ dữ liệu. Nhưng chúng ta có thể chuẩn bị Elasticsearch bằng các tài liệu mà không cần định nghĩa lược đồ cho mô hình dữ liệu của mình. Tính năng không có lược đồ này giúp các nhà phát triển bắt đầu và chạy hệ thống ngay từ ngày đầu tiên. Tuy nhiên, cách làm tốt nhất là phát triển lược đồ ngay từ đầu thay vì để Elasticsearch định nghĩa lược đồ cho chúng ta, trừ khi các yêu cầu của chúng ta không cần lược đồ.

Elasticsearch mong đợi chúng ta cung cấp manh mối về cách nó nên xử lý một trường khi lập chỉ mục dữ liệu. Những manh mối này được chúng ta cung cấp dưới dạng định nghĩa lược đồ trong khi tạo chỉ mục hoặc được công cụ ngầm suy ra nếu chúng ta cho phép nó làm như vậy. Quá trình tạo định nghĩa lược đồ này được gọi là ánh xạ.

Mapping cho phép Elasticsearch hiểu được hình dạng của dữ liệu để có thể áp dụng một tập hợp các quy tắc được xác định trước trên các trường trước khi lập chỉ mục cho chúng. Elasticsearch cũng tham khảo hướng dẫn về các quy tắc mapping để áp dụng các quy tắc toàn văn trên các trường văn bản. Các trường có cấu trúc (giá trị chính xác, như số hoặc ngày) có một tập hợp các hướng dẫn riêng cho phép chúng trở thành một phần của các hàm tổng hợp và sắp xếp và lọc, ngoài việc có sẵn cho các tìm kiếm chung.

Trong chương này, chúng tôi thiết lập bối cảnh để sử dụng lược đồ ánh xạ, khám phá quy trình ánh xạ-ping và làm việc với các kiểu dữ liệu, xem xét cách xác định chúng bằng API ánh xạ-ping. Dữ liệu được lập chỉ mục cho Elasticsearch có hình dạng và dạng thức xác định. Việc định hình dữ liệu một cách tỉ mỉ cho phép Elasticsearch thực hiện phân tích hoàn hảo, cung cấp cho người dùng cuối kết quả chính xác. Chương này thảo luận về cách xử lý dữ liệu trong Elasticsearch và cách lược đồ ánh xạ giúp chúng ta tránh được các trở ngại và có được các tìm kiếm chính xác.

**Chạy 100m hay vượt rào 400m?**

Chương này đề cập đến hàng chục ví dụ thực hành xung quanh cả cốt lõi và nâng caokiểu dữ liệu. Mặc dù tôi khuyên bạn nên đọc về chúng theo thứ tự đã cho, nhưng nếu bạn mới bắt đầu sử dụng Elasticsearch và muốn tập trung vào các thành phần dành cho người mới bắt đầu, bạn có thể bỏ qua phần 4.6 và xem lại khi bạn tự tin hơn và muốn tìm hiểu thêm.

Nếu tất cả những gì bạn muốn là chạy 100 m thay vì vượt rào 400 m, hãy đọc chương này cho đến phần dữ liệu cốt lõi (phần 4.4) rồi chuyển sang chương tiếp theo.

#### 4.1 Tổng quan về lập bản đồ

Mapping là quá trình xác định và phát triển định nghĩa lược đồ biểu diễn các trường dữ liệu của tài liệu và các kiểu dữ liệu liên quan của chúng. Mapping cho công cụ biết hình dạng và dạng thức của dữ liệu đang được lập chỉ mục. Vì Elasticsearch là cơ sở dữ liệu hướng tài liệu, nên nó mong đợi một định nghĩa ánh xạ duy nhất cho mỗi chỉ mục. Mỗi trường được xử lý theo quy tắc ánh xạ. Ví dụ: trường chuỗi được xử lý như trường văn bản, trường số được lưu trữ dưới dạng số nguyên, trường ngày được lập chỉ mục như ngày để cho phép các hoạt động liên quan đến ngày, v.v. Ánh xạ chính xác và không có lỗi cho phép Elasticsearch phân tích dữ liệu một cách hoàn hảo, hỗ trợ các chức năng liên quan đến tìm kiếm, sắp xếp, lọc và tổng hợp.

**102 CPHẦN4*Bản đồ***

**GHI CHÚ**Để đơn giản hóa các bài tập mã hóa, tôi đã tạo một tệp ch04\_mapping.txt trong thư mục kibana\_scripts ở gốc kho lưu trữ của cuốn sách ([http://](http://mng.bz/OxXo) [mng.bz/OxXo](http://mng.bz/OxXo)). Sao chép nội dung của tệp này vào Kibana theo nguyên trạng. Bạn có thể thực hiện các ví dụ bằng cách thực thi từng đoạn mã riêng lẻ trong khi làm theo nội dung của chương.

* + 1. Định nghĩa bản đồ

Mỗi tài liệu bao gồm một tập hợp các trường đại diện cho dữ liệu kinh doanh và mỗi trường có một hoặc nhiều kiểu dữ liệu cụ thể liên kết với nó. Định nghĩa ánh xạ là lược đồ của các trường và kiểu dữ liệu của chúng trong một tài liệu. Mỗi trường được lưu trữ và lập chỉ mục theo một cách cụ thể dựa trên kiểu dữ liệu. Điều này giúp Elasticsearch hỗ trợ nhiều truy vấn tìm kiếm như toàn văn, mờ, thuật ngữ và địa lý.

Trong ngôn ngữ lập trình, chúng ta biểu diễn dữ liệu bằng các kiểu dữ liệu cụ thể (chuỗi, ngày, số, đối tượng, v.v.). Một thông lệ phổ biến là cho hệ thống biết các kiểu biến trong quá trình biên dịch. Trong cơ sở dữ liệu quan hệ, chúng ta định nghĩa một lược đồ bảng với các định nghĩa trường thích hợp để lưu trữ các bản ghi và lược đồ phải tồn tại trước khi chúng ta bắt đầu lưu trữ trong cơ sở dữ liệu.

Tìm kiếm đàn hồihiểu các kiểu dữ liệu của các trường trong khi lập chỉ mục tài liệu và lưu trữ các trường vào các cấu trúc dữ liệu phù hợp (ví dụ: chỉ mục đảo ngược cho các trường văn bản và cây khối k chiều (BKD) cho các trường số) để truy xuất dữ liệu. Dữ liệu được lập chỉ mục với các kiểu dữ liệu được hình thành chính xác dẫn đến kết quả tìm kiếm chính xác và giúp sắp xếp và tổng hợp dữ liệu.

Hình 4.1 cho thấy cấu trúc của lược đồ ánh xạ cho một chỉ mục. Như bạn có thể thấy, chỉ mục bao gồm một đối tượng ánh xạ được tạo thành từ các trường riêng lẻ và các loại của chúngđược bao gồm trong một đối tượng thuộc tính.

**Mọichỉ số bao gồm mộtlập bản đồsự vật.**

của cải

lập bản đồ

Mục lục

**Mỗi trường được xác định bằng một loại cụ thể và các tham số bổ sung.**

**Cácphimchỉ mục và các trường, kiểu dữ liệu và tham số của nó**

cánh đồng 3

<kiểu dữ liệu>

<tham số>

cánh đồng 2

<kiểu dữ liệu>

<tham số>

cánh đồng 1

<kiểu dữ liệu>

<tham số>

**Các trường và thông tin chi tiết của chúng**

**được bao bọc trong một**

**của cảisự vật.**

**Mỗi trường có thể có**

**nhiều loại được định nghĩa bởi mộtcánh đồngvật bên trong.**

cánh đồng

(nhiều lĩnh vực)

cánh đồng 4

<kiểu dữ liệu>

<tham số>

**Các loại khác được bao gồm trongcánh đồngsự vật.**

**Hình 4.1 Giải phẫu của một lược đồ ánh xạ**

* 1. ***Tổng quan vềlập bản đồ* 103**
     1. Lập chỉ mục tài liệu lần đầu tiên

Hãy cùng tìm hiểu điều gì sẽ xảy ra nếu chúng ta lập chỉ mục một tài liệu mà không tạo lược đồmặt trước. Giả sử chúng ta muốn lập chỉ mục cho một tài liệu phim, như được hiển thị trong danh sách sau.

**Liệt kê 4.1 Lập chỉ mục một tài liệu có ID 1 vàophimchỉ số**

PUT phim/\_doc/1

{

"title":"Bố già","xếp hạng":4.9,

**Tiêu đề của**

**cáibộ phim Đánh giá đã đưa ra**

**đến bộ phim**

"năm phát hành":"1972/08/01"

}

**Năm phát hành của phim (lưu ý định dạng ngày)**

Đây là tài liệu đầu tiên được gửi đến Elasticsearch để được lập chỉ mục. Hãy nhớ rằng, chúng tôi đã không tạo chỉ mục (phim) hoặc lược đồ cho dữ liệu tài liệu này trước khi nhập tài liệu. Sau đây là những gì xảy ra khi tài liệu này đến công cụ:

**1** Một chỉ mục mới (phim) sẽ được tạo tự động theo cài đặt mặc định.

**2**Một lược đồ mới được tạo ra chophimchỉ mục sử dụng các kiểu dữ liệu (sẽ được thảo luận ngắn gọn) được suy ra từ các trường của tài liệu này. Ví dụ,tiêu đềđược thiết lập thànhchữVàtừ khóacác loại,xếp hạngĐẾNtrôi nổi, Vànăm phát hànhđến mộtngàykiểu.

**3** Tài liệu được lập chỉ mục và lưu trữ trong kho dữ liệu Elasticsearch.

**4**Các tài liệu tiếp theo được lập chỉ mục mà không cần trải qua các bước trước đó vì Elasticsearch sẽ tham khảo lược đồ mới tạo để lập chỉ mục thêm.

Elasticsearch thực hiện một số công việc cơ bản đằng sau hậu trường khi thực hiện các bước này để tạo định nghĩa lược đồ với tên trường và thông tin bổ sung. Chúng ta có thể sử dụng API ánh xạ để lấy động định nghĩa lược đồ mà Elasticsearch đã tạo. Phản hồi cho lệnh GET được hiển thị trong hình 4.2.

...

"của cải" : {

"xếp hạng" : {"loại" : "float"}, "năm phát hành" : {

"loại" : "ngày",

**Do đóxếp hạnggiá trị 4,9, một số dấu phẩy động, kiểu được xác định làtrôi nổi.**

"định dạng": "năm/tháng/ngày HH:mm:ss||năm/tháng/ngày||epoch\_millis"

NHẬN phim/\_mapping

},

"tiêu đề" : {

"loại": "văn bản",

**Các**



**tiêu đềlà văn bản**

**Như lànăm phát hànhgiá trị ở định dạng năm (định dạng ISO), loại được xác định làngày.**

"các trường" : {

"từ khóa" : {

"type" : "từ khóa", "ignore\_above" : 256

**thông tin, vì vậy mộtchữ**

**kiểu dữ liệu được chỉ định.**

} **Cáccánh đồngđối tượng chỉ ra**

} **kiểu dữ liệu thứ hai được xác định cho cùng mộttiêu đềtrường (nhiều dữ liệu**

} **các loại).**

}

**Hình 4.2Bản đồ chỉ mục phim được lấy từgiá trị tài liệu**

**104 CPHẦN4*Bản đồ***

Elasticsearch sử dụng ánh xạ động để suy ra các kiểu dữ liệu của các trường khi một tài liệu được lập chỉ mục lần đầu tiên bằng cách xem các giá trị trường và suy ra các giá trị này.types. Mỗi trường có một kiểu dữ liệu cụ thể được xác định: trường rating được khai báo là float, release\_year là kiểu date, v.v. Elasticsearch xác định động rằng kiểu của trường title là text dựa trên giá trị chuỗi của nó ("Godfather"). Vì trường được đóng dấu là kiểu text, nên tất cả các truy vấn liên quan đến toàn văn bản có thể được thực hiện trên trường này.

Một trường riêng lẻ cũng có thể bao gồm các trường khác, đại diện cho nhiềukiểu dữ liệu. Ngoài việc tạo trường tiêu đề dưới dạng kiểu văn bản, Elasticsearch còn làm thêm một việc nữa: sử dụng đối tượng fields, nó đã tạo thêm một kiểu từ khóa cho trường tiêu đề, do đó biến title thành trường có nhiều kiểu. Tuy nhiên, phải truy cập bằng title.keyword.

Các trường đa kiểu có thể được liên kết với nhiều kiểu dữ liệu. Trong ví dụ của chúng tôi, bằngmặc định, trường tiêu đề được ánh xạ tới văn bản cũng như loại từ khóa. Các trường từ khóa được sử dụng để tìm kiếm giá trị chính xác. Các trường có loại dữ liệu này được giữ nguyên và không trải qua giai đoạn phân tích: chúng không được mã hóa, đồng nghĩa hóa hoặc bắt nguồn. Xem phần 4.7 để biết thêm thông tin về các trường có nhiều loại.

**Phân tích các trường từ khóa**

Các trường được khai báo làtừ khóakiểu dữ liệu sử dụng một trình phân tích đặc biệt được gọi làKhông(bộ phân tích không hoạt động), vàtừ khóadữ liệu không bị ảnh hưởng trong quá trình lập chỉ mục.Công cụ phân tích từ khóa này đưa ra toàn bộ trường dưới dạng một mã thông báo lớn.

Theo mặc định,từ khóacác trường không được chuẩn hóa, vìchất bình thường hóatài sản trêntừ khóaloại được thiết lập thànhvô giá trị. Nhưng chúng ta có thể tùy chỉnh và kích hoạtchất bình thường hóatrêntừ khóakiểu dữ liệu bằng cách thiết lập các bộ lọc nhưTiếng Đức,chữ in hoa, v.v. Làm như vậy cho thấy rằng chúng tôi muốntừ khóatrường để trải qua quá trình chuẩn hóatrước khi lập chỉ mục.

Quá trình mà Elasticsearch suy ra ánh xạ từ các giá trị trường được gọi là ánh xạ động (thông tin chi tiết hơn ở phần tiếp theo). Mặc dù ánh xạ động thông minh và tiện lợi, nhưng hãy lưu ý rằng nó cũng có thể định nghĩa sai lược đồ. Elasticsearch chỉ có thể đi xa đến vậy khi suy ra lược đồ dựa trên các giá trị trường của tài liệu. Nó có thể đưa ra các giả định không chính xác, dẫn đến các lược đồ chỉ mục sai có thể tạo ra kết quả tìm kiếm không chính xác.

Chúng ta đã biết rằng Elasticsearch lấy và xác định kiểu dữ liệu của các trường và tạo sơ đồ cho chúng ta một cách động nếu sơ đồ không tồn tại khi tài liệu đầu tiên được chuyển đến để lập chỉ mục. Trong phần tiếp theo, chúng ta sẽ thảo luận về cách Elasticsearch xác định kiểu và nhược điểm của ánh xạ động.

#### Năng động lập bản đồ

Khi chúng ta lập chỉ mục một tài liệu lần đầu tiên, cả ánh xạ và chỉ mục đều được tạo tự động. Công cụ không phàn nàn nếu chúng ta không cung cấp lược đồ

***4.2 Độnglập bản đồ* 105**

trước tiên—Elasticsearch rất dễ tính trong vấn đề này. Chúng ta có thể lập chỉ mục một tài liệu mà không cần cho công cụ biết bất cứ điều gì về kiểu dữ liệu của các trường. Hãy xem xét một tài liệu phim bao gồm một vài trường, được hiển thị trong hình 4.3.

**Tài liệu JSON**

{

"title":"Bố già",

"ngày phát hành":"1972-14-03", "thời gian chạy":177

}

tiêu đềgiá trị trường là một chuỗi -> kiểu dữ liệu làchữ

ngày phát hànhgiá trị phù hợp Một ngày sợi dây -> dữ liệu kiểu là ngày

thời gian chạygiá trị trường phù hợp với dữ liệu số -> kiểu dữ liệu là float

Suy ra sơ đồ

"tiêu đề" : { "kiểu" : "văn bản" },

"ngày phát hành" : { "loại" "ngày"},

"running\_time" : {"type" : "dài"}

**Định nghĩa bản đồ**

**Hình 4.3 Động lực học lược đồ lập chỉ mục**

Khi đọc các trường và giá trị trong tài liệu này, Elasticsearch sẽ tự động suy ra các kiểu dữ liệu:

* Cáctiêu đềgiá trị trường là một chuỗi, do đó Elasticsearch ánh xạ trường thành mộtchữkiểu dữ liệu.
* Cácngày phát hànhgiá trị trường là một chuỗi, nhưng giá trị khớp với định dạng ngày ISO 8601, do đó nó được ánh xạ tới mộtngàykiểu.
* Cácxếp hạngtrường là một số, vì vậy Elasticsearch ánh xạ nó thành mộttrôi nổikiểu dữ liệu.

Vì chúng ta không tạo bản đồ một cách rõ ràng mà thay vào đó để Elasticsearch suy ra lược đồ một cách nhanh chóng, loại bản đồ này được gọi là bản đồ động. Tính năng này rất tiện lợi trong quá trình phát triển ứng dụng hoặc trong môi trường thử nghiệm.

Elasticsearch biết kiểu dữ liệu của trường rating là kiểu float như thế nào? Về vấn đề đó, Elasticsearch suy ra bất kỳ kiểu nào trong số các kiểu đó như thế nào? Tiếp theo, chúng ta hãy xem xét cơ chế và quy tắc về cách các kiểu được Elasticsearch suy ra.

* + 1. Cơ chế suy luận các loại

Elasticsearch phân tích các giá trị trường và đoán kiểu dữ liệu tương đương dựa trên các giá trị đó. Khi đối tượng JSON đến được phân tích cú pháp, giá trị xếp hạng 4,9, ví dụ, giống với số nguyên dấu phẩy động trong thuật ngữ ngôn ngữ lập trình. "Phỏng đoán" này khá đơn giản; do đó, Elasticsearch đóng dấu trường xếp hạng là trường dấu phẩy động.

**106 CPHẦN4*Bản đồ***

Trong khi việc suy ra một kiểu số có thể dễ dàng, thì kiểu của trường release\_year thì sao? Giá trị trường này được so sánh và khớp với một trong hai định dạng ngày mặc định: yyyy/MM/dd hoặc yyyy-MM-dd. Nếu khớp, nó sẽ được gán một kiểu dữ liệu ngày.

**Định dạng ngày trong quá trình ánh xạ động**

Elasticsearch có thể suy ra rằng một trường là mộtngàyloại nếu các giá trị trong tài liệu JSON được cung cấp theo định dạngnăm-tháng-ngàyhoặcnăm/tháng/ngày,mặc dù định dạng sau không phải là định dạng ngày ISO. Tuy nhiên, tính linh hoạt này chỉ khả dụng cho các trường hợp ánh xạ động.

Nếu chúng ta khai báo một trường làngàynhập rõ ràng (sử dụng ánh xạ rõ ràng—thêm thông tin về điều nàysau đó), trừ khi chúng tôi cung cấp định dạng tùy chỉnh, trường này sẽ tuân thủ theo mặc địnhstrict\_date\_optional\_timeĐịnh dạng này tuân theo định dạng ngày tháng theo tiêu chuẩn ISO:năm-tháng-ngàyhoặcnăm-tháng-ngàyTHH:mm:giây.

Phỏng đoán này đủ cho hầu hết các trường hợp nhưng không đủ nếu dữ liệu hơi lệch khỏi đường dẫn mặc định. Hãy xem xét trường release\_year trong tài liệu phim của chúng tôi. Nếu tài liệu của chúng tôiment có giá trị cho release\_year theo định dạng khác (ví dụ: ddMMyyyy hoặc dd/MM/yyyy), quy tắc ánh xạ động sẽ bị phá vỡ. Elasticsearch coi giá trị đó là trường văn bản chứ không phải trường ngày.

Tương tự như vậy, hãy thử lập chỉ mục một tài liệu khác với trường xếp hạng được đặt thành 4 (chúng tôi dự định cung cấp xếp hạng dưới dạng số thập phân). Elasticsearch xác định rằng trường nàylà một kiểu dữ liệu dài vì giá trị phù hợp với quy tắc kiểu dữ liệu "dài" của nó. Mặc dù chúng ta muốn trường xếp hạng lưu trữ dữ liệu dấu phẩy động, Elasticsearch đã không xác định được kiểu dữ liệu phù hợp. Khi xem xét giá trị (4), nó cho rằng kiểu dữ liệu của trường là dài. Trong cả hai trường hợp, chúng ta đều có một lược đồ với các kiểu dữ liệu không chính xác. Việc có các kiểu dữ liệu không chính xác có thể gây ra sự cố trong ứng dụng, khiến các trường không đủ điều kiện để sắp xếp, lọc và tổng hợp dữ liệu. Nhưng đây là điều tốt nhất mà Elasticsearch có thể làm khi chúng ta sử dụng

tính năng lập bản đồ động.

**Mang dữ liệu của bạn đến—làm chosơ đồ**

Trong khi Elasticsearch đủ thông minh để có thể lấy được thông tin lập bản đồ dựa trên tài liệu của chúng tôiuments, vì vậy chúng ta không cần phải lo lắng về lược đồ, có khả năng mọi thứ sẽ diễn ra không tốt và chúng ta sẽ đưa ra định nghĩa lược đồ không chính xác. Có một lược đồ không tốt là tự chuốc lấy rắc rối.

Bởi vì chúng tôi thường hiểu rõ các miền của mình và biết các mô hình dữ liệu của mìnhTrong và ngoài, lời khuyên của tôi là không nên để Elasticsearch tạo lược đồ, đặc biệt là trong môi trường sản xuất. Thay vào đó, tốt nhất là tạo lược đồ của chúng ta trước, thường là bằng cách tạo chiến lược ánh xạ trên toàn tổ chức bằng cách sử dụng các mẫu ánh xạ (chúng ta sẽ tìm hiểu một số mẫu ánh xạ trong chương 6). Tất nhiên, nếu trường hợp sử dụng của bạn bao gồm dữ liệu không có định dạng chuẩn, ánh xạ động có thể giúp ích rất nhiều.

* 1. ***Năng độnglập bản đồ* 107**

Cho đến nay, chúng ta đã xem xét về ánh xạ động và thấy rằng mặc dù tính năng này rất hấp dẫn nhưng nó vẫn có những hạn chế, như sẽ thảo luận ở phần tiếp theo.

* + 1. Hạn chế của ánh xạ động

Có một số hạn chế khi cho phép Elasticsearch suy ra lược đồ tài liệu. Elasticsearch có thể hiểu sai các giá trị trường tài liệu và suy ra ánh xạ không chính xác làm mất tính đủ điều kiện của trường đối với các khả năng tìm kiếm, sắp xếp và tổng hợp phù hợp. Hãy cùng tìm hiểu cách Elasticsearch xác định các kiểu dữ liệu không chính xác và do đó hỗ trợ xây dựng các quy tắc ánh xạ không chính xác và sai.

**DLẬP MỘT ÁNH XẠ KHÔNG CHÍNH XÁC**

Giả sử chúng ta muốn cung cấp một trường dữ liệu số nhưng dữ liệu sẽ được gói dưới dạng chuỗi (ví dụ: "3.14"—lưu ý dấu ngoặc kép). Thật không may, Elastic-search xử lý dữ liệu như vậy không đúng: trong ví dụ này, nó mong đợi kiểu dữ liệu là văn bản, không phải float hoặc double.

Hãy sửa đổi tài liệu sinh viên mà chúng ta đã xem trong chương 3 và thêm một trường:tuổi của học sinh. Nghe giống như một con số, nhưng chúng tôi đang lập chỉ mục nó như một trường văn bản bằng cách đặt giá trị trong dấu ngoặc kép.

**Liệt kê 4.2 Thêm mộttuổitrường có giá trị văn bản đếnhọc sinhtài liệu**

ĐẶT students\_temp/\_doc/1

{

"tên":"John",

**"tuổi": "12"**

}

**Giá trị của biến age được đặt dưới dạng chuỗi (trong dấu ngoặc kép).**

**Tài liệu đầu tiên**

ĐẶT students\_temp/\_doc/2

{

"tên":"William",

**"tuổi": "14"**

**Tài liệu thứ hai**

}

Lưu ý rằng giá trị của trường age được đặt trong dấu ngoặc kép, do đó Elasticsearch coi trường này là trường văn bản (mặc dù trường này bao gồm số). Sau khi lập chỉ mục cho hai tài liệu này, chúng ta có thể viết truy vấn tìm kiếm để sắp xếp học sinh theo độ tuổi tăng dần hoặc giảm dần (xem hình 4.4). Phản hồi cho biết thao tác này không được phép; hãy cùng xem lý do.

Vì trường tuổi của tài liệu học sinh được lập chỉ mục dưới dạng trường văn bản (theo mặc định, bất kỳ trường chuỗi nào cũng được ánh xạ tới trường văn bản trong quá trình ánh xạ động), Elasticsearch không thể sử dụng trường đó trong các hoạt động sắp xếp. Chức năng sắp xếp không khả dụng trên các trường văn bản theo mặc định.

**108 CPHẦN4*Bản đồ***

**Cáctuổiđã được lập bản đồ như mộtchữ**

**kiểu dữ liệu (ánh xạ không chính xác).**

GET sinh viên/\_tìm kiếm

{

"lỗi" : {

...

**Elasticsearch đưa ra lỗi phàn nàn rằngchữcác trường không đủ điều kiện để tổng hợp và sắp xếp.**

"loại": [

]

}

{

"tuổi": {

"thứ tự": "desc"

}

}

**Sử dụng mộtloạihoạt động trên mộtchữlĩnh vực này là bất hợp pháp.**

"ngoại lệ\_đối\_luận\_bất\_pháp",

"lý do": "Các trường văn bản không được tối ưu hóa cho các thao tác yêu cầu dữ liệu trường theo từng tài liệu như tổng hợp và sắp xếp, do đó các thao tác này bị vô hiệu hóa theo mặc định.

**Vui lòng sử dụng trường từ khóa thay thế.**

**...**}

**Hình 4.4 Thao tác sắp xếp trên trường văn bản sẽ dẫn đến lỗi.**

Nếu chúng ta thử sắp xếp trên một trường văn bản, Elasticsearch sẽ đưa ra thông báo lỗi cho biết trường đó không được tối ưu hóa để sắp xếp. Nếu chúng ta muốn ghi đè hành vi này và sắp xếp các trường văn bản, chúng ta có thể thực hiện bằng cách đặt fielddata thành true khi xác định lược đồ:

PUT students\_with\_fielddata\_setting

{

"ánh xạ": { "thuộc tính": {

"tuổi": {

"loại": "văn bản",

**"fielddata": đúng**

}

}

}

}

Một lời khuyên: dữ liệu trường được lưu trữ trên bộ nhớ heap của cụm trong bộ đệm dữ liệu trường. Việc bật fielddata dẫn đến các phép tính tốn kém, từ đó có thể dẫn đến các vấn đề về hiệu suất trên cụm. Tôi đề xuất sử dụng các trường từ khóa thay vì bậtfielddata (xem phần tiếp theo).

**FIXING SẮP XẾP ... BẰNG CÁC LOẠI TỪ KHÓA**

May mắn thay, Elasticsearch có thể tạo bất kỳ trường nào dưới dạng nhiều trường với từ khóa là kiểu thứ hai theo mặc định. Sử dụng tính năng mặc định này, trường age được gắn thẻ là nhiều trường, do đó tạo age.keyword với từ khóa là kiểu dữ liệu. Để sắp xếp học sinh, tất cả những gì chúng ta cần làm là đổi tên trường từ age thành age.keyword.

 **Danh sách 4.3 Đã sửa đổiloạisắp xếp truy vấn trêntuổi.từ khóacánh đồng**

NHẬN students\_temp/\_search

{

**Sử dụng API \_search để tìm kiếm tất cả các tài liệu**

* 1. ***Rõ rànglập bản đồ* 109**

"loại": [

{

**Chức năng sắp xếp**

"tuổi.từ khóa": {



"thứ tự": "tăng dần"

**age.keyword là tên của trường.**

}

} **Sắp xếp dữ liệu trong**

] **đang tăng dầnđặt hàng**

}

Truy vấn này sắp xếp tất cả học sinh theo thứ tự tăng dần theo độ tuổi. Truy vấn chạy thành côngvì age.keyword là trường kiểu từ khóa mà chúng ta có thể áp dụng hàm sắp xếp. Theo mặc định, chúng ta sắp xếp theo kiểu dữ liệu thứ hai (age.keyword) do Elasticsearch tạo ra.

**GHI CHÚ**Trong ví dụ, age.keyword là tên mặc định do Elastic-search cung cấp trong quá trình ánh xạ động được suy ra một cách ngầm định. Trong khi đó, chúng ta có toàn quyền kiểm soát việc tạo trường, tên của chúng và loại của chúng khi xác định lược đồ ánh xạ một cách rõ ràng.

Việc xử lý một trường như một kiểu văn bản thay vì một từ khóa sẽ ảnh hưởng không cần thiết đến công cụ tìm kiếm khi dữ liệu được phân tích và chia thành các mã thông báo.

**DGẮN RA ĐỊNH DẠNG NGÀY KHÔNG CHÍNH XÁC**

Một vấn đề tiềm ẩn khác với ánh xạ động là định dạng ngày có thể được xác định không chính xác nếu không được cung cấp trong định dạng ngày mặc định của Elasticsearch (yyyy-MM-dd hoặc yyyy/MM/dd). Ngày được coi là loại văn bản nếu chúng ta đăng nó ở Vương quốc Anh-định dạng dd-MM-yyyy của dom hoặc định dạng MM-dd-yyyy của Hoa Kỳ. Chúng tôi không thể thực hiện phép toán ngày trên các trường liên kết với các kiểu dữ liệu không phải ngày. Các trường như vậy cũng không đủ điều kiện để sắp xếp, lọc và tổng hợp.

**GHI CHÚ**Không có kiểu ngày trong JSON, do đó, tùy thuộc vào các ứng dụng sử dụng để giải mã giá trị và xác định xem đó có phải là ngày hay không. Khi làm việc với Elasticsearch, chúng tôi cung cấp dữ liệu ở định dạng chuỗi cho các trường này: ví dụ: "release\_date" : "2021-07-28".

Điểm mấu chốt khi làm việc với các tính năng ánh xạ động là Elasticsearch có thể có phạm vi ánh xạ sai. Việc chuẩn bị lược đồ của chúng ta dựa trên các giá trị trường có thể không phải lúc nào cũng phù hợp. Vì vậy, lời khuyên chung là phát triển lược đồ dựa trên các yêu cầu về mô hình dữ liệu của chúng ta thay vì dựa vào sự thương xót của công cụ.

Để khắc phục những hạn chế đó, chúng ta có thể chọn một giải pháp thay thế: ánh xạ rõ ràng, trong đó chúng ta xác định lược đồ của mình và tạo nó trước khi quá trình lập chỉ mục bắt đầu. Chúng ta sẽ thảo luận về ánh xạ rõ ràng trong phần tiếp theo.

#### 4.3 Bản đồ rõ ràng

Trong phần trước, chúng ta đã thảo luận về ánh xạ động không có lược đồ của Elasticsearch. Elasticsearch đủ thông minh để lấy thông tin ánh xạ dựa trên tài liệu của chúng ta, nhưng chúng ta có thể đưa ra định nghĩa lược đồ không chính xác. May mắn thay, Elasticsearch cung cấp cho chúng ta cách để ra lệnh cho các định nghĩa ánh xạ.

**110 CPHẦN4*Bản đồ***

Sau đây là hai cách tiếp cận để tạo (hoặc cập nhật) một lược đồ một cách rõ ràng. Hình

* 1. chứng minh việc sử dụng cả hai API để tạo ra mộtphimmục lục:
     + *API lập chỉ mục*—Chúng ta có thể tạo định nghĩa lược đồ tại thời điểm tạo chỉ mục bằng cách sử dụngtạo chỉ mụcAPI (không phải làlập bản đồAPI) cho mục đích này.tạo chỉ mụcAPI mong đợi một yêu cầu bao gồm định nghĩa lược đồ bắt buộc dưới dạng tài liệu JSON. Chỉ mục mới và định nghĩa ánh xạ của nó được tạo đồng thời.
     + *API lập bản đồ*—Khi mô hình dữ liệu của chúng tôi trưởng thành, đôi khi chúng tôi sẽ cần cập nhật định nghĩa lược đồ với các thuộc tính mới. Elasticsearch cung cấp\_lập bản đồđiểm cuối để thực hiện hành động này, cho phép chúng ta thêm các trường và kiểu dữ liệu của chúng.

**Tạo ra mộtphimlập chỉ mục với một tập hợp các trường ban đầu và các loại của chúng (sử dụng API lập chỉ mục)**

**Cập nhậtphimlập chỉ mục với một tập hợp các trường bổ sung và các loại của chúng (sử dụng API ánh xạ)**

ĐẶT phim

{

"ánh xạ": { "thuộc tính": {

"tiêu đề":{

"loại": "văn bản"

}

}

}

PUT phim/\_bản đồ

{

"thuộc tính":{ "ngày phát hành":{

"loại":"ngày",

"định dạng":"dd-mm-yyyy"

}

}

}

}

|  |  |
| --- | --- |
| tiêu đề | chữ |
| ngày phát hành | ngày |

**Cuối cùngphimchỉ mục bao gồm các trường được thêm vào trong quá trình lập chỉ mục và sau đó được cập nhật sau**

**Hình 4.5 Tạo và cập nhật lược đồ bằng cách sử dụngAPI lập chỉ mục và lập bản đồ**

Phần tiếp theo sẽ xem xét hai phương pháp này. Chúng ta sẽ làm theo một ví dụ trong đó chúng ta đang làm việc với dữ liệu nhân viên.

* + 1. Bản đồ sử dụng API lập chỉ mục

Việc tạo định nghĩa ánh xạ tại thời điểm tạo chỉ mục tương đối đơn giản.ward. Chúng tôi chỉ cần đưa ra lệnh PUT theo sau là tên chỉ mục và truyền đối tượng map-pings với tất cả các trường bắt buộc và thông tin chi tiết của chúng làm nội dung của yêu cầu (xem hình 4.6).

***4.3 Rõ rànglập bản đồ* 111**

**Tạo một chỉ mục mới**

ĐẶT <tên\_chỉ\_mục>

{

"ánh xạ": { "thuộc tính": {

"tên\_trường":{

**Cácánh xạđối tượng bao gồm các thuộc tính trường.**

**Tên của trường**

**Đối tượng thuộc tính bao gồm tên trường, kiểu dữ liệu và bất kỳ**

"kiểu":"kiểu\_dữ\_liệu", "tham\_số":"giá\_trị\_tham\_số",

}

**các thông số khác.**

"tên\_trường2":{

..

**Thuộc tính của trường:**

**loại và bất kỳ thuộc tính nào khác**

}

} **Tên của**

} **cánh đồng thứ hai**

}

**Hình 4.6 Tạo định nghĩa ánh xạ trong quá trình tạo chỉ mục**

Bây giờ chúng ta đã biết lý thuyết đằng sau việc sử dụng API ánh xạ để tạo lược đồ, hãy cùng áp dụng lý thuyết này bằng cách phát triển lược đồ ánh xạ cho mô hình nhân viên, như trong danh sách sau. Thông tin nhân viên được mô hình hóa bằng các trường như tên, tuổi, email và địa chỉ.

Chúng tôi lập chỉ mục tài liệu chứa các trường này thành chỉ mục nhân viên bằng cách gọi API ánh xạ qua hành động HTTP PUT. Nội dung yêu cầu được đóng gói với các thuộc tính cho các trường của chúng tôi.

**Liệt kê 4.4 Tạo sơ đồ trước**

Nhân viên PUT

{

**Trường email là một loại từ khóa.**

**Thuộc tính đối tượng bên trong**

"ánh xạ": { "thuộc tính": {

"tên":{ "loại": "văn bản"},

"tuổi": {"kiểu": "số nguyên"},

"email": {"type": "từ khóa"}, "địa chỉ":{

"của cải": {

"đường phố":{ "loại":"văn bản" },

"quốc gia":{ "loại":"văn bản" }

**Trường tên là kiểu dữ liệu văn bản.**

**Trường tuổi là một kiểu số.**

**Địa chỉ đối tượng trong một đối tượng bên trong với các trường tiếp theo**

}

}

}

}

}

Khi tập lệnh đã sẵn sàng, hãy thực hiện lệnh này trong Dev Tools của Kibana. Bạn sẽ nhận được phản hồi thành công ở khung bên phải, cho biết chỉ mục đã được tạo.

**112 CPHẦN4*Bản đồ***

Bây giờ chúng tôi có một chỉ số nhân viên vớiánh xạ lược đồ mong đợi. Trong ví dụ này, chúng tôi đã chỉ định các kiểu cho Elasticsearch; bây giờ chúng tôi kiểm soát lược đồ.

Bạn có để ý không?trường địa chỉ trong danh sách 4.4? Đây là một loại đối tượng bao gồmcác trường bổ sung: phố và quốc gia. Tuy nhiên, loại trường địa chỉ chưa được đề cập là đối tượng, mặc dù chúng tôi đã nói rằng đó là một đối tượng đóng gói các trường dữ liệu khác. Điều này là do loại dữ liệu đối tượng được Elasticsearch suy ra cho bất kỳ đối tượng bên trong nào theo mặc định—do đó chúng tôi đã không đề cập rõ ràng. Ngoài ra, thuộc tính trường con đối tượng được gói trong địa chỉ giúp xác định các thuộc tính khác của đối tượng bên trong.

Bây giờ chỉ mục nhân viên của chúng tôi đang trong quá trình sản xuất, giả sử chúng tôi được đưa ra một yêu cầu mới: ban quản lý muốn chúng tôi mở rộng mô hình để bao gồm thêm một số thuộc tính như phòng ban, số điện thoại và các thuộc tính khác. Để đáp ứng yêu cầu này, chúng tôi phải thêm các trường này bằng API ánh xạ trên chỉ mục trực tiếp. Chúng tôi sẽ xem xét chủ đề này một cách ngắn gọn trong phần tiếp theo.

* + 1. Cập nhật lược đồ bằng API ánh xạ

Khi dự án của chúng tôi trưởng thành, mô hình dữ liệu chắc chắn cũng sẽ thay đổi. Tiếp tụcvới tài liệu nhân viên của chúng ta, hãy thêm các thuộc tính joined\_date và phone\_number:

{

"tên":"John Smith", "ngày\_tham\_gia":"01-05-2021", "số\_điện\_thoại":"01234567899"

...

}

Ngày tham gia là một kiểu ngày vì chúng ta muốn thực hiện các hoạt động liên quan đến ngày, chẳng hạn như sắp xếp nhân viên theo ngày này. Số điện thoại dự kiến ​​sẽ được lưu trữ nguyên trạng (chúng ta không muốn dữ liệu được mã hóa—sẽ nói thêm về điều này trong các phần tiếp theo), do đó, nó phù hợp với kiểu dữ liệu từ khóa. Để sửa đổi định nghĩa lược đồ nhân viên hiện có bằng các trường bổ sung này, chúng ta gọi điểm cuối \_mapping trên chỉ mục hiện có, khai báo các trường mới trong đối tượng yêu cầu.

**Liệt kê 4.5 Cập nhật chỉ mục hiện có với các trường bổ sung**

**Định dạng ngày dự kiến**

PUT nhân viên/\_bản đồ

{

"thuộc tính":{ "ngày\_tham\_gia":{

"loại":"ngày",

"định dạng":"dd-MM-yyyy"

},

"phone\_number":{ "type":"từ khóa"

}

}

}

**\_mapping điểm cuối để cập nhật chỉ mục hiện có**

**Trường ngày tham gia là một loại ngày.**

**Số điện thoại được lưu trữ nguyên trạng.**

* 1. ***Rõ rànglập bản đồ* 113**

Nếu bạn xem xét kỹ phần thân yêu cầu, đối tượng thuộc tính được định nghĩa ở cấp gốc. Điều này không giống với phương pháp trước đó là tạo lược đồ bằng API lập chỉ mục, trong đó đối tượng thuộc tính được gói trong đối tượng ánh xạ cấp gốc.

**TRONGPDATING MỘT CHỈ MỤC TRỐNG**

Chúng ta cũng có thể sử dụng cùng một nguyên tắc cập nhật lược đồ trên một chỉ mục trống.*chỉ mục trống*được tạo mà không có ánh xạ lược đồ (ví dụ: thực hiện lệnh PUT books sẽ tạo ra một chỉ mục books trống).

Tương tự như cách chúng ta cập nhật chỉ mục trong danh sách 4.5 bằng cách gọi điểm cuối \_mapping với định nghĩa lược đồ bắt buộc, chúng ta có thể sử dụng cùng cách tiếp cận cho chỉ mục trống. Đoạn mã sau cập nhật lược đồ trên chỉ mục department với một trường bổ sung.

**Liệt kê 4.6 Cập nhật lược đồ ánh xạ của một chỉ mục trống**

Các phòng ban của PUT

PUT các phòng ban/\_bản đồ

{

"thuộc tính":{ "tên":{

"loại":"văn bản"

}

}

}

**Đầu tiên tạo một chỉ mục trống**

**Sử dụng API ánh xạ để cập nhật chỉ mục trống**

**Trường tên được khai báo là kiểu văn bản.**

Cho đến nay, chúng ta đã thấy trường hợp bổ sung của việc cập nhật lược đồ với các trường bổ sung. Nhưng nếu chúng ta muốn thay đổi kiểu dữ liệu của các trường hiện có thì sao? Giả sử chúng ta muốnthay đổi kiểu trường từ văn bản thành từ khóa. Chúng ta có thể cập nhật ánh xạ dễ dàng như khi chúng ta thêm trường không? Câu trả lời ngắn gọn là không. Elasticsearch không cho phép chúng ta thay đổi hoặc sửa đổi kiểu dữ liệu của các trường hiện có. Thay vào đó, chúng ta phải làm thêm một chút công việc, như đã thảo luận trong phần tiếp theo.

* + 1. Không được phép sửa đổi các trường hiện có

Khi một chỉ mục đã hoạt động (được tạo bằng các trường dữ liệu và hoạt động), mọi sửa đổi đối với các trường hiện có trên chỉ mục đang hoạt động đều bị cấm. Ví dụ, nếu một trường được định nghĩa làkiểu dữ liệu từ khóa và được lập chỉ mục, nó không thể được thay đổi thành kiểu dữ liệu khác (ví dụ, từ từ khóa thành văn bản). Có một lý do chính đáng cho việc này. Dữ liệu được lập chỉ mục bằng cách sử dụng các định nghĩa lược đồ hiện có và do đó, được lưu trữ trong chỉ mục. Việc tìm kiếm trên dữ liệu trường đó sẽ không thành công nếu kiểu dữ liệu đã được sửa đổi, dẫn đến trải nghiệm tìm kiếm không chính xác. Để tránh lỗi tìm kiếm, Elasticsearch không cho phép chúng ta sửa đổi các trường hiện có.

Vâng, bạn có thể hỏi, giải pháp thay thế là gì? Các yêu cầu kinh doanh thay đổi, cũng như các yêu cầu kỹ thuật. Làm thế nào chúng ta có thể sửa các kiểu dữ liệu (có thể đã được tạo không chính xác) trên một chỉ mục trực tiếp?

Chúng ta có thể sử dụng reindexing. Các hoạt động reindexing lấy dữ liệu từ chỉ mục gốc sang chỉ mục mới (có thể có định nghĩa lược đồ được cập nhật). Ý tưởng như sau:

**114 CPHẦN4*Bản đồ***

**1** Tạo một chỉ mục mới với các định nghĩa lược đồ đã cập nhật.

**2**Sao chép dữ liệu từ chỉ mục cũ vào chỉ mục mới bằng API lập chỉ mục lại. Chỉ mục mới với lược đồ mới đã sẵn sàng để sử dụng sau khi lập chỉ mục lại hoàn tất. Chỉ mục mở cho cả hoạt động đọc và ghi.

**3** Khi chỉ mục mới đã sẵn sàng, ứng dụng của chúng tôi sẽ chuyển sang chỉ mục đó.

**4** Chúng tôi sẽ xóa chỉ mục cũ sau khi xác nhận chỉ mục mới hoạt động như mong đợi.

Lập chỉ mục lại là một hoạt động mạnh mẽ và được thảo luận chi tiết trong chương 5, nhưng chúng ta hãy xem nhanh cách API hoạt động. Giả sử chúng ta muốn di chuyển dữ liệu từ một chỉ mục hiện có (nguồn) sang một chỉ mục đích (đích). Chúng ta đưa ra lệnh gọi lập chỉ mục lại, như được hiển thị trong danh sách 4.7.

**GHI CHÚ**Các chỉ số này hiện chưa tồn tại. Danh sách này chỉ mang tính thông tin tham khảo.

**Liệt kê 4.7 Di chuyển dữ liệu giữa các chỉ mục bằng cách sử dụng lập chỉ mục lạiGiao diện lập trình ứng dụng (API)**

POST \_reindex

{

"nguồn": {"index": "đơn hàng"},

"đích": {"index": "orders\_new"}

}

Chỉ mục orders\_new mới được tạo bằng các thay đổi đối với lược đồ, sau đó dữ liệu từ chỉ mục cũ (orders) được di chuyển đến chỉ mục mới tạo này với các khai báo được cập nhật.

**GHI CHÚ**Nếu ứng dụng của bạn được liên kết chặt chẽ với một chỉ mục hiện có, việc di chuyển đến chỉ mục mới có thể yêu cầu thay đổi mã hoặc cấu hình. Cách lý tưởng để tránh những tình huống như vậy là sử dụng bí danh. Bí danh là tên thay thế được đặt cho các chỉ mục. Việc đặt bí danh giúp chúng ta chuyển đổi giữa các chỉ mục một cách liền mạch mà không mất thời gian chết. Chúng ta sẽ xem xét bí danh trong chương 6.

Đôi khi dữ liệu có kiểu không chính xác khi tài liệu được lập chỉ mục. Ví dụ, mộttrường rating kiểu float có thể nhận giá trị được bao quanh dưới dạng chuỗi: "rating": "4.9" thay vì "rating":4.9. May mắn thay, Elasticsearch rất dễ tha thứ khi gặp phải các giá trị không khớp với kiểu dữ liệu. Nó tiến hành lập chỉ mục tài liệu bằng cách trích xuất giá trị và lưu trữ trong kiểu dữ liệu gốc. Chúng ta sẽ thảo luận về cơ chế này trong phần tiếp theo.

* + 1. Kiểu ép buộc

Đôi khi các tài liệu JSON chứa các giá trị không chính xác khác với các giá trị trong định nghĩa lược đồ. Ví dụ, một trường được định nghĩa bằng số nguyên có thể được lập chỉ mục bằng giá trị chuỗi. Elasticsearch cố gắng chuyển đổi các loại không nhất quán như vậy, do đó tránh được các vấn đề lập chỉ mục. Quá trình này được gọi là ép kiểu. Ví dụ, giả sử chúng ta đã định nghĩatrường tuổi của xe là một kiểu số nguyên:

"tuổi":{"loại":"ngắn"}

* 1. ***Dữ liệucác loại* 115**

Lý tưởng nhất là chúng ta lập chỉ mục tài liệu của mình với age là giá trị số nguyên. Tuy nhiên, người dùng có thể vô tình đặt giá trị age là kiểu chuỗi ("2") và gọi lệnh gọi.

**Liệt kê 4.8 Thiết lập giá trị chuỗi cho trường số**

ĐẶT xe/\_doc/1

{

"làm":"BMW",

"mô hình": "X3",

"tuổi": "2"

}

**Một số nguyên đầu vào dưới dạng chuỗi**

Elasticsearch lập chỉ mục tài liệu này mà không có bất kỳ lỗi nào. Mặc dù trường age được mong đợi sẽ chứa một số nguyên, Elasticsearch ép kiểu dữ liệu để chúng ta tránh lỗi. Việc ép kiểu dữ liệu chỉ hoạt động nếu giá trị phân tích cú pháp của trường khớp với kiểu dữ liệu mong đợi. Trong trường hợp này, việc phân tích cú pháp "2" sẽ cho kết quả là 2, là một số.

Cho đến nay, chúng ta đã nói nhiều về ánh xạ và kiểu dữ liệu mà không nhấn mạnh nhiều vào kiểu dữ liệu. Thiết kế lược đồ của bạn với các kiểu dữ liệu phù hợp là rất quan trọng trong việc tối ưu hóa trải nghiệm tìm kiếm. Chúng ta cần hiểu các kiểu dữ liệu, đặc điểm của chúng và thời điểm sử dụng chúng. Không giống như cơ sở dữ liệu hoặc ngôn ngữ lập trình, Elasticsearch đi kèm với một danh sách dài các kiểu dữ liệu, đáp ứng hầu hết mọi hình dạng và dạng dữ liệu. Phần tiếp theo sẽ xem xét chi tiết các kiểu dữ liệu.

#### Kiểu dữ liệu

Tương tự như kiểu dữ liệu của biến trong ngôn ngữ lập trình, các trường trong tài liệu có kiểu dữ liệu cụ thể được liên kết với chúng. Elasticsearch cung cấp danh sách phong phú các kiểu dữ liệu, từ đơn giản đến phức tạp đến chuyên biệt. Danh sách các kiểu này ngày càng dài ra, vì vậy hãy theo dõi để biết thêm.

Elasticsearch cung cấp hơn hai chục loại dữ liệu khác nhau, do đó chúng ta có thể lựa chọn tốt các loại dữ liệu phù hợp dựa trên nhu cầu cụ thể của mình. Các loại dữ liệu có thể được phân loại rộng rãi thành các loại sau:

* + - *Các loại đơn giản*—Các kiểu dữ liệu phổ biến biểu diễn chuỗi (thông tin văn bản), ngày tháng, số và các biến thể dữ liệu cơ bản khác. Ví dụ làchữ,Boolean,dài,ngày,gấp đôi, Vànhị phân.
    - *Các loại phức tạp*—Được tạo ra bằng cách kết hợp các kiểu bổ sung, tương tự như một đối tượng - xây dựng trong ngôn ngữ lập trình, trong đó các đối tượng có thể chứa các đối tượng bên trong. Các kiểu phức tạp có thể được làm phẳng hoặc lồng nhau để tạo ra dữ liệu phức tạp hơncấu trúc dựa trên các yêu cầu. Ví dụ làsự vật,lồng nhau,phẳng, Vàtham gia.
    - *Các loại chuyên biệt*—Được sử dụng chủ yếu cho các trường hợp chuyên biệt như định vị địa lý và địa chỉ IP. Ví dụ làhình dạng địa lý,điểm địa lý, VàIPvà các loại phạm vichẳng hạn nhưphạm vi ngày thángVàphạm vi ip.

**GHI CHÚ**Có thể tham khảo đầy đủ các loại trong tài liệu chính thức:<http://mng.bz/yQpe>.

**116 CPHẦN4*Bản đồ***

Mỗi trường trong tài liệu có thể có một hoặc nhiều loại liên quan dựa trên yêu cầu về dữ liệu và kinh doanh. Bảng 4.1 cung cấp danh sách các loại dữ liệu phổ biến với một số ví dụ.

**Bảng 4.1 Các kiểu dữ liệu phổ biến với ví dụ**

|  |  |  |
| --- | --- | --- |
| **Kiểu** | **Sự miêu tả** | **Ví dụ** |
| chữ | Biểu thị thông tin dạng văn bản (như giá trị chuỗi); văn bản không có cấu trúc | Tiêu đề phim, bài đăng trên blog, bài đánh giá sách, tin nhắn nhật ký |
| số nguyên, dài,ngắn, byte | Biểu thị một số | Số ca nhiễm, chuyến bay bị hủy, sản phẩm bán ra, thứ hạng của sách |
| nổi, đôi | Đại diệnmột số dấu phẩy động | Điểm trung bình của sinh viên, điểm trung bình động của doanh số, xếp hạng trung bình của người đánh giá, nhiệt độ |
| Boolean | Biểu thị một lựa chọn nhị phân: đúng hoặc sai | Bộ phim có phải là bom tấn không? Các ca mắc COVID có đang gia tăng không? Học sinh đã vượt qua kỳ thi chưa? |
| từ khóa | Biểu thị văn bản có cấu trúc: văn bản không được chia nhỏ hoặc phân tích | Mã lỗi, địa chỉ email, số điện thoại, số An sinh xã hội |
| sự vật | Biểu diễn một đối tượng JSON | (Theo định dạng JSON) thông tin chi tiết về nhân viên,tweet, đối tượng phim |
| lồng nhau | Biểu diễn một mảng các đối tượng | Địa chỉ nhân viên, dữ liệu định tuyến email, đội ngũ kỹ thuật của một bộ phim |

Như bạn có thể hình dung, đây không phải là danh sách toàn diện về các kiểu dữ liệu. Cuốn sách này được viết cho phiên bản 8.6 và gần 40 kiểu dữ liệu được Elasticsearch định nghĩa. Elasticsearch định nghĩa các kiểu ở mức độ vi mô trong một số trường hợp để tối ưu hóa các truy vấn tìm kiếm. Ví dụ, các kiểu văn bản được phân loại thành các kiểu cụ thể hơn như search\_as\_you\_type, match\_only\_text, completion, token\_count và các kiểu khác.

Elasticsearch cũng cố gắng nhóm các kiểu dữ liệu theo họ để tối ưu hóa không gian và hiệu suất. Ví dụ, từ khóa họ có các kiểu dữ liệu từ khóa, ký tự đại diện và hằng số\_. Hiện tại đây là nhóm họ duy nhất; chúng ta có thể mong đợi nhiều hơn trong tương lai gần.

**Một trường duy nhất với nhiều kiểu dữ liệu**

Trong các ngôn ngữ lập trình như Java và C#, chúng ta không thể định nghĩa một biến có hai kiểu dữ liệu khác nhau. Tuy nhiên, Elasticsearch không có hạn chế như vậy. Elasticsearch khá tuyệt khi biểu diễn một trường có nhiều kiểu dữ liệu, cho phép chúng ta phát triển lược đồ theo cách chúng ta muốn.

Ví dụ, chúng ta có thể muốn tác giả của một cuốn sách vừa là mộtchữloại và mộttừ khóaloại. Mỗi trường có một đặc điểm cụ thể và các từ khóa sẽ không được phân tích, nghĩa là trường được lưu trữ nguyên trạng. Chúng ta cũng có thể có nhiều loại hơn—ví dụ:tác giảtrường có thể được khai báo làhoàn thànhloại ngoài hai loại kia.

* 1. ***Dữ liệucác loại* 117**

Trong một vài phần tiếp theo, chúng ta sẽ xem xét các kiểu dữ liệu này, đặc điểm của chúng và cách sử dụng của chúng. Nhưng trước khi làm việc chi tiết với các kiểu dữ liệu, chúng ta cần hiểu cách tạo định nghĩa ánh xạ. Phần 4.3 đào sâu hơn vào cách làm việc với các tính năng ánh xạ, nhưng hiện tại chúng ta sẽ xem xét sơ qua một cơ chế để tạo định nghĩa ánh xạ vì đây là điều kiện tiên quyết cho các phần tiếp theo.



**Phát triển sơ đồ lập bản đồ**

Khi chúng tôi lập chỉ mục một tài liệu phim (danh sách 4.1), Elasticsearch đã tạo một lược đồ động (hình 4.2), lấy các kiểu bằng cách phân tích các giá trị trường. Bây giờ chúng ta đang ở thời điểm mà chúng ta muốn tạo các định nghĩa lược đồ của mình. Elasticsearch sử dụng API lập chỉ mục để tạo các định nghĩa này, vì vậy hãy kiểm tra bằng một ví dụ.

Giả sử chúng ta muốn tạo một sinh viên vớitênVàtuổicác trường có kiểu dữ liệuchữVàbyte,tương ứng (phần tiếp theo giới thiệu các kiểu dữ liệu này). Chúng ta có thể tạosinh viênlập chỉ mục với các định nghĩa ánh xạ này ở phía trước, do đó có thểkiểm soát việc tạo lược đồ. Hình minh họa điều này một cách ngắn gọn, với việc thực thi định nghĩa ánh xạ ở phía bên trái và lấy định nghĩa ở phía bên phải.

Học sinh PUT

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

},

"tuổi":{

"kiểu": "byte"

}

}

}

}

GET sinh viên/\_mapping

**Lấy các ánh xạ cho chỉ mục sinh viên**

{

**Cácsinh viênchỉ mục được định nghĩa bằng một lược đồ**

**bao gồmtên(chữ) Vàtuổi(byte).**

"sinh viên" : {

"ánh xạ" : { "thuộc tính" : {

"tuổi" : {

"kiểu" : "byte"

},

"tên" : {

"loại" : "văn bản"

}

}

}

**Cácsinh viên**

}

}

**định nghĩa lập bản đồ**

**Định nghĩa bản đồ (trái) và truy xuất lược đồ (phải)**

Chúng tôi tạo chỉ mục bằng một yêu cầu chứa một tập hợp các thuộc tính trường được gói trong một

ánh xạđối tượng. Chúng tôi gọi\_lập bản đồđiểm cuối trên chỉ mục để kiểm tra định nghĩacủa sơ đồ (hiển thị ở phía bên phải).

Bây giờ chúng ta đã biết cách tạo lược đồ ở cấp độ cao, chúng ta đã sẵn sàng để kiểm tra các kiểu dữ liệu cốt lõi.

**118 CPHẦN4*Bản đồ***

#### Các kiểu dữ liệu cốt lõi

Elasticsearch cung cấp một tập hợp các kiểu dữ liệu cốt lõi để biểu diễn dữ liệu, bao gồm text, keyword, date, long và boolean. Chúng ta cần hiểu biết cơ bản về các kiểu dữ liệu cốt lõi trước khi có thể hiểu các kiểu nâng cao và ánh xạ động và rõ ràng. Trong một vài phần tiếp theo, chúng ta sẽ xem xét các kiểu dữ liệu cơ bản cốt lõi với các ví dụ.

**GHI CHÚ** Để tránh làm chương này dài dòng, bạn có thể tìm thấy mã cho một số ví dụ trên GitHub (<http://mng.bz/GyBR>) và trang web của cuốn sách ([https://](https://www.manning.com/books/elasticsearch-in-action-second-edition) [www.manning.com/books/elasticsearch-in-action-second-edition](https://www.manning.com/books/elasticsearch-in-action-second-edition)).

* + 1. Kiểu dữ liệu văn bản

Nếu có một loại dữ liệu mà công cụ tìm kiếm phải làm tốt, thì đó là loại dữ liệu toàn văn. Văn bản có thể đọc được của con người, còn được gọi là văn bản đầy đủ hoặc văn bản phi cấu trúc trong thuật ngữ của công cụ tìm kiếm, là nền tảng của công cụ tìm kiếm hiện đại. Chúng ta sử dụng rất nhiều thông tin dạng văn bản trong thế giới kỹ thuật số hiện tại: các mục tin tức, tweet, bài đăng trên blog, bài báo nghiên cứu và nhiều hơn nữa. Elasticsearch định nghĩa một loại dữ liệu chuyên dụng để xử lý dữ liệu toàn văn: loại dữ liệu văn bản.

Bất kỳ trường nào được đóng dấu bằng kiểu dữ liệu văn bản đều được phân tích trước khi được lưu lại. Trong quá trình phân tích, các trình phân tích sẽ xử lý (làm giàu, tăng cường và chuyển đổi) dữ liệu văn bản thành nhiều dạng khác nhau và lưu trữ chúng trong các cấu trúc dữ liệu nội bộ để dễ dàng truy cập.

Như chúng ta đã thấy trong các ví dụ trước, việc thiết lập kiểu rất đơn giản. Sử dụng cú pháp này trong định nghĩa ánh xạ của bạn: "field name":{ "type": "text" }. Chúng ta đã thực hiện một ví dụ trong phần 4.2.2, vì vậy chúng ta sẽ không lặp lại bài tập đó. Thay vào đó, hãy xem cách Elasticsearch xử lý các trường văn bản trong quá trình lập chỉ mục.

**MỘTPHÂN TÍCH CÁC TRƯỜNG VĂN BẢN**

Elasticsearch hỗ trợ cả hai loại dữ liệu văn bản—có cấu trúc và không có cấu trúc. Văn bản không có cấu trúc là dữ liệu toàn văn, thường được viết bằng ngôn ngữ mà con người có thể đọc được (tiếng Anh, tiếng Trung, tiếng Bồ Đào Nha, v.v.). Tìm kiếm hiệu quả và hiệu suất trên văn bản không có cấu trúc giúp công cụ tìm kiếm nổi bật.

Văn bản phi cấu trúc trải qua quá trình phân tích trong đó dữ liệu được chia thành các mã thông báo, các ký tự được lọc ra, các từ được rút gọn thành từ gốc (gốc), các từ đồng nghĩa được thêm vào và các quy tắc xử lý ngôn ngữ tự nhiên khác được áp dụng. Chúng tôi dành chương 7 cho phân tích văn bản, nhưng hãy nhanh chóng xem một ví dụ về cách Elasticsearch xử lý toàn bộ văn bản. Văn bản sau đây là đánh giá của người dùng về một bộ phim:

"Bộ phim thật tuyệt!!! Hài hước :) :) và HƯỚC MƠ ;) một Kẻ giết người"

Khi tài liệu này được lập chỉ mục, nó sẽ trải qua quá trình phân tích dựa trên trình phân tích. Trình phân tích là các mô-đun phân tích văn bản phân tích văn bản đến để mã hóa và chuẩn hóa văn bản. Theo mặc định, Elasticsearch sử dụng trình phân tích chuẩn và việc phân tích bình luận đánh giá bao gồm các bước sau (hình 4.7):

**1**Thẻ, dấu câu và ký tự đặc biệt được loại bỏ bằng bộ lọc ký tự. Sau bước này, đánh giá sẽ trông như thế này:

Bộ phim thật là buồn cười và dí dỏm, một kẻ giết người

***4.5 Kiểu dữ liệu cốt lõi***

**2** Câu được chia thành các mã thông báo bằng cách sử dụng công cụ phân tích mã thông báo:

[bộ phim, thật, buồn cười, và, dí dỏm, là, Kẻ giết người,]

**3** Các mã thông báo được đổi thành chữ thường bằng cách sử dụng bộ lọc mã thông báo:

[bộ phim, thật là, bệnh hoạn, buồn cười, và dí dỏm, một, kẻ giết người,]

"Bộ phim thật tuyệt vời!!! Thật buồn cười :)

:) và HƯỚC MƠ ;) một Kẻ giết người"

**119**

Bộ phim thật là buồn cười và dí dỏm, một kẻ giết người

**Các ký tự và dấu câu không cần thiết sẽ được loại bỏ bằng bộ lọc ký tự.**

[bộ phim, là,

bệnh tật, buồn cười,

**Văn bản được chia thành các từ (mã thông báo) bằng**

và, WITTY, một, Kẻ giết người,]

**bộ phân tích dữ liệu.**

**Mã thông báo được viết thường bằng cách sử dụng bộ lọc mã thông báo.**

[bộ phim, thật là, bệnh hoạn, buồn cười, và dí dỏm, một, kẻ giết người,]

**Hình 4.7 Xử lý trường toàn văn trong quá trình lập chỉ mục bằng mô-đun phân tích chuẩn**

Các bước này có thể khác nhau, tùy thuộc vào trình phân tích của bạn. Ví dụ, nếu bạn chọn trình phân tích tiếng Anh, các mã thông báo được rút gọn thành các từ gốc (gốc):

[movi,bệnh,hilari, witti, kẻ giết người,]

Bạn có để ý đến các từ gốc movi, hilari và witti không? Chúng không phải là những từ thực sự, nhưng cách viết sai không quan trọng miễn là tất cả các dạng phái sinh có thể khớp với các từ gốc.

**GHI CHÚ**Chúng ta có thể sử dụng API \_analyze do Elasticsearch cung cấp để kiểm tra cách phân tích văn bản. API này giúp chúng ta hiểu được hoạt động bên trong của các trình phân tích và hỗ trợ xây dựng các mô-đun trình phân tích phức tạp và tùy chỉnh cho các yêu cầu ngôn ngữ khác nhau.

Mã có sẵn trên kho lưu trữ GitHub. Bạn có thể thoải mái thử nghiệm chạy trình phân tích trên dữ liệu văn bản.

**Nhét đầy**

*Nhét đầy*là một quá trình rút gọn các từ thành các từ gốc của chúng. Ví dụ,*máy bay chiến đấu*,*trận đánh*, Và*đã chiến đấu*tất cả có thể được rút gọn thành một từ:*trận đánh*. Tương tự như vậy,*phim*có thể được giảm xuống*di chuyển*Và*vui vẻ*ĐẾN*vui vẻ*, như trong các ví dụ trước.

**120 CPHẦN4*Bản đồ***

***(tiếp theo)***

Stemmer phụ thuộc vào ngôn ngữ. Ví dụ, chúng ta có thể sử dụng một stemmer tiếng Pháp nếu ngôn ngữ được chọn của tài liệu là tiếng Pháp. Stemmer được khai báo thông qua bộ lọc mã thông báo khi soạn thảo mô-đun phân tích trong giai đoạn phân tích văn bản trong Elasticsearch.

Quá trình tương tự được kích hoạt lại trong quá trình thực hiện truy vấn tìm kiếm trên cùng một trường. Chương 7 dành riêng cho việc phân tích toàn văn và thảo luận về sự phức tạp của trình phân tích và cách Elasticsearch quản lý dữ liệu toàn văn.

Chúng tôi đã đề cập trước đó rằng Elasticsearch định nghĩa các loại ở mức độ vi mô: để kiểm tra-ple, phân loại thêm các trường văn bản thành các loại cụ thể hơn như search\_as\_you\_ type, match\_only\_text, completion, token\_count và các loại khác. Chúng ta hãy cùng xem xét sơ qua các loại văn bản chuyên biệt này trong một vài phần tiếp theo để xem Elasticsearch chú trọng và nỗ lực hơn vào toàn văn như thế nào.

**T****ĐỒNG HỒ CỦA ANH TA\_KIỂU DỮ LIỆU ĐẾM**

Một dạng chuyên biệt của kiểu dữ liệu văn bản, token\_count, định nghĩa một trường nắm bắt số lượng mã thông báo trong trường đó. Ví dụ, nếu chúng ta đã định nghĩa tiêu đề của một cuốn sách là token\_count, chúng ta có thể truy xuất tất cả các cuốn sách dựa trên số lượng mã thông báo mà một cuốn sách có. Hãy tạo một ánh xạ cho điều này bằng cách tạo một chỉ mục với một trường tiêu đề.

**Liệt kê 4.9 Chỉ mục với mộtsố lượng tokenkiểu dữ liệu chotiêu đềcánh đồng**

ĐẶT sách công nghệ

{

"ánh xạ": { "thuộc tính": {

"tiêu đề": {

**Của lĩnh vực**

**tên Kiểu dữ liệu của tiêu đề**

"loại": "token\_count", "máy phân tích": "chuẩn"

}

}

}

}

**là token\_count.**

**Dự kiến ​​máy phân tích sẽ được cung cấp.**

Như bạn thấy, tiêu đề được định nghĩa là kiểu token\_count. Dự kiến ​​trình phân tích sẽ được cung cấp, vì vậy chúng ta đặt trình phân tích chuẩn trên trường tiêu đề. Bây giờ, hãy lập chỉ mục ba cuốn sách kỹ thuật—cuốn này và những cuốn sách trong tương lai của tôi (hy vọng là vậy)—trước khi tìm kiếm chúng dựa trên tính năng số lượng mã thông báo của tiêu đề.

**Liệt kê 4.10 Lập chỉ mục ba tài liệu mới vàosách công nghệchỉ số**

ĐẶT tech\_books/\_doc/1

{

"title":"Elasticsearch trong hành động"

}

***4.5 Dữ liệu cốt lõicác loại* 121**

ĐẶT tech\_books/\_doc/2

{

"title":"Elasticsearch dành cho nhà phát triển Java"

}

ĐẶT tech\_books/\_doc/3

{

"title":"Elastic Stack đang hoạt động"

}

Bây giờ chỉ mục tech\_books chứa một vài cuốn sách, hãy sử dụng loại token\_count. Truy vấn phạm vi sau đây sẽ lấy các cuốn sách có tiêu đề gồm hơn ba từ (gt là viết tắt của lớn hơn) nhưng ít hơn hoặc bằng năm từ (lte là viết tắt của nhỏ hơn hoặc bằng).

**Liệt kê 4.11phạm vitruy vấn tìm kiếm sách có số lượng từ nhất định**

NHẬN tech\_books/\_search

{

"truy vấn": {

"phạm vi": {

"tiêu đề": {

"gt": 3,

"lte": 5

}

}

}

}

Cácphạm vitruy vấn tìm kiếm sách dựa trên số lượng từ trong tiêu đề. Truy vấn này tìm kiếm Elasticsearch for Java Developers (bốn mã thông báo) và Elastic Stack in Action (bốn mã thông báo) nhưng bỏ qua Elasticsearch in Action (ba mã thông báo).

Chúng ta cũng có thể kết hợp trường tiêu đề dưới dạng kiểu văn bản và token\_type, vì Elastic-search cho phép khai báo một trường duy nhất với nhiều kiểu dữ liệu (multi-fields—được thảo luận chi tiết trong phần 4.7). Danh sách sau đây tạo một chỉ mục mới (tech\_books2) bằng kỹ thuật này.

**Liệt kê 4.12 Thêmsố lượng tokennhư một kiểu dữ liệu bổ sung cho mộtchữcánh đồng**

ĐẶT tech\_books2

{

"ánh xạ": { "thuộc tính": {

"tiêu đề": { "kiểu": "văn bản", "trường": {

**Trường tiêu đề được định nghĩa là kiểu dữ liệu văn bản.**

**Trường tiêu đề được khai báo có nhiều kiểu dữ liệu.**

**Loạicủa số lượng từ**

"số\_từ": {

"loại": "token\_count", "máy phân tích": "chuẩn"

}

}

**word\_count là trường bổ sung.**

**Việc cung cấp máy phân tích là bắt buộc.**

**122 CPHẦN4*Bản đồ***

}

}

}

}

Vì trường word\_count là thuộc tính bên trong của trường tiêu đề nên chúng ta có thể sử dụng truy vấn thuật ngữ (một loại truy vấn chạy trên dữ liệu có cấu trúc như số, ngày, Boolean, v.v.).

**Liệt kê 4.13 Tìm kiếm sách có tiêu đề bốn từ**

NHẬN tech\_books/\_search

{

"truy vấn": {

"thuật ngữ": {

**Chúng tôi đang sử dụng**

**một truy vấn thuật ngữ.**

"title.word\_count": { "giá trị": 4

**Tên của trường**

}

}

}

}

Chúng tôi sử dụng<trường\_ngoài>.<trường\_trong>như làsố lượng từtên trường, vì vậytiêu đề

.số\_từlà phần tử truy cập của trường.

Ngoài rasố lượng token, cácchữloại có các hậu duệ khác, chẳng hạn nhưtìm kiếm\_theo\_kiểu\_bạnVàhoàn thành. Vì lý do không gian, chúng tôi sẽ không thảo luận về chúng trongcuốn sách này. Chúng ta hãy tiếp tục tìm hiểu về các kiểu dữ liệu phổ biến, vớitừ khóalà người tiếp theo.

* + 1. Các loại dữ liệu từ khóa

Cáctừ khóahọ các kiểu dữ liệu bao gồm từ khóa, từ khóa hằng số, Và

ký tự đại diện. Chúng ta hãy xem xét các loại này.

**TLOẠI TỪ KHÓA**

Dữ liệu có cấu trúc, chẳng hạn như mã PIN, tài khoản ngân hàng và số điện thoại, không cần phải được tìm kiếm như các kết quả khớp một phần hoặc tạo ra các kết quả có liên quan. Các kết quả có xu hướng cung cấp đầu ra nhị phân: một kết quả được trả về nếu có sự khớp hoặc không có kết quả nào được trả về. Kiểu truy vấn này không quan tâm đến việc tài liệu khớp tốt như thế nào, vì vậy chúng tôi không mong đợi điểm liên quan liên quan đến các kết quả. Dữ liệu có cấu trúc như vậy được biểu diễn dưới dạngkiểu dữ liệu từ khóa trong Elasticsearch.

Kiểu dữ liệu từ khóa không đụng đến các trường. Trường không được mã hóa và khôngđã phân tích. Ưu điểm của trường từ khóa là chúng có thể được sử dụng trong tổng hợp dữ liệu, truy vấn phạm vi và các hoạt động lọc và sắp xếp trên dữ liệu. Để đặt loại từ khóa, hãy sử dụng định dạng này:

"field\_name":{ "type": "từ khóa" }

Ví dụ, đoạn mã sau đây tạo ra một thuộc tính email với kiểu dữ liệu từ khóa.

***4.5 Dữ liệu cốt lõicác loại* 123**

**Liệt kê 4.14 Định nghĩae-mailnhư mộttừ khóagõ cho mộtkhoatài liệu**

Khoa PUT

{

"ánh xạ": { "thuộc tính": {

"thư điện tử": {

**Xác định thuộc tính email**

"loại": "từ khóa"

}

}

}

}

**Khai báo email**

**như một loại từ khóa**

Chúng ta cũng có thể khai báo các giá trị số dưới dạng từ khóa: ví dụ, credit\_card\_number có thể được khai báo dưới dạng từ khóa để truy cập hiệu quả hơn là dưới dạng kiểu số như long. Không có cách nào chúng ta có thể xây dựng các truy vấn phạm vi trên dữ liệu như vậy. Nguyên tắc chung là nếu các trường số không được sử dụng trong các truy vấn phạm vi, thì nên khai báo chúng dưới dạng kiểu từ khóa vì nó giúp truy xuất nhanh hơn.

**GHI CHÚ**Mã mẫu minh họa kiểu dữ liệu từ khóa có sẵn trong các tệp của cuốn sách.

**TAnh ấy liên tục****\_LOẠI TỪ KHÓA**

Khi tập hợp các tài liệu được mong đợi có cùng giá trị, bất kể số lượng, kiểu constant\_keyword trở nên hữu ích. Giả sử Vương quốc Anh đang tiến hành điều tra dân số vào năm 2031 và vì những lý do hiển nhiên, trường quốc gia của tài liệu điều tra dân số của mỗi công dân sẽ là "United Kingdom" theo mặc định. Không cần phải gửi trường quốc gia cho mỗi tài liệu khi nó được lập chỉ mục vào chỉ mục điều tra dân số. Sơ đồ map-ping định nghĩa một chỉ mục (điều tra dân số) với một trường được gọi là quốc gia có kiểu constant\_keyword.

**Danh sách4.15 điều tra dân sốchỉ số vớitừ khóa hằng số**

điều tra dân số PUT

{

"ánh xạ": { "thuộc tính": {

"quốc gia":{

"type": "constant\_keyword", "value":"Vương quốc Anh"

}

}

}

}

Lưu ý rằng chúng ta đặt giá trị mặc định cho trường này là "United Kingdom" khi khai báo định nghĩa ánh xạ. Bây giờ chúng ta lập chỉ mục một tài liệu cho John Doe, chỉ với tên của anh ấy (không có trường quốc gia):

**124 CPHẦN4*Bản đồ***

PUT điều tra dân số/\_doc/1

{

"tên":"John Doe"

}

Khi chúng tôi tìm kiếm tất cả cư dân Vương quốc Anh, ngay cả khi tài liệu không có trường đó trong quá trình lập chỉ mục, chúng tôi vẫn nhận được kết quả dương tính trả về tài liệu của John:

NHẬN điều tra dân số/\_tìm kiếm

{

"truy vấn": {

"thuật ngữ": {

"quốc gia": {

"giá trị": "Vương quốc Anh"

}

}

}

}

Trường constant\_keyword sẽ có cùng một giá trị cho mọi tài liệu trong chỉ mục đó.

**TKIỂU DỮ LIỆU WILDCARD**

Kiểu dữ liệu ký tự đại diện là một kiểu dữ liệu đặc biệt khác thuộc họ từ khóa. Nó hỗ trợ tìm kiếm dữ liệu bằng ký tự đại diện và biểu thức chính quy. Chúng tôi định nghĩa một trường là kiểu ký tự đại diện bằng cách khai báo trường đó là "type": "wildcard" trong định nghĩa ánh xạ. Sau đó, chúng tôi truy vấn trường bằng cách đưa ra truy vấn ký tự đại diện, như được hiển thị trong danh sách 4.16.

**GHI CHÚ**Một tài liệu với"description":"Ngoại lệ con trỏ Null vì đối tượng là null"đã được lập chỉ mục trước truy vấn này.

**Liệt kê 4.16 Aký tự đại diệntruy vấn với giá trị đại diện**

GET lỗi/\_tìm kiếm

{

"truy vấn": {

"ký tự đại diện": { "mô tả": {

"giá trị": "\*đối tượng\*"

}

**Sử dụng ký tự đại diện**

**truy vấn**

**Tìm kiếm bằng ký tự đại diện**

}

}

}

Các trường từ khóa rất hiệu quả và có hiệu suất cao, do đó việc sử dụng chúng một cách hợp lý sẽ cải thiện hiệu suất lập chỉ mục và truy vấn tìm kiếm.

* + 1. Kiểu dữ liệu ngày tháng

Tìm kiếm đàn hồicung cấp kiểu dữ liệu ngày để hỗ trợ lập chỉ mục và tìm kiếm các hoạt động dựa trên ngày. Các trường ngày được coi là dữ liệu có cấu trúc, do đó chúng ta có thể sử dụng chúng để sắp xếp, lọc và tổng hợp.

***4.5 Dữ liệu cốt lõicác loại* 125**

Elasticsearch phân tích cú pháp giá trị chuỗi và suy ra rằng đó là ngày nếu giá trị tuân thủ tiêu chuẩn ngày ISO 8601. Nghĩa là, giá trị ngày dự kiến ​​sẽ ở định dạngyyyy-MM-dd hoặc (có thành phần thời gian) yyyy-MM-ddTHH:mm:ss.

JSON không có kiểu ngày, vì vậy ngày trong các tài liệu đến được thể hiện dưới dạng chuỗi. Chúng được phân tích cú pháp bởi Elasticsearch và được lập chỉ mục phù hợp. Ví dụ: một giá trị như

"article\_date":"2021-05-01" hoặc "article\_date":"2021-05-01T15:45:50"

được coi là ngày và được lập chỉ mục là loại ngày vì giá trị tuân thủ theo tiêu chuẩn ISO.

Như chúng ta đã làm với các kiểu dữ liệu khác, chúng ta có thể tạo một trường với kiểu ngày trong quá trình định nghĩa ánh xạ. Danh sách tiếp theo tạo một trường ngày khởi hành cho một tài liệu chuyến bay.

**Liệt kê 4.17 Tạo một chỉ mục vớingàykiểu**

Chuyến bay PUT

{

"ánh xạ": { "thuộc tính": {

"ngày\_giờ\_khởi\_hành":{"loại": "ngày"

}

}

}

}

Khi lập chỉ mục một tài liệu chuyến bay, hãy thiết lập"ngày\_khởi\_hành" :"2021-08-06"(hoặc"2021-08-06T05:30:00"(có thành phần thời gian) sẽ lập chỉ mục tài liệu theo ngày như mong đợi.

**GHI CHÚ**Khi không có định nghĩa ánh xạ cho trường ngày trong chỉ mục, Elastic-search sẽ phân tích cú pháp thành công một tài liệu khi định dạng ngày là yyyy-MM-dd (định dạng ngày ISO) hoặc yyyy/MM/dd (định dạng ngày không phải ISO). Tuy nhiên, sau khi chúng tôi tạo định nghĩa ánh xạ cho ngày, định dạng ngày của tài liệu đến được mong đợi dựa trên định dạng được xác định trong định nghĩa ánh xạ.

Chúng ta có thể thay đổi định dạng ngày nếu cần: thay vì đặt ngày theo định dạng ISO (yyyy-MM-dd), chúng ta có thể tùy chỉnh định dạng dựa trên nhu cầu của mình bằng cách đặt định dạng đó trên trường trong quá trình tạo trường:

Chuyến bay PUT

{

"ánh xạ": { "thuộc tính": {

"ngày\_giờ\_khởi\_hành":{"loại": "ngày",

**126 CPHẦN4*Bản đồ***

"định dạng": "dd-MM-yyyy||dd-MM-yy"

}

}

}

}

**Ngày được thiết lập theo một trong hai định dạng này.**

Các tài liệu đến hiện có thể có trường khởi hành được thiết lập là

"ngày\_khởi\_hành" :"06-08-2021"

hoặc

"ngày\_khởi\_hành" :"06-08-21"

Ngoài việc cung cấp ngày dưới dạng giá trị chuỗi, chúng ta có thể cung cấp ngày theo định dạng số—giây hoặc mili giây kể từ kỷ nguyên (ngày 1 tháng 1 năm 1970). Định nghĩa ánh xạ sau đây thiết lập ba ngày với ba định dạng khác nhau:

{

...

"của cải": {

"string\_date":{ "type": "ngày", "format": "dd-MM-yyyy" },

"millis\_date":{ "type": "ngày", "format": "epoch\_millis" }, "seconds\_date":{ "type": "ngày", "format": "epoch\_second"}

}

}

Các ngày được cung cấp sẽ được chuyển đổi nội bộ thành các giá trị dài được lưu trữ tính bằng mili giây kể từ kỷ nguyên, tương đương với epoch\_millis.

Chúng ta có thể sử dụng truy vấn phạm vi để lấy ngày. Ví dụ, đoạn mã sau đây lấy các chuyến bay được lên lịch từ 5:00 đến 5:30 sáng vào một ngày nhất định:

"phạm vi": { "ngày\_khởi\_hành": {

"gte": "2021-08-06T05:00:00",

"lte": "2021-08-06T05:30:00"

}

**Một truy vấn phạm vi tìm nạp tài liệu giữa hai ngày**

**Khoảng thời gian là từ 5:00 đến 5:30 sáng**

}

Cuối cùng, chúng ta có thể chấp nhận nhiều định dạng ngày tháng trên một trường bằng cách khai báo các định dạng bắt buộc:

"ngày\_khởi\_hành":{ "loại": "ngày",

"định dạng": "dd-MM-yyyy||dd/MM/yyyy||yyyy-MM-dd||yyyy/MM/dd"

**Thiết lập bốn định dạng khác nhau trên sân**

}

Tham khảo tài liệu về kiểu dữ liệu ngày của Elasticsearch để biết thêm thông tin, vì chúng tôi không thể đề cập đến tất cả các tùy chọn trong chương này. Ngoài ra, hãy tham khảo các tệp của cuốn sách để biết ví dụ đầy đủ.

***4.5 Dữ liệu cốt lõicác loại* 127**

* + 1. Kiểu dữ liệu số

Elasticsearch cung cấp một số kiểu dữ liệu số để xử lý dữ liệu số nguyên và số thực. Bảng 4.2 liệt kê các kiểu số.

**Bảng 4.2 Kiểu dữ liệu số**

|  |  |  |
| --- | --- | --- |
| Kiểu số nguyên | byte ngắn số nguyên dài  không dấu\_dài | Số nguyên 8 bit có dấu Số nguyên 16 bit có dấu Số nguyên 32 bit có dấu Số nguyên 64 bit có dấu  Số nguyên không dấu 64 bit |
| Các kiểu số dấu phẩy động | float đôi nửa\_float  tỷ lệ nổi | Số dấu phẩy động độ chính xác đơn 32 bit Số dấu phẩy động độ chính xác kép 64 bit Số dấu phẩy động độ chính xác nửa 16 bit  Số dấu phẩy động được hỗ trợ bởidài |

Chúng tôi khai báo trường và kiểu dữ liệu của nó là "field\_name":{ "type": "short"}. Đoạn mã sau đây minh họa cách chúng ta có thể tạo lược đồ ánh xạ với các trường số:

"tuổi":{

"loại": "ngắn"

},

"cấp":{

"kiểu": "nửa\_float"

},

"roll\_number":{ "type": "dài"

}

Ví dụ này tạo ra ba trường với ba kiểu dữ liệu số khác nhau.

* + 1. Kiểu dữ liệu boolean

Kiểu dữ liệu boolean biểu diễn giá trị nhị phân của một trường: true hoặc false. Ví dụ này khai báo kiểu của một trường là boolean:

ĐẶT bom tấn

{

"ánh xạ": { "thuộc tính": {

"bom tấn":{ "loại": "boolean"

}

}

}

}

**128 CPHẦN4*Bản đồ***

Sau đó, chúng ta có thể lập chỉ mục một vài bộ phim—Avatar (2009) là phim bom tấn và The Matrix Resurrections (2021) là phim thất bại:

PUT phim bom tấn/\_doc/1

{

"title":"Avatar", "bom tấn":đúng

}

PUT phim bom tấn/\_doc/2

{

"title":"Ma Trận Phục Sinh", "blockbuster":"false"

}

Ngoài việc thiết lập trường là kiểu boolean của JSON (true hoặc false), trường này cũng chấp nhận các giá trị Boolean "được chuỗi hóa" như "true" và "false", như thể hiện trong ví dụ thứ hai.

Chúng ta có thể sử dụng truy vấn thuật ngữ (Boolean được phân loại là dữ liệu có cấu trúc) để lấy kết quả. Ví dụ, truy vấn sau đây lấy Avatar dưới dạng phim bom tấn:

NHẬN bom tấn/\_tìm kiếm

{

"truy vấn": {

"term": { "bom tấn": {

"giá trị": "đúng"

}

}

}

}

Chúng tôi cũng có thể cung cấp một chuỗi rỗng cho mộtSAIgiá trị:"bom tấn":"".

* + 1. Các kiểu dữ liệu phạm vi

Các loại dữ liệu phạm vi biểu thị giới hạn dưới và trên của một trường. Ví dụ, nếu chúng ta muốn chọn một nhóm tình nguyện viên cho thử nghiệm vắc-xin, chúng ta có thể phân loại tình nguyện viên dựa trên các danh mục như độ tuổi 25–50 hoặc 51–70, thông tin nhân khẩu học như mức thu nhập,cư dân thành phố, v.v. Elasticsearch cung cấp một kiểu dữ liệu phạm vi để hỗ trợ các truy vấn tìm kiếm trên dữ liệu phạm vi. Phạm vi được xác định bởi các toán tử như lte (nhỏ hơn hoặc bằng) và lt (nhỏ hơn) cho các giới hạn trên và gte (lớn hơn hoặc bằng) và gt (lớn hơn) cho các giới hạn dưới.

Elasticsearch cung cấp nhiều kiểu dữ liệu phạm vi khác nhau:phạm vi ngày tháng,dãy số nguyên,phạm vi float,phạm vi ipvà nhiều hơn nữa. Trong phần tiếp theo, chúng ta sẽ thấyphạm vi ngày thángnhập vào hành động.

**TAnh ấy hẹn hò****\_VÍ DỤ LOẠI PHẠM VI**

Kiểu date\_range giúp lập chỉ mục một phạm vi ngày cho một trường. Sau đó, chúng ta có thể sử dụng

phạm vi truy vấn để khớp với tiêu chídựa trên giới hạn trên và dưới của ngày.

* 1. ***Dữ liệu cốt lõicác loại* 129**

Hãy mã hóa một ví dụđể chứng minh kiểu date\_range. Venkat Subramaniam là một tác giả từng đoạt giải thưởng, người cung cấp các buổi đào tạo về nhiều chủ đề khác nhau, từ lập trình đến thiết kế đến thử nghiệm. Hãy cùng xem xét danh sách các khóa đào tạo của ông vàngày cho ví dụ của chúng tôi. Chúng tôi tạo một chỉ mục đào tạo với hai trường—tên khóa học và ngày đào tạo—có kiểu text và date\_range tương ứng.

**Liệt kê 4.18 Chỉ mục với mộtphạm vi ngày thángkiểu**

Đào tạo PUT

{

"ánh xạ": { "thuộc tính":



{

"tên":{

"loại": "văn bản"

},

**Tên của buổi đào tạo**

"ngày\_đào\_tạo":{ "kiểu": "phạm\_hạn\_ngày"

}

}

}

}

**Trường training\_dates được khai báo là kiểu date\_range.**

Bây giờ mục lục đã sẵn sàng, chúng ta hãy lập mục lục một số tài liệu có chứa các khóa đào tạo và ngày đào tạo của Venkat:

Đào tạo PUT/\_doc/1

{

"name":"Lập trình hàm trong Java", "training\_dates":{

"gte":"2021-08-07",

"lte":"2021-08-10"

}

}

Đào tạo PUT/\_doc/2

{

"name":"Lập trình Kotlin", "training\_dates":{

"gte":"2021-08-09",

"lte":"2021-08-12"

}

}

Đào tạo PUT/\_doc/3

{

"name":"Lập trình phản ứng", "training\_dates":{

"gte":"2021-08-17",

"lte":"2021-08-20"

}

}

**130 CPHẦN4*Bản đồ***

Trường kiểu date\_range mong đợi hai giá trị: một giới hạn trên và một giới hạn dưới. Chúng thường được biểu thị bằng các chữ viết tắt như gte (lớn hơn hoặc bằng), lt (nhỏ hơn), v.v.

Sau khi chuẩn bị dữ liệu, hãy đưa ra yêu cầu tìm kiếm để tìm các khóa học của Venkat giữa hai ngày.

**Liệt kê 4.19 Tìm kiếm các khóa học giữa haingày tháng**

GET đào tạo/\_tìm kiếm

{

"truy vấn": {

"phạm vi": {

**Chúng tôi sử dụng một phạm vi**

**truy vấn để tìm kiếm.**

"ngày\_đào\_tạo": { "gt": "2021-08-10",

"lt": "2021-08-12"

**Tìm kiếm các khóa học giữa hai ngày này**

}

}

}

}

Để trả lời cho truy vấn (kết quả bị bỏ qua vì lý do ngắn gọn), chúng tôi biết rằng Venkat đang cung cấp khóa đào tạo “Lập trình Kotlin” giữa hai ngày (tài liệu thứ haiment khớp với ngày). Date\_range giúp dễ dàng tìm kiếm một phạm vi dữ liệu. Ngoài date\_range, chúng ta có thể tạo các phạm vi khác bao gồm ip\_range, float\_range, double\_range, integer\_range, v.v.

* + 1. Kiểu dữ liệu địa chỉ IP (ip)

Elasticsearch cung cấp một kiểu dữ liệu cụ thể để hỗ trợ địa chỉ giao thức internet (IP): kiểu dữ liệu ip. Kiểu dữ liệu này hỗ trợ cả địa chỉ IP IPv4 và IPv6. Để tạo mộttrường loại ip, sử dụng "field":{"type": "ip"} như ví dụ sau đây cho thấy:

Mạng PUT

{

"ánh xạ": { "thuộc tính": {

"router\_ip":{ "kiểu": "ip" }

}

**Kiểu dữ liệu của trường là ip.**

}

}

Việc lập chỉ mục tài liệu rất đơn giản:

Mạng PUT/\_doc/1

{

"router\_ip":"35.177.57.111"

**Lập chỉ mục một tài liệu có địa chỉ IP**

}

Cuối cùng, chúng ta có thể sử dụng điểm cuối tìm kiếm của mình để tìm kiếm các địa chỉ IP khớp với truy vấn của chúng ta. Truy vấn sau đây tìm kiếm dữ liệu trong chỉ mục mạng để có được địa chỉ IP khớp:

* 1. ***Dữ liệu nâng caocác loại* 131**

NHẬN mạng lưới/\_tìm kiếm

{

"truy vấn":{

"thuật ngữ": {

**Tìm kiếm theo cấp độ thuật ngữ**

**cho địa chỉ IP**

"router\_ip": { "giá trị": "35.177.0.0/16" }

}

}

}

**Tìm kiếm địa chỉ IPtrong phạm vi này**

Các phần trước đã xem xét các kiểu dữ liệu cốt lõi. Elasticsearch cung cấp một tập hợp phong phú các kiểu dữ liệu cho hầu hết mọi trường hợp sử dụng mà chúng ta có thể nghĩ đến. Một số kiểu dữ liệu cốt lõi dễ hiểu và trực quan để làm việc, trong khi những kiểu khác, như đối tượng, lồng nhau,join, completion và search\_as\_you\_type cần được chú ý đặc biệt. Trong phần tiếp theo, chúng ta sẽ khám phá một số kiểu dữ liệu nâng cao này.

#### Kiểu dữ liệu nâng cao

Chúng ta đã xem xét các kiểu dữ liệu cốt lõi và phổ biến để biểu diễn các trường dữ liệu. Một số có thể được phân loại là các kiểu nâng cao, bao gồm một số kiểu chuyên biệt. Trong một vài phần tiếp theo, chúng ta sẽ đề cập đến các kiểu bổ sung này với các định nghĩa và ví dụ.

**GHI CHÚ** Bao gồm tất cả các kiểu dữ liệu sẽ làm cho cuốn sách này trở nên cồng kềnh, và tôi cũng không thích những chương quá dài. Vì vậy, tôi đã đưa ra quyết định sáng suốt là chỉ đưa vào những kiểu dữ liệu nâng cao quan trọng và hữu ích nhất. Kho lưu trữ GitHub và trang web của cuốn sách bao gồm các ví dụ về các kiểu được trích dẫn ở đây cũng như những kiểu bị bỏ qua.

* + 1. Kiểu dữ liệu geo\_point

Với sự ra đời của điện thoại thông minh và các thiết bị, dịch vụ định vị và tìm kiếm địa điểm gần đó đã trở nên phổ biến. Hầu hết chúng ta đã sử dụng thiết bị thông minh để làm những việc như tìm vị trí của nhà hàng gần nhất hoặc yêu cầu chỉ đường GPS đến nhà mẹ vợ vào dịp Giáng sinh. Elasticsearch có một kiểu dữ liệu chuyên biệt để nắm bắt vị trí của một địa điểm.

Dữ liệu vị trí được thể hiện nhưkiểu dữ liệu geo\_point biểu diễn kinh độ và vĩ độ. Chúng ta có thể sử dụng dữ liệu này để xác định địa chỉ của nhà hàng, trường học, sân golf, v.v. Hãy cùng xem nó hoạt động như thế nào.

Đoạn mã sau đây hiển thị định nghĩa lược đồ của mộtnhà hàngmục lục: nhà hàng vớitênVàĐịa chỉcác lĩnh vực. Điểm đáng chú ý duy nhất làĐịa chỉtrường được định nghĩa là mộtđiểm địa lýkiểu dữ liệu.

**Liệt kê 4.20 Sơ đồ ánh xạ với một trường được khai báo làđiểm địa lýkiểu**

Nhà hàng PUT

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

**132 CPHẦN4*Bản đồ***

},

"Địa chỉ":{

"kiểu": "điểm\_địa\_phương"

}

}

}

}

Hãy lập chỉ mục cho một nhà hàng mẫu (Sticky Fingers hư cấu có trụ sở tại London) với địa chỉ được cung cấp dưới dạng kinh độ và vĩ độ.

**Danh sách 4.21 Địa chỉ được biểu diễn bằng kinh độ và vĩ độ**

PUT nhà hàng/\_doc/1

{

"tên":"Ngón tay dính", "địa chỉ":{

"lớn":"0,1278",

"vĩ độ":"51.5074"

**Địa chỉ được cung cấp nhưmột cặp giá trị kinh độ và vĩ độ.**

}

}

Địa chỉ của nhà hàng được đưa ra dưới dạng kinh độ (cặp kinh độ) và vĩ độ (vĩ độ). Có những cách khác để cung cấp các thông tin đầu vào này, như chúng ta sẽ thấy ngay sau đây; nhưng trước tiên, hãy tìm các nhà hàng trong phạm vi vị trí.

Chúng ta có thể lấy các nhà hàng bằng cách sử dụng geo\_bounding\_boxtruy vấn để tìm kiếm dữ liệu liên quan đến địa chỉ địa lý. Truy vấn lấy dữ liệu đầu vào là các điểm địa lý top\_left và bottom\_right (được cung cấp dưới dạng cặp kinh độ và vĩ độ) để tạo vùng hộp xung quanh điểm quan tâm của chúng tôi, như thể hiện trong hình 4.8.



**Hình 4.8 Hộp giới hạn địa lý xung quanh một vị trí ở trung tâm London**

***4.6 Dữ liệu nâng caocác loại* 133**

**Liệt kê 4.22 Tìm kiếm nhà hàng xung quanh một vị trí địa lý**

NHẬN nhà hàng/\_tìm kiếm

{

"truy vấn": { "hộp giới hạn địa lý":{

"Địa chỉ":{

"trên cùng bên trái":{ "lon":"0",

"vĩ độ":"52"

},

"bottom\_right":{ "lon":"1",

"vĩ độ":"50"

**Trên cùng bên tráigóc hộp**

**Dưới cùng bên phảigóc hộp**

}

}

}

}

}

Truy vấn này sẽ lấy nhà hàng Sticky Fingers của chúng tôi vì nó nằm trong hộp giới hạn địa lý được biểu thị bằng hai điểm địa lý.

**GHI CHÚ**Khi tìm kiếm địa chỉ bằng geo\_bounding\_box, một lỗi thường gặp là cung cấp thông tin đầu vào không chính xác (top\_left và bottom\_right) cho truy vấn. Đảm bảo kinh độ và vĩ độ của hai thông tin đầu vào này tạo thành bounding box.

Trước đó, tôi đã đề cập rằng chúng ta có thể cung cấp thông tin vị trí ở nhiều định dạng khác nhau, không chỉ kinh độ và vĩ độ, bao gồm cả dạng mảng hoặc chuỗi. Bảng 4.3 liệt kê các định dạng này.

**Bảng 4.3 Định dạng liên quan đến thông tin vị trí**

|  |  |  |
| --- | --- | --- |
| **Định dạng** | **Giải thích** | **Ví dụ** |
| Mảng | Geopoint đại diệndưới dạng một mảng. Lưu ý thứ tự của các đầu vào geopoint: nó lấy kinh độ và vĩ độ, không phải ngược lại (không giống như định dạng chuỗi—xem hàng tiếp theo). | "địa chỉ":[0.12,51.5] |
| Sợi dây | Geopoint dưới dạng dữ liệu chuỗi với đầu vào là vĩ độ và kinh độ. | "địa chỉ": "51.5,0.12" |
| Geohash | Một chuỗi được mã hóa được hình thành bằng cách băm tọa độ kinh độ và vĩ độ. Chuỗi chữ số chỉ đến một địa điểm trên trái đất. | u10j4 |
| Điểm | Vị trí chính xác trên bản đồ. Được gọi là văn bản nổi tiếng (WKT), một cơ chế chuẩn để biểu diễn dữ liệu hình học. | ĐIỂM(51.5,-0.12) |

Trong phần này, chúng ta đã làm việc với các truy vấn địa lý mà không có kiến ​​thức trước về chúng. Chúng tôi sẽ đề cập chi tiết về chủ đề này trong chương 12.

**134 CPHẦN4*Bản đồ***

* + 1. Kiểu dữ liệu đối tượng

Thông thường, dữ liệu có tính phân cấp: ví dụ, một đối tượng email bao gồm các trường cấp cao nhấtgiống như chủ thể cũng như đối tượng bên trong để chứa tệp đính kèm, lần lượt có thể có các thuộc tính như tên tệp đính kèm, loại tệp đính kèm, v.v. JSON cho phép chúng ta tạo các đối tượng phân cấp—các đối tượng được bao bọc trong các đối tượng khác. Elasticsearch có một kiểu dữ liệu đặc biệt để biểu diễn phân cấp các đối tượng: kiểu đối tượng.

Các kiểu dữ liệu cho cấp cao nhấtchủ thểVàĐẾNcác cánh đồng làchữVàtừ khóa,tương ứng. Bởi vì tệp đính kèm là một đối tượng, nên kiểu dữ liệu của nó làsự vật. Haicủa cảitên tập tinVàkiểu tập tintrongtệp đính kèmđối tượng có thể được mô hình hóa nhưchữtrường. Với thông tin này, chúng ta có thể tạo định nghĩa ánh xạ.

**Liệt kê 4.23 Định nghĩa lược đồ với các kiểu dữ liệu đối tượng**

ĐẶT email

{

"ánh xạ": { "thuộc tính": {

"ĐẾN":{

"loại": "từ khóa"

},

"chủ thể":{

"loại": "văn bản"

},

"tệp đính kèm":{ "thuộc tính": {

"tên tệp":{

"loại":"văn bản"

**Thuộc tính cấp cao nhất**

**cho mục lục email**

**Đối tượng bên trong bao gồm các thuộc tính cấp hai**

},

"kiểu tệp":{

"loại":"văn bản"

}

}

}

}

}

}

Thuộc tính attachments là thứ chúng ta cần chú ý. Kiểu của trường này là object vì nó bao gồm hai trường khác. Hai trường được định nghĩa trong đối tượng bên trong attachments không khác gì subject và các trường được khai báo ở cấp cao nhất, ngoại trừ việc chúng ở cấp thấp hơn một cấp.

Sau khi lệnh được thực thi thành công, chúng ta có thể kiểm tra lược đồ bằng cách gọi lệnh GET emails/\_mapping.

**Liệt kê 4.24NHẬN email/\_bản đồphản ứng**

{

"email" : {

"ánh xạ" : {

***4.6 Dữ liệu nâng caocác loại* 135**

"thuộc tính" : { "tệp đính kèm" : {

**Loại bị ẩn**

**tập tin đính kèm là một bên trongđối tượng với các trường khác.**

**Của lĩnh vựcloại được hiển thị như mong đợi.**

}

"của cải" : {

"tên tệp" : { "kiểu" : "văn bản",

. . .

**ở đây, nhưng theo mặc định thì nó là một đối tượng.**

Phản hồi bao gồm chủ đề, đến và tệp đính kèm dưới dạng các trường cấp cao nhất (không phải tất cả các thuộc tính đều được hiển thị, vì lý do ngắn gọn). Đối tượng tệp đính kèm có các trường khác được đóng gói dưới dạng các thuộc tính với các trường thích hợp và định nghĩa của chúng. Khi chúng tôi lấy bản đồ- ping (GET emails/\_mapping), trong khi tất cả các trường khác hiển thị các kiểu dữ liệu liên quan của chúng, tệp đính kèm thì không: theo mặc định, Elasticsearch suy ra kiểu đối tượng cho các đối tượng bên trong.

Tiếp theo, chúng ta hãy lập chỉ mục cho một tài liệu.

**Liệt kê 4.25 Lập chỉ mục một tài liệu email**

ĐẶT email/\_doc/1

{

"ĐẾN:":[" johndoe@johndoe.com ",](mailto:johndoe@johndoe.com) "subject":"Loại đối tượng thử nghiệm", "attachments":{

"tên tệp":"file1.txt", "kiểu tệp":"bí mật"

}

}

Bây giờ chúng ta đã chuẩn bị chỉ mục email của mình bằng một tài liệu, chúng ta có thể đưa ra truy vấn tìm kiếm đơn giản trên các trường đối tượng bên trong để lấy các tài liệu có liên quan (và chứng minh quan điểm của mình).

**Liệt kê 4.26 Tìm kiếm email có tệp đính kèm**

NHẬN email/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tệp đính kèm.tên tệp": "tệp1.txt"

}

}

}

Truy vấn này trả về tài liệu từ kho lưu trữ của chúng tôi vì tên tệp khớp vớitài liệu. Lưu ý rằng chúng tôi sử dụng truy vấn thuật ngữ trên trường từ khóa vì chúng tôi muốn khớp với giá trị trường chính xác (file1.txt).

Mặc dù các kiểu đối tượng khá đơn giản, chúng có một hạn chế: các đối tượng bên trong được làm phẳng và không được lưu trữ dưới dạng các tài liệu riêng lẻ. Nhược điểm của hành động này là mối quan hệ bị mất giữa các đối tượng được lập chỉ mục từ một mảng. Hãy cùng xem xét hạn chế này bằng một ví dụ chi tiết.

**136 CPHẦN4*Bản đồ***

**L****GIẢ LẬP LOẠI ĐỐI TƯỢNG**

Trong ví dụ email trước đó của chúng tôi, trường tệp đính kèm được khai báo là một kiểu đối tượng. Mặc dù chúng tôi đã tạo một email chỉ với một đối tượng tệp đính kèm, nhưng không có gì ngăn cản chúng tôi tạo nhiều tệp đính kèm (email thường có nhiều tệp đính kèm), như được hiển thị trong danh sách tiếp theo.

**Liệt kê 4.27 Lập chỉ mục một tài liệu có nhiều tệp đính kèm**

ĐẶT email/\_doc/2

{

"ĐẾN:":[" bà.doe@johndoe.com ",](mailto:mrs.doe@johndoe.com) "subject":"Kiểm tra nhiều tệp đính kèm", "attachments":[{

"tên tệp":"file2.txt", "kiểu tệp":"bí mật"

},{

"tên tệp":"file3.txt", "kiểu tệp":"riêng tư"

}]

}

Theo mặc định, trường tệp đính kèm là loại đối tượng: một đối tượng bên trong bao gồm một mảng các tệp đính kèm. Lưu ý rằng loại phân loại của tệp file2.txt là bí mật và của tệp file3.txt là riêng tư (xem bảng 4.4).

**Bảng 4.4 Tên tệp đính kèm và các loại được phân loại**

|  |  |
| --- | --- |
| **Tên tệp đính kèm** | **Loại tập tin** |
| file2.txt  file3.txt | bảo mật  riêng tư |

Tài liệu email của chúng tôi được lập chỉ mục và có ID 1 và một vài tệp đính kèm. Chúng ta hãy thực hiện yêu cầu tìm kiếm đơn giản: tìm kiếm các tài liệu phù hợp với tên tệp file2.txt và loại tệp private. Xem dữ liệu trong bảng 4.4, truy vấn này sẽ không trả về kết quả nào vì phân loại của file2.txt là bí mật, không phải private. Chúng ta hãy truy vấn và kiểm tra kết quả.

Để làm như vậy, chúng ta cần sử dụng một truy vấn nâng cao được gọi là truy vấn hợp chất kết hợp nhiều truy vấn lá khác nhau để tạo ra một truy vấn phức tạp. Một truy vấn hợp chất như vậy là booltruy vấn tìm kiếm. Không đi sâu vào chi tiết về cách truy vấn bool được xây dựng, chúng ta hãy xem xét nó trong hành động. Chúng ta viết truy vấn bool của mình với hai mệnh đề truy vấn khác:

* MỘTphảiđiều khoản kiểm tra tất cả các tài liệu khớp với tên tệp đính kèm bằng cách sử dụngthuật ngữtruy vấn
* Một giâyphảimệnh đề kiểm tra xem phân loại tập tin córiêng tư

Truy vấn được đưa ra trong danh sách sau.

***4.6 Dữ liệu nâng caocác loại* 137**

**Liệt kê 4.28 Nâng caobooltruy vấn vớithuật ngữtruy vấn**

**Xác định truy vấn**

NHẬN email/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [

**Một truy vấn bool tìm kiếm sự khớp với tên tệp và loại tệp**

**mệnh đề phải xác địnhcác điều khoản bắt buộc**

**như một bool**

**truy vấn**

{"term": { "attachments.filename.keyword": "file2.txt"}},

{"term": { "attachments.filetype.keyword": "riêng tư" }}

]

}

}

}

Khi truy vấn này được thực thi, nó sẽ trả về tài liệu sau:

"lượt truy cập" : [[

{

...

"\_nguồn" : {

"ĐẾN:" :[" bà.doe@johndoe.com "](mailto:mrs.doe@johndoe.com), "chủ đề": "Kiểm tra nhiều tệp đính kèm", "tệp đính kèm": [

{

"tên tệp": "file2.txt", "kiểu tệp": "bí mật"

},

..

]

}

}]

Thật không may, kết quả này là không chính xác—không có tài liệu nào tồn tại với sự kết hợp củatên file2.txt và kiểu private (kiểm tra lại bảng 4.4). Đây là nơi kiểu dữ liệu đối tượng bị hỏng—nó không thể tôn trọng mối quan hệ giữa các đối tượng bên trong.

Lý tưởng nhất là các giá trị file2.txt và private nằm trong các đối tượng khác nhau, do đó tìm kiếm không nên coi chúng là một thực thể duy nhất. Lý do là các đối tượng bên trong không được lưu trữ dưới dạng các tài liệu riêng lẻ—chúng được làm phẳng:

{

...

"attachments.filename" :["file1.txt","file2.txt","file3.txt"] "attachments.filetype":["private","confidential"]

}

Như bạn có thể thấy,tên tập tins được thu thập dưới dạng một mảng và được lưu trữ trongtệp đính kèm

.filename, vì vậy chúng cũng là các loại tệp. Thật không may, vì chúng được lưu trữ theo cách này nên mối quan hệ của chúng bị mất. Chúng ta không thể nói file1.txt là riêng tư hay bí mật vì mảng không chứa trạng thái.

Đây là hạn chế của việc lập chỉ mục một mảng các đối tượng cho một trường và cố gắng tìm kiếm các đối tượng như các tài liệu riêng lẻ. Tin tốt là một kiểu dữ liệu được gọi là nested giải quyết được vấn đề này, như chúng ta sẽ thảo luận trong phần tiếp theo.

**138 CPHẦN4*Bản đồ***

* + 1. Kiểu dữ liệu lồng nhau

Ví dụ trước của chúng tôi cho thấy truy vấn tìm kiếm không tôn trọng tính toàn vẹn của từng tài liệu. Chúng tôi có thể khắc phục vấn đề này bằng cách giới thiệu kiểu dữ liệu lồng nhau. Kiểu dữ liệu lồng nhaukiểu là một dạng chuyên biệt của kiểu đối tượng duy trì mối quan hệ giữa các mảng đối tượng trong một tài liệu.

Tiếp tục ví dụ về email và tệp đính kèm, chúng ta hãy định nghĩa tệp đính kèmtrường là kiểu dữ liệu lồng nhau thay vì để Elasticsearch suy ra nó như một kiểu đối tượng. Điều này đòi hỏi phải tạo một lược đồ bằng cách khai báo trường tệp đính kèm là một kiểu dữ liệu lồng nhau.

**Liệt kê 4.29 Ánh xạ định nghĩa lược đồ cholồng nhaukiểu**

ĐẶT email\_lồng nhau

{

"ánh xạ": { "thuộc tính": {

"tệp đính kèm": { "kiểu": "lồng nhau", "thuộc tính": {

"tên tệp": {

"loại": "từ khóa"

**Khai báo trường tệp đính kèm là một kiểu lồng nhau**

**Trường tên tệp được khai báo là loại từ khóa để tránh việc phân tách.**

},

"kiểu tệp": {

"loại": "văn bản"

}

}

}

}

}

}

**Chúng ta có thể để trường này dưới dạng văn bản.**

Ngoài việc tạo ratệp đính kèmlĩnh vực như mộtlồng nhauloại, chúng tôi đã tuyên bốtên tập tinnhư mộttừ khóaloại. Giá trị của trường này được mã hóa: ví dụ,file1.txtđược chia thànhtập tin1Vàvăn bản. Kết quả là, một truy vấn tìm kiếm có thể được khớp vớivăn bảnVàbảo mậthoặcvăn bảnVàriêng tư, vì cả hai bản ghi đều cóvăn bảnnhư một mã thông báo chung. Để tránh điều này, chúng ta có thể sử dụngtên tập tinnhư mộttừ khóatrường. Bạn cũng có thể thấy phương pháp này trong danh sách 4.28, trong đó chúng tôi sử dụngtệp đính kèm.tên tệp.từ khóatrong chúng tôitìm kiếmtruy vấn.

Quay lại vấn đề trước mắt: chúng ta có một định nghĩa lược đồ, vì vậy tất cả những gì chúng ta cần làm là lập chỉ mục cho một tài liệu.

**Liệt kê 4.30 Lập chỉ mục một tài liệu vớilồng nhautài sản**

ĐẶT email\_nested/\_doc/1

{

"tệp đính kèm" : [

{

**Cung cấp một vàicác đối tượng như là phần đính kèm**

"tên tệp": "file1.txt", "kiểu tệp": "bí mật"

},

***4.6 Dữ liệu nâng caocác loại* 139**

{

"tên tệp" : "file2.txt", "kiểu tệp" : "riêng tư"

}

]

}

Sau khi tài liệu này được lập chỉ mục thành công, phần cuối cùng của trò chơi ghép hình là tìm kiếm. Liệt kê 4.31 hiển thị truy vấn tìm kiếm được viết để lấy tài liệu. Các tiêu chí làemail có tệp đính kèm có file1.txt và private là tên tệp và loại phân loại tương ứng. Sự kết hợp này không tồn tại, do đó kết quả phải để trống, không giống như trường hợp đối tượng mà dữ liệu được tìm kiếm trên các tài liệu, dẫn đến kết quả dương tính giả.

**Liệt kê 4.31 Lấy kết quả khớp với tên tệp vàphân loại**

NHẬN email\_nested/\_search

{

"truy vấn": {

"lồng nhau": {

**Một truy vấn lồng nhau để lấy**

**dữ liệu từ các trường lồng nhau**

"đường dẫn": "tệp đính kèm", "truy vấn": {

"bool": {

"phải": [

**Con đường chỉ đến**

**tên của lồng nhaucánh đồng. Các mệnh đề tìm kiếm: phải**

**khớp file1.txt và riêng tư**

{ "trận đấu": { "tệp đính kèm.tên tệp": "file1.txt" }},

{ "match": { "attachments.filetype": "riêng tư" }}

]

}

}

}

}

}

Truy vấn này tìm kiếm một tệp có tên file1.txt với phân loại riêng tư, không tồn tại. Không có tài liệu nào được trả về cho truy vấn này, đây chính xác là những gì chúng tôi mong đợi. Phân loại của file1.txt là bí mật, không riêng tư, vì vậy nó không khớp. Khi một kiểu lồng nhau biểu diễn một mảng các đối tượng bên trong, đối tượng riêng lẻ được lưu trữ và lập chỉ mục dưới dạng một tài liệu ẩn.

Kiểu dữ liệu lồng nhau rất tốt trong việc tôn trọng các mối liên kết và mối quan hệ. Nếu bạn cần tạo một mảng các đối tượng, mỗi đối tượng phải được coi là một cá thểđối tượng, kiểu dữ liệu lồng nhau có thể trở thành bạn của bạn.

**Elasticsearch không có kiểu mảng**

Trong khi chúng ta đang nói về chủ đề mảng, điều thú vị là không cómảngkiểu dữ liệu trongElasticsearch. Tuy nhiên, chúng ta có thể thiết lập bất kỳ trường nào có nhiều hơn một giá trị, do đó biểu diễn trường đó dưới dạng một mảng. Ví dụ, một tài liệu có một trường tên có thể được thay đổi từ một giá trị duy nhất thành một mảng chỉ bằng cách thêm danh sách các giá trị dữ liệu vào trường:"tên": "John Doe" đến "tên": ["John Smith", "John Doe"].

**140 CPHẦN4*Bản đồ***

***(tiếp theo)***

Elasticsearch suy ra kiểu dữ liệu như thế nào khi chúng ta cung cấp các giá trị dưới dạng một mảngtrong quá trình lập bản đồ động?kiểu dữ liệu được lấy từ kiểu của phần tử đầu tiên trong mảng. Ví dụ,"John Smith"là một chuỗi, vì vậy tên là mộtchữgõ mặc dùbiểu diễn của nó dưới dạng một mảng.

Một điểm quan trọng cần lưu ý khi tạo mảng là chúng ta không thể trộn các kiểu khác nhau trong một mảng. Ví dụ, chúng ta không thể khai báo trường tên như thế này:"tên": ["John Smith", 13, "Neverland"]. Điều này là bất hợp pháp vì trường này bao gồm nhiều loại.

* + 1. Kiểu dữ liệu phẳng

Cho đến nay, chúng ta đã xem xét các trường lập chỉ mục được phân tích cú pháp từ một tài liệu JSON. Mỗi trường được coi là riêng lẻ và độc lập khi được phân tích và lưu trữ. Tuy nhiên, đôi khi chúng ta có thể không cần lập chỉ mục tất cả các trường con thành các trường riêng lẻ, do đó tránh được quá trình phân tích tốn kém. Hãy nghĩ đến một luồng tin nhắn trò chuyện trên hệ thống trò chuyện, bình luận đang chạy trong một trận bóng đá trực tiếp hoặc một bác sĩ ghi chú về bệnh tật của bệnh nhân. Chúng ta có thể tải loại dữ liệu này dưới dạng một blob lớn thay vì khai báo từng trường một cách rõ ràng (hoặc suy ra nó một cách động). Elasticsearch cung cấp một dữ liệu đặc biệtloại được gọi là phẳng cho mục đích này.

Một trường phẳng chứa thông tin dưới dạng một hoặc nhiều trường con và giá trị của mỗi trường con được lập chỉ mục dưới dạng từ khóa. Không có giá trị nào được coi là trường văn bản và do đó chúng không trải qua quá trình phân tích văn bản.

Hãy xem xét một ví dụ về một bác sĩ ghi chép liên tục về bệnh nhân của mình trong quá trình tham vấn. Bản đồ bao gồm hai trường: tên bệnh nhân và ghi chú của bác sĩ. Điểm chính trong bản đồ này là khai báo trường doctor\_notes là một kiểu phẳng.

**Liệt kê 4.32 Tạo bản đồ vớiphẳngkiểu dữ liệu**

Tư vấn PUT

{

"ánh xạ": { "thuộc tính": {

"tên\_bệnh\_nhân":{ "loại": "văn\_bản"

},

"doctor\_notes":{ "type": "làm phẳng"

}

**Trường này có thể bao gồm nhiều trường con khác nhau.**

**Trường được tuyên bố là phẳng.**

}

}

}

Ý tưởng là bất kỳ trường nào (và các trường con của nó) được khai báo là phẳng sẽ không được phân tích; tất cả các giá trị được lập chỉ mục dưới dạng từ khóa. Hãy tạo một tài liệu tham vấn bệnh nhân và lập chỉ mục cho nó.

***4.6 Dữ liệu nâng caocác loại* 141**

**Liệt kê 4.33 Lập chỉ mục tài liệu tham vấn với ghi chú của bác sĩ**

Tư vấn PUT/\_doc/1

{

"patient\_name":"John Doe", "doctor\_notes":{

"nhiệt độ":103,

**Cánh đồng phẳng có thể giữ được**

**bất kỳ số lượng trường con nào.**

"triệu chứng":["ớn lạnh","sốt","đau đầu"], "tiền sử":"không có", "thuốc":["Kháng sinh","Paracetamol"]

}

}

**Tất cả các trường này đều được lập chỉ mục dưới dạng từ khóa.**

Như bạn có thể thấy, doctor\_notes chứa rất nhiều thông tin—nhưng hãy nhớ rằng chúng ta không tạo các trường bên trong này trong định nghĩa ánh xạ của mình. Vì doctor\_notes là một kiểu dữ liệu phẳng, tất cả các giá trị đều được lập chỉ mục như là từ khóa.

Cuối cùng, chúng ta có thể tìm kiếm trong chỉ mục bằng bất kỳ từ khóa nào trong ghi chú của bác sĩ.

**Liệt kê 4.34 Tìm kiếmphẳngtrường kiểu dữ liệu**

NHẬN tư vấn/\_tìm kiếm

{

}

"truy vấn": {

"cuộc thi đấu": {

"doctor\_notes": "Paracetamol"

}

}

**Tìm kiếm thuốc của bệnh nhân**

Tìm kiếm "Paracetamol" sẽ trả về tài liệu tham vấn của John Doe. Bạn có thể thử nghiệm bằng cách thay đổi truy vấn khớp thành bất kỳ trường nào—ví dụ: doctor\_ notes:chills—hoặc viết truy vấn phức tạp như truy vấn được hiển thị bên dưới.

**Liệt kê 4.35 Truy vấn nâng cao trên mộtphẳngkiểu dữ liệu**

NHẬN tư vấn/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [

{"match": {"doctor\_notes": "đau đầu"}},

{"match": {"doctor\_notes": "Thuốc kháng sinh"}}

],

"must\_not": [

{"term": {"doctor\_notes": {"value": "bệnh tiểu đường"}}}

]

}

}

}

**142 CPHẦN4*Bản đồ***

Trong truy vấn, chúng tôi kiểm tra xem bệnh nhân có bị đau đầu và dùng thuốc kháng sinh không, nhưng bệnh nhân không bị tiểu đường—truy vấn trả về John Doe vì anh ta không bị tiểu đường nhưng bị đau đầu và đang dùng thuốc kháng sinh.

Các kiểu dữ liệu phẳng có ích, đặc biệt là khi chúng ta mong đợi nhiều trường trên cơ sở ad hoc và việc xác định định nghĩa ánh xạ cho tất cả các trường trước là không khả thi. Hãy nhớ rằng các tệp con của một trường phẳng luôn là các kiểu từ khóa.

* + 1. Kiểu dữ liệu tham gia

Nếu bạn đến từ thế giới cơ sở dữ liệu quan hệ, bạn biết mối quan hệ giữa dữ liệu—các phép nối cho phép mối quan hệ cha-con. Mọi tài liệu được Elasticsearch lập chỉ mục đều độc lập và không duy trì mối quan hệ với bất kỳ tài liệu nào khác trong chỉ mục đó. Elasticsearch phi chuẩn hóa dữ liệu để đạt được tốc độ và tăng hiệu suất trong quá trình lập chỉ mục và tìm kiếm. Mặc dù chúng ta được khuyên nên thận trọng khi duy trì và quản lý các mối quan hệ trong Elasticsearch, nhưng kiểu dữ liệu nối có sẵn để tạo mối quan hệ cha-con nếu chúng ta cần.

Chúng ta hãy tìm hiểu về dữ liệu liên kết bằng cách xem xét một ví dụ về mối quan hệ bác sĩ-bệnh nhân (một-nhiều): một bác sĩ có thể có nhiều bệnh nhân và mỗi bệnh nhân được chỉ định cho một bác sĩ. Để làm việc với các mối quan hệ cha-con bằng cách sử dụng kiểu dữ liệu liên kết, chúng ta cần tạo một trường là kiểu liên kết và thêm thông tin thông quađối tượng quan hệ đề cập đến mối quan hệ (trong trường hợp này là mối quan hệ bác sĩ-bệnh nhân). Danh sách sau đây chuẩn bị chỉ mục bác sĩ với định nghĩa lược đồ.

**Liệt kê 4.36 Bản đồ củabác sĩđịnh nghĩa lược đồ**

bác sĩ PUT

{

"ánh xạ": { "thuộc tính": {

"mối quan hệ":{ "kiểu": "tham gia", "mối quan hệ":{

"bác sĩ":"bệnh nhân"

}

**Khai báo một thuộc tính là kiểu liên kết**

**Tên của các mối quan hệ**

}

}

}

}

Truy vấn này có hai điểm quan trọng cần lưu ý:

* Chúng tôi tuyên bố mộtmối quan hệthuộc tính của loạitham gia.
* Chúng tôi tuyên bố mộtquan hệthuộc tính và đưa ra tên của các mối quan hệ (trong trường hợp này, chỉ có mộtbác sĩ:bệnh nhânmối quan hệ).

Sau khi sơ đồ đã sẵn sàng và được lập chỉ mục, hãy lập chỉ mục cho hai loại tài liệu: một loại đại diện cho bác sĩ (cha mẹ) và loại còn lại đại diện cho hai bệnh nhân (con cái). Đây là tài liệu của bác sĩ, với mối quan hệ có tên là bác sĩ.

***4.6 Dữ liệu nâng caocác loại* 143**

**Liệt kê 4.37 Lập chỉ mục tài liệu bác sĩ**

PUT bác sĩ/\_doc/1

{

"tên":"Tiến sĩ Mary Montgomery", "mối quan hệ":{

"tên":"bác sĩ"

}

}

**Thuộc tính mối quan hệ phải là một trong các mối quan hệ.**

Đối tượng mối quan hệ tuyên bố rằng loại tài liệu là doctor. Thuộc tính name phải là giá trị cha (doctor) như đã khai báo trong lược đồ ánh xạ. Bây giờ, khi Dr. Mary Montgomery đã sẵn sàng, bước tiếp theo là liên kết hai bệnh nhân với cô ấy.

 **Liệt kê 4.38 Tạo hai bệnh nhân cho bác sĩ của chúng tôi**

PUT bác sĩ/\_doc/2?routing=mary

{

"tên":"John Doe",

**Tài liệu phải được thiết lập cờ định tuyến.**

**Tài liệu này là của một bệnh nhân.**

"mối quan hệ":{ "tên":"bệnh nhân", "cha mẹ":1

}

}

**Chúng ta định nghĩa loại mối quan hệ trong đối tượng này.**

**Cha mẹ bệnh nhân (bác sĩ) là tài liệu có ID 1.**

PUT bác sĩ/\_doc/3?routing=mary

{

"tên":"Bà Doe", "mối quan hệ":{

"tên":"bệnh nhân", "cha mẹ":1

}

}

Đối tượng quan hệ phải có giá trị là bệnh nhân (bạn còn nhớ phần cha-con của thuộc tính quan hệ trong lược đồ không?) và cha phải được chỉ định một mã định danh tài liệu của bác sĩ liên quan (ID 1 trong ví dụ của chúng ta).

Chúng ta cần hiểu thêm một điều nữa khi làm việc với các mối quan hệ cha-con. Cha mẹ và con cái liên quan được lập chỉ mục vào cùng một phân đoạn để tránh chi phí tìm kiếm nhiều phân đoạn. Và vì các tài liệu cùng tồn tại, chúng ta cần sử dụng tham số định tuyến bắt buộc trong URL. (Định tuyến là một hàm xác định phân đoạn nơi tài liệu nằm; chúng ta sẽ xem xét thuật toán định tuyến trong chương 5.)

Cuối cùng, đã đến lúc tìm kiếm bệnh nhân thuộc về bác sĩ có ID 1. Truy vấn sau đây sẽ tìm kiếm tất cả bệnh nhân có liên quan đến Tiến sĩ Montgomery.

**Liệt kê 4.39 Lấy bệnh nhân của Bác sĩ.Montgomery**

GET bác sĩ/\_tìm kiếm

{

"truy vấn": {

"parent\_id":{

**144 CPHẦN4*Bản đồ***

"loại":"bệnh nhân", "id":1

}

}

}

Để tìm kiếm bệnh nhân thuộc về một bác sĩ, chúng tôi sử dụng truy vấn tìm kiếm có tên parent\_id, truy vấn này yêu cầu loại con (bệnh nhân) và ID của cha mẹ (ID tài liệu của Bác sĩ Montgomery là 1). Truy vấn này trả về bệnh nhân của Bác sĩ Montgomery: Ông và Bà Doe.

**GHI CHÚ**Việc triển khai mối quan hệ cha-con trong Elasticsearch có tác động đến hiệu suất. Như chúng ta đã thảo luận trong chương 1, Elasticsearch có thể không phải là công cụ phù hợp nếu bạn đang cân nhắc mối quan hệ tài liệu, vì vậy hãy sử dụng tính năng này một cách thận trọng.

* + 1. Kiểu dữ liệu search\_as\_you\_type

Hầu hết các công cụ tìm kiếm đều gợi ý các từ và cụm từ khi chúng ta nhập vào thanh tìm kiếm. Tính năng này có nhiều tên gọi khác nhau: tìm kiếm khi bạn nhập, gõ trước, tự động hoàn thành hoặc gợi ý. Elasticsearch cung cấp một kiểu dữ liệu tiện lợi có tên là search\_as\_you\_type để hỗ trợ tính năng này. Đằng sau hậu trường, Elasticsearch làm việc rất chăm chỉ để đảm bảo rằng các trường được gắn thẻ là search\_as\_you\_type được lập chỉ mục để tạo ra n-gram, mà chúng ta thấy trong hành động trong phần này.

Giả sử chúng ta được yêu cầu hỗ trợ các truy vấn gõ trước trên một chỉ mục sách: khi người dùng bắt đầu nhập từng chữ cái tiêu đề của một cuốn sách vào thanh tìm kiếm, chúng ta có thể gợi ý sách dựa trên các chữ cái họ nhập. Trước tiên, chúng ta cần tạo một lược đồ với trường có kiểu search\_as\_you\_type.

**Liệt kê 4.40 Sơ đồ ánh xạ vớitìm kiếm theo kiểu bạnkiểu**

ĐẶT tech\_books4

{

"ánh xạ": { "thuộc tính": {

"tiêu đề": {

"loại": "tìm kiếm theo loại bạn"

}

**Tiêu đề hỗ trợ tính năng gõ trước.**

}

}

}

Điểm đáng chú ý là tiêu đề trong định nghĩa lược đồ được khai báo làKiểu dữ liệu search\_as\_you\_type. Hãy lập chỉ mục một số tài liệu có nhiều tiêu đề khác nhau (cuốn sách này cùng với các tiêu đề tương lai của tôi).

**Liệt kê 4.41 Lập chỉ mục một sốsách**

ĐẶT tech\_books4/\_doc/1

{

"title":"Elasticsearch trong hành động"

}

* 1. ***Dữ liệu nâng caocác loại* 145**

ĐẶT tech\_books4/\_doc/2

{

"title":"Elasticsearch dành cho nhà phát triển Java"

}

ĐẶT tech\_books4/\_doc/3

{

"title":"Elastic Stack đang hoạt động"

}

Bởi vìtiêu đềKiểu dữ liệu của trường làtìm kiếm theo kiểu bạn, Elasticsearch tạo ra một tập hợp các trường con được gọi là n-gram, một phần khớp với tìm kiếm của người dùng.\_tiền tố\_chỉ\_sốtạo ra các gram cạnh-n như[Và,Anh ta,cô ấy,họ,đàn hồi,đàn hồi,đàn hồi]cho từ Elastic. Một token 2 gram là một bộ lọc token dạng tấm lợp tạo ra hai token, ["elastic- ticsearch","TRONG"]Và["TRONG","hoạt động"]", cho tiêu đề Elasticsearch in Action. Tương tự như vậy, 3-gram là một bộ lọc mã thông báo được sắp xếp tạo ra các mã thông báo như ["elasticsearch cho java","dành cho các nhà phát triển java"]cho tiêu đề Elasticsearch dành cho nhà phát triển Java. (Mã nguồn có sẵn trong các tệp của cuốn sách, nếu bạn muốn thử các ví dụ này.)

Ngoài các n-gram này, trường gốc (tiêu đề) được lập chỉ mục như hiện tại với trình phân tích đã cho hoặc mặc định. Tất cả các n-gram khác được tạo ra bằng cách sử dụng nhiều bộ lọc shingle-token khác nhau, như thể hiện trong bảng 4.5.

**Bảng 4.5 Các trường con được tạo tự động bởi công cụ**

|  |  |  |
| --- | --- | --- |
| **Các cánh đồng** | **Giải thích** | **Ví dụ** |
| tiêu đề | Trường tiêu đề được lập chỉ mục vớihoặc là trình phân tích được chọn hoặc là trình phân tích mặc định nếu không chọn trình phân tích nào. | Nếu sử dụng trình phân tích chuẩn, tiêu đề sẽ được mã hóa và chuẩn hóa dựa trên các quy tắc của trình phân tích chuẩn đó. |
| tiêu đề.\_2gram | Bộ phân tích trường tiêu đề được tùy chỉnh-được xử lý bằng bộ lọc shingle-token. Kích thước shingle được đặt thành 2 trên bộ lọc này. | Tạo ra hai mã thông báo chovăn bản. Ví dụ, 2-gram cho tiêu đề"*Elastic-tìm kiếm trong hành động*"là["elasticsearch","trong"],["trong","hành động"]. |
| tiêu đề.\_3gram | Bộ phân tích trường tiêu đề được tùy chỉnh-được xử lý bằng bộ lọc shingle-token. Kích thước shingle được đặt thành 3 trên bộ lọc này. | Tạo ra ba mã thông báo cho văn bản đã cho. Ví dụ, 3-gram cho tiêu đề"*Elasticsearch cho Javacác nhà phát triển*"là["elasticsearch","cho","java"],  ["cho","java","nhà phát triển"] |
| Tiêu đề.\_index tiền tố | Bộ phân tích của title.\_3gram được áp dụng cùng với một cạnh n-grambộ lọc mã thông báo | Tạo n-gram cạnh cho trường  title.\_3grams. Ví dụ,  \_index\_prefix tạo ra nội dung saucạnh n-gram chothế giới"Đàn hồi": [và, el, cô ấy, đàn hồi, đàn hồi, đàn hồi] |

Vì các trường này được tạo cho chúng ta, nên việc tìm kiếm trên trường sẽ trả về các gợi ý nhập trước, vì n-gram giúp tạo ra chúng một cách hiệu quả. Hãy tạo truy vấn tìm kiếm như được hiển thị trong danh sách tiếp theo.

**146 CPHẦN4*Bản đồ***

**Liệt kê 4.42 Tìm kiếm trongtìm kiếm theo kiểu bạnvà nócác lĩnh vực phụ**

NHẬN tech\_books4/\_search

{

"truy vấn": { "multi\_match": {

"truy vấn": "trong",

"kiểu": "bool\_prefix",

"các trường": ["tiêu đề","title.\_2gram","title.\_3gram"]

}

}

}

Như truy vấn cho thấy, chúng tôi đang tạo ra mộtnhiều\_trận\_hợptruy vấn (với mộtbool\_tiền tốnhư kiểu) vì tìm kiếm được thực hiện trên toàn bộ trường chính và các trường con của nó (tiêu đềvà các lĩnh vực phụ củatiêu đề:\_2 gam,\_3 gam, v.v.). Truy vấn này sẽ trả về các cuốn sách*Elasticsearch trong hành động*Và*Elastic Stack trong hành động*. Chúng tôi sử dụng mộtnhiều trận đấutruy vấn vì chúng ta đang tìm kiếm một giá trị trên nhiều trường:tiêu đề,tiêu đề.\_2gram, Vàtiêu đề.\_3gram.

**N-gram, n-gram cạnh và bệnh zona**

Nếu bạn lần đầu tiên nghe về n-gram, n-gram cạnh và bệnh zona, thì đây làcác khái niệm có thể làm bạn bối rối. Tôi giải thích ngắn gọn ở đây và chúng tôi sẽ trình bày chi tiết trong chương 7.

MỘT*n-gram*là một chuỗi các từ có kích thước nhất định. Chúng ta có thể có 2-gram, 3-gram, v.v. Ví dụ, nếu từ là*hoạt động*, 3-gram (n-gram cho kích thước 3) là["hành động", "cti", "tio", "ion"], các bi-gram (kích thước 2) là["ac", "ct","ti","io","TRÊN"], và vân vân.

*n-gram cạnh*là n-gram của mọi token, trong đó điểm bắt đầu của n-gram được neođến đầu của từ. Đối với*hoạt động*,các n-gram cạnh là["a", "ac", "act", "acti", "actio", "hành động"].

*Bệnh zona*là các từ n-gram. Ví dụ, câu“Elasticsearch in Action” vượt trội hơn hẳn["Tìm kiếm đàn hồi","Elasticsearch trong", "Elasticsearch trong hành động", "trong", "trong hành động", "Hành động"].

Đôi khi chúng ta có thể muốn một trường mà chúng ta có thể khai báo là nhiều hơn một kiểu dữ liệu duy nhất.Ví dụ, tiêu đề phim có thể là cả kiểu dữ liệu văn bản và hoàn thành. May mắn thay, Elasticsearch cho phép chúng ta khai báo một trường duy nhất với nhiều kiểu dữ liệu. Hãy cùng tìm hiểu cách chúng ta có thể tạo các trường có nhiều kiểu.

#### Các trường có nhiều kiểu dữ liệu

Chúng ta đã thấy rằng mỗi trường trong một tài liệu được liên kết với một kiểu dữ liệu. Tuy nhiên, Elasticsearch rất linh hoạt và cũng cho phép chúng ta định nghĩa các trường có nhiều kiểu dữ liệu. Ví dụ:trường chủ đề trong dữ liệu email của chúng tôi có thể là văn bản, từ khóa hoặc loại hoàn thành,

***Bản tóm tắt* 147**

tùy thuộc vào yêu cầu của chúng tôi. Chúng tôi có thể tạo nhiều loại trong định nghĩa lược đồ bằng cách sử dụng đối tượng fields bên trong định nghĩa trường chính. Nó có cú pháp sau:

"my\_field1":{ "type": "văn bản", "các trường": {

**Khai báo kiểu**

**củatrường\_của\_tôi1 Xác định một đối tượng trường để bao gồm nhiều loại hơn**

"kw":{ "type":"từ khóa" }

}

}

**Khai báo một trường bổ sung với nhãn kw**

Về cơ bản,trường\_của\_tôi1được lập chỉ mục như mộtchữloại cũng như mộttừ khóaloại. Khi chúng ta mong đợi sử dụng nó nhưchữ, chúng tôi có thể cung cấp lĩnh vực này nhưtrường\_của\_tôi1trong các truy vấn. Chúng tôi sử dụng nhãnmy\_fields1.kw(lưu ý ký hiệu dấu chấm) làm tên trường khi tìm kiếm nó như mộttừ khóa.

Định nghĩa lược đồ ví dụ sau đây tạo ra trường đơn của chúng tôichủ thểvới nhiều kiểu dữ liệu (chữ,từ khóa, Vàhoàn thành).

**Liệt kê 4.43 Định nghĩa lược đồ với trường đa kiểu**

PUT email\_multi\_type

{

"ánh xạ": { "thuộc tính": {

"chủ đề":{ "kiểu": "văn bản", "trường": {

**Văn bảnkiểu**

**Chủ đề là**

"kw":{ "type":"từ khóa" },

"comp":{ "type":"hoàn thành" }

}

}

}

}

}

**cũng là một từ khóa.**

**Chủ ngữ cũng là một loại hoàn thành.**

Cácchủ thểtrường có ba loại liên quan đến nó:chữ,từ khóa, Vàhoàn thành. Để truy cập những thứ này, chúng ta phải sử dụng định dạngchủ đề.kwchotừ khóa-loại trường hoặcchủ đề.compcho loại hoàn thành.

Chúng ta đã học được rất nhiều về khái niệm lập bản đồ trong chương này. Hãy kết thúc và hướng tới chương tiếp theo, thảo luận về cách làm việc với tài liệu.

#### Bản tóm tắt

* Mỗi tài liệu bao gồm các trường có giá trị và mỗi trường có một kiểu dữ liệu. Elasticsearch cung cấp một tập hợp phong phú các kiểu dữ liệu để biểu diễn các giá trị này.
* Elasticsearch tham khảo một bộ quy tắc khi lập chỉ mục và tìm kiếm dữ liệu. Các quy tắc này, được gọi là quy tắc ánh xạ, cho Elasticsearch biết cách xử lý các hình dạng dữ liệu khác nhau.
* Các quy tắc ánh xạ được hình thành thông qua các quá trình ánh xạ động hoặc rõ ràng.

**148 CPHẦN4*Bản đồ***

* Mapping là cơ chế tạo định nghĩa lược đồ trường trước. Elastic-search tham khảo định nghĩa lược đồ trong khi lập chỉ mục tài liệu để dữ liệu được phân tích và lưu trữ để truy xuất nhanh hơn.
* Elasticsearch cũng có một tính năng ánh xạ mặc định: chúng ta có thể để Elasticsearch suy ra ánh xạ thay vì tự cung cấp rõ ràng. Elasticsearch xác định lược đồ dựa trên lần đầu tiên nhìn thấy một trường.
* Mặc dù ánh xạ động rất tiện lợi, đặc biệt là trong quá trình phát triển, nhưng nếu chúng ta biết nhiều hơn về mô hình dữ liệu, tốt nhất là nên tạo ánh xạ trước.
* Elasticsearch cung cấp nhiều kiểu dữ liệu cho văn bản, Boolean, giá trị số, ngày tháng, v.v., bao gồm cả các trường phức tạp như nối, hoàn thành, điểm địa lý, lồng nhau và các trường khác.

*Làm việc với tài liệu*

***Chương này bao gồm***

* Lập chỉ mục, truy xuất và lập chỉ mục lại tài liệu
* Thao tác và điều chỉnh phản hồi
* Cập nhật và xóa tài liệu bằng API vàphương pháp truy vấn
* Làm việc với các bản cập nhật theo kịch bản
* Lập chỉ mục tài liệu hàng loạt

Đã đến lúc làm việc với và hiểu các thao tác trên tài liệu trong Elasticsearch. Tài liệu được lập chỉ mục, tìm nạp, cập nhật hoặc xóa dựa trên yêu cầu của chúng tôi. Chúng tôi có thể tải dữ liệu vào Elasticsearch từ các kho lưu trữ như cơ sở dữ liệu và tệp hoặc từ các luồng thời gian thực. Tương tự như vậy, chúng tôi có thể cập nhật hoặc sửa đổi dữ liệu hiện có trong Elasticsearch. Nếu cần, chúng tôi thậm chí có thể xóa và xóa tài liệu. Ví dụ, chúng tôi có thể có cơ sở dữ liệu danh mục sản phẩm cần được nhập vào Elasticsearch để kích hoạt khả năng tìm kiếm trên các sản phẩm.

Elasticsearch cung cấp một bộ API để làm việc với các tài liệu của chúng tôi:

* + *API lập chỉ mục tài liệu*—Để lập chỉ mục tài liệu vào Elasticsearch
  + *Đọc và tìm kiếm API*—Cho phép khách hàng tìm kiếm/lấy tài liệu

**149**

**150 CPHẦN5*Làm việc với tài liệu***

* *Cập nhật API*—Để chỉnh sửa và sửa đổi các trường của một tài liệu
* *Xóa API*—Để xóa tài liệu khỏi kho lưu trữ

Elasticsearch phân loại các API này thành hai loại: API một tài liệu và API nhiều tài liệu. Như tên gọi của nó, API một tài liệu thực hiện các hoạt động như lập chỉ mục, tìm nạp, sửa đổi và xóa từng tài liệu một, bằng cách sử dụng các điểm cuối thích hợp. Các API này hữu ích khi làm việc với các sự kiện như đơn hàng do ứng dụng thương mại điện tử tạo ra hoặc tweet từ một tập hợp các tài khoản Twitter. Chúng tôi sử dụng API một tài liệu để vận hành trên các tài liệu này riêng lẻ.

*API đa tài liệu*Mặt khác, hướng đến việc làm việc với các tài liệu theo lô. Ví dụ, chúng ta có thể có yêu cầu nhập danh mục sản phẩm vào Elasticsearch từ cơ sở dữ liệu gồm hàng triệu bản ghi. Elasticsearch đưa ra API điểm cuối hàng loạt (\_bulk) cho mục đích này để giúp nhập dữ liệu theo lô.

Elasticsearch cũng cung cấp các API dựa trên truy vấn nâng cao để làm việc với các tài liệu của chúng tôi. Chúng tôi có thể xóa và cập nhật một số tài liệu phù hợp với các tiêu chí nhất định bằng cách phát triển một truy vấn phức tạp nếu cần. Chúng tôi cũng có thể sử dụng các truy vấn tìm kiếm phức tạp để tìm và cập nhật hoặc xóa các tài liệu. Cuối cùng, chúng tôi có thể di chuyển dữ liệu từ chỉ mục này sang chỉ mục khác bằng cách sử dụng API lập chỉ mục lại. Việc lập chỉ mục lại giúp chúng tôi di chuyển dữ liệu mà không có bất kỳ thời gian chết nào trong quá trình sản xuất, mặc dù chúng tôi phải cân nhắc đến các tác động về hiệu suất.

Trong chương này, chúng ta sẽ thảo luận về API đơn và đa tài liệu và các hoạt động khác nhau trên tài liệu. Bước đầu tiên trong quá trình làm việc với Elasticsearch là đưa một số dữ liệu vào công cụ. Chúng ta hãy bắt đầu bằng cách xem xét các API lập chỉ mục tài liệu và cơ chế lập chỉ mục tài liệu.

**GHI CHÚ**Mã cho chương này có sẵn trên GitHub ([http://mng.bz/](http://mng.bz/MBr8) [MBr8](http://mng.bz/MBr8)) và trên trang web của cuốn sách ([https://www.manning.com/books/elastic](https://www.manning.com/books/elasticsearch-in-action-second-edition) [tìm kiếm-trong-hành-động-phiên-bản-thứ-hai](https://www.manning.com/books/elasticsearch-in-action-second-edition)). Bạn có thể tìm thấy bộ dữ liệu phim được sử dụng trong các ví dụ tại<http://mng.bz/a1pX>.

#### Lập chỉ mục tài liệu

Cũng giống như chúng ta chèn bản ghi vào cơ sở dữ liệu quan hệ, chúng ta thêm dữ liệu (dưới dạng tài liệu) vào Elasticsearch. Các tài liệu này nằm trong một thùng logic gọi là chỉ mục. Hành động lưu trữ tài liệu vào các chỉ mục này được gọi là lập chỉ mục. Vì vậy, khi chúng ta nghe thuật ngữ lập chỉ mục, nó có nghĩa là lưu trữ hoặc lưu trữ tài liệu vào Elasticsearch.

**Phân tích văn bản**

Các tài liệu trải qua một quá trình gọi là*chữPhân tích*trong Elasticsearch trước khi được lưu trữ. Quá trình phân tích chuẩn bị dữ liệu để phù hợp với nhiều tính năng tìm kiếm và phân tích khác nhau. Phân tích này cung cấp cho công cụ tìm kiếm khả năng cung cấp tính liên quan và khả năng tìm kiếm toàn văn. Phân tích văn bản được thảo luận sâu hơn trong chương 7.

***5.1 Lập chỉ mụctài liệu* 151**

Như chúng tôi đã đề cập trong các đoạn mở đầu của chương này, Elasticsearch cung cấp API để lập chỉ mục cả tài liệu đơn lẻ và nhiều tài liệu vào Elasticsearch. Chúng tôi sẽ xem xét chi tiết các API tài liệu này trong phần tiếp theo.

* + 1. API tài liệu

Chúng ta giao tiếp với Elasticsearch bằng API RESTful qua HTTP. Chúng ta có thể thực hiện các thao tác CRUD (tạo, đọc, cập nhật và xóa) cơ bản bằng API tài liệu đơn. Ngoài ra còn có các API được thiết kế để làm việc với nhiều tài liệu thay vì nhắm mục tiêu vào một tài liệu duy nhất. Chúng ta thảo luận về cả hai loại trong chương này, nhưng bây giờ, hãy tập trung vào cách lập chỉ mục tài liệu bằng API tài liệu đơn. Để bắt đầu, có một khái niệm quan trọng cần hiểu: định danh tài liệu.

**DNHẬN DẠNG TÀI SẢN**

Mỗi tài liệu chúng tôi lập chỉ mục có thể có một mã định danh (ID), thường do người dùng chỉ định. Đối vớiVí dụ, tài liệu cho bộ phim The Godfather có thể được cấp một ID (ví dụ, id = 1) và bộ phim The Shawshank Redemption có thể được cấp một ID khác (id = 2). Tương tự như khóa chính trong cơ sở dữ liệu quan hệ, ID được liên kết với tài liệu đó trong suốt thời gian tồn tại của nó (trừ khi nó được cố ý thay đổi).

Đôi khi, khách hàng (người dùng) không cần cung cấp ID cho tài liệu. Hãy tưởng tượng một chiếc xe tự động gửi hàng nghìn cảnh báo và nhịp tim đến máy chủ. Không phải mọi thông báo đều cần một chuỗi ID; nó có thể có một ID ngẫu nhiên, mặc dù là ID duy nhất. Trong trường hợp này, hệ thống tạo ra một mã định danh duy nhất phổ biến ngẫu nhiên (UUID) cho tài liệu đang được lập chỉ mục.

API tài liệu cho phép chúng ta lập chỉ mục tài liệu có hoặc không có ID. Nhưng có một sự khác biệt tinh tế khi sử dụng các động từ HTTP như POST và PUT:

* + - * Nếu một tài liệu có ID do khách hàng cung cấp, chúng tôi sử dụng HTTP ĐẶTphương pháp gọi API tài liệu để lập chỉ mục tài liệu.
      * Nếu tài liệu không có ID do khách hàng cung cấp, chúng tôi sử dụng HTTPBƯU KIỆNphương pháp khi lập chỉ mục. Trong trường hợp này, sau khi tài liệu được lập chỉ mục, nó sẽ kế thừa ID do hệ thống tạo ra.

Hãy cùng xem xét quá trình lập chỉ mục tài liệu bằng cả hai phương pháp.

**TÔIXÁC ĐỊNH MỘT TÀI LIỆU CÓ MỘT MÃ SỐ NHẬN DẠNG****(ĐẶT)**

Khi một tài liệu có ID, chúng ta có thể sử dụng API chỉ mục tài liệu đơn (\_doc) với hành động PUT của HTTP để lập chỉ mục tài liệu. Cú pháp cho phương pháp này như sau:

PUT <tên\_chỉ\_mục>/\_doc/<mã\_danh>

Tại đây, <index\_name> là tên của chỉ mục nơi tài liệu sẽ được lưu trữ và \_doc là điểm cuối phải có khi lập chỉ mục tài liệu.

<identifier> là định danh của tài liệu (giống như khóa chính trong cơ sở dữ liệu), là tham số đường dẫn bắt buộc khi sử dụng phương thức HTTP PUT.

Hãy lập chỉ mục một tài liệu phim bằng API. Vào Kibana, viết và thực thi truy vấn sau.

**152 CPHẦN5*Làm việc với tài liệu***

**Liệt kê 5.1 Lập chỉ mục một tài liệu mới vàophimchỉ số**

PUT phim/\_doc/1

{ 

"title":"Bố già",

**Nội dung yêu cầu**

**URL lập chỉ mục tài liệu**

"tóm tắt": "Người đứng đầu già nua của một triều đại tội phạm có tổ chức chuyển giao

➥quyền kiểm soát đế chế bí mật của mình cho đứa con trai miễn cưỡng của mình"

}

URL PUT movies/\_doc/1 là phương thức RESTful gọi API chỉ mục tài liệu. Yêu cầu này có phần thân được biểu diễn bằng tài liệu JSON kèm theo (có dữ liệu phim). Kết quả được hiển thị trong hình 5.1. Chúng ta hãy nhanh chóng xem qua các phần của URL, như được liệt kê trong bảng 5.1.

PUT phim/\_doc/1

{

"title":"Bố già", "synopsis":"Người cha già

của một tội phạm có tổ chức .."

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "1",

"\_phiên bản" : 1,

**Phản hồi từ máy chủ cho biết rằngbộ phimtài liệu đã được**

} "kết quả" : "đã tạo",**được lập chỉ mục**

**Một yêu cầu gửi đến máy chủ để lập chỉ mụcbộ phimtài liệu có ID của1. Nội dung của yêu cầu là một tài liệu JSON.**

"\_mảnh vỡ" : {

"tổng cộng" : 4,

"thành công" : 1,

"thất bại" : 0

},

"\_seq\_no" : 0,

"\_primary\_term" : 1

}

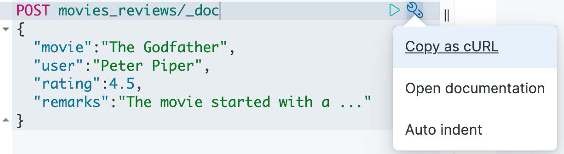
**thành công**

**Hình 5.1 Lập chỉ mục tài liệu bằng ID**

**Bàn5.1 ĐẶTThành phần URL**

|  |  |
| --- | --- |
| ĐẶT | Động từ HTTP chỉ ra rằng chúng ta đang yêu cầu máy chủ tạo một tài nguyên mới. Quy ước thông thường cho mộtĐẶThành động này là gửi một số dữ liệu đến URL tài nguyên để máy chủ tạo tài nguyên mới trong kho lưu trữ của nó. |
| phim | Tên của chỉ mục mà chúng tôi muốnbộ phimtài liệu để tồn tại. |
| \_tài liệu | Điểm cuối của lệnh gọi dịch vụ. Trong các phiên bản Elasticsearch trước đó (trước 5.x), URL có một loại được liên kết với nó (giống như movies/movie/1). Loại tài liệu đã lỗi thời và điểm cuối chung \_doc đã được sửa đổi thành URL (làm cho nó thành movies/\_doc/1). |
| 1 | Tham số đường dẫn để chỉ ra ID của tài nguyên (bộ phimID của tài liệu). |

***5.1 Lập chỉ mụctài liệu* 153**



**Yêu cầu theo định dạng cURL**

Kibana rút ngắn URL để trông đẹp. Đằng sau hậu trường, nó mở rộng URL bằng cách thêm thông tin chi tiết về máy chủ vào yêu cầu. Điều này có thể thực hiện được vì mọi phiên bản Kibana đều được kết nối ngầm với máy chủ Elasticsearch (tệp cấu hình kibana.yml xác định thông tin chi tiết của máy chủ). URL ở định dạng cURL là

curl -XPUT "http://localhost:9200/movies/\_doc/1"

-H 'Loại nội dung: ứng dụng/json'

-d'{ "title":"Bố già", "synopsis":"Người cha già già nua .."}')

Bạn có thể lấy lệnh cURL từ Dev Tools của Kibana—chỉ cần nhấp vào biểu tượng cờ lê và sao chép yêu cầu của bạn dưới dạng URL bằng cách chọn Sao chép dưới dạng cURL như trong hình.

**Xuất truy vấn dưới dạng cURL**

**MẸO**Khi bạn ở đây, hãy nhấp vào liên kết Tự động thụt lề để thụt lề mã của bạn theo cách JSON.

Sau khi thực thi tập lệnh trước, chúng tôi mong đợi phản hồi sẽ xuất hiện trong ngăn bên phải của Kibana (xem hình 5.2). Chúng ta hãy phân tích nó một cách ngắn gọn. Phản hồi là một tài liệu JSON có một vài thuộc tính. Thuộc tính result chỉ ra hoạt động

**Tên chỉ mục tài liệu (phim)**

**Loại tài liệu: mặc định là\_tài liệu(kiểu tài liệu đã lỗi thời, vì vậy giá trị được đặt thành\_tài liệu)**

**Mã định danh tài liệu**

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "1",

"\_version" : 1, "result" : "đã tạo", "\_shards" : {

"tổng cộng" : 4,

"thành công" : 1,

"thất bại" : 0

},

"\_seq\_no" : 0,

"\_primary\_term" : 1

}

**Phiên bản tài liệu:1cho biết đây là phiên bản đầu tiên.**

**Tài nguyên đã được tạo thành công, vì vậy**

**kết quả = đã tạo.**

**Hình 5.2 Phản hồi từ máy chủ khi tài liệu được lập chỉ mục thành công**

**154 CPHẦN5*Làm việc với tài liệu***

hành động: created cho biết chúng tôi đã lập chỉ mục thành công một tài liệu mới trong Elastic-store của search. Các thuộc tính \_index, \_type và \_id được lấy từ yêu cầu đến của chúng tôi và được gán cho tài liệu. \_version chỉ ra phiên bản hiện tại của tài liệu này; giá trị 1 có nghĩa là phiên bản đầu tiên của tài liệu. Số này tăng lên nếu chúng tôi sửa đổi tài liệu và lập chỉ mục lại. Chúng tôi truy cập thông tin \_shards theo dòng, nhưng tất cả những gì nó nói ở đây là hành trình của tài liệu đến cửa hàng đã hoàn tất.

Nếu bạn tò mò, hãy thực hiện lại lệnh tương tự và kiểm tra phản hồi. Bạn sẽlưu ý rằng thuộc tính kết quả thay đổi thành cập nhật và số phiên bản tăng dần. Mỗi lần bạn thực hiện truy vấn, phiên bản sẽ được cập nhật.

Khi chúng tôi lần đầu tiên thử chèn tài liệu, không có chỉ mục phim nào được tạo trước cho chúng tôi (nếu chỉ mục đã tồn tại, hãy đảm bảo bạn xóa nó bằng cách chạy lệnh DELETE movies). Khi máy chủ lần đầu tiên nhìn thấy tài liệu, máy chủ nhận ra rằng chỉ mục vẫn chưa tồn tại và tạo chỉ mục phim cho chúng tôi để quá trình lập chỉ mục tài liệu diễn ra suôn sẻ. Sơ đồ của chỉ mục này cũng không tồn tại, nhưng máy chủ đã lấy sơ đồ bằng cách phân tích tài liệu đầu tiên đến. Chúng tôi đã tìm hiểu về định nghĩa ánh xạ trong chương 4; tham khảo phần 4.1.1 nếu cần.

Chúng tôi vừa lập chỉ mục một tài liệu có ID. Không phải mọi tài liệu đều có thể hoặc sẽ có danh tính với nhu cầu kinh doanh. Ví dụ, trong khi một tài liệu phim có ID là một ý tưởng hay, thì cảnh báo giao thông từ đèn giao thông do hệ thống quản lý giao thông nhận được không cần phải liên kết với ID. Tương tự như vậy, luồng tweet, tin nhắn bình luận trận đấu cricket, v.v. không cần danh tính—một ID ngẫu nhiên là đủ.

Việc lập chỉ mục một tài liệu không có ID tuân theo cùng một quy trình như lập chỉ mục một tài liệu có ID, ngoại trừ động từ HTTP thay đổi từ PUT thành POST. Hãy cùng xem điều này hoạt động như thế nào.

**TÔI****XÁC ĐỊNH MỘT TÀI LIỆU KHÔNG CÓ MÃ SỐ NHẬN DẠNG(BƯU KIỆN)**

Trong phần trước, chúng ta đã lập chỉ mục một tài liệu có ID được liên kết với nó. Chúng ta đã sử dụng phương thức HTTP PUT để tạo tài nguyên trong Elasticsearch. Nhưng không phải mọi mô hình kinh doanh đều quảng bá tài liệu dữ liệu có ID (ví dụ: giá cổ phiếu hoặc tweet). Trong những trường hợp như vậy, chúng ta có thể để Elasticsearch tạo ID ngẫu nhiên. Để lập chỉ mục nhữngtài liệu, thay vì sử dụng phương thức PUT như chúng ta đã làm trước đó, chúng ta sử dụng hành động POST.

Phương thức HTTP POST có định dạng tương tự như PUT, nhưng ID không được cung cấp như một phần của tham số URL.

**Liệt kê 5.2 Lập chỉ mục một tài liệu không có ID bằng cách sử dụngBƯU KIỆN**

ĐĂNG myindex/\_doc

{

"title":"Elasticsearch trong hành động"

}

**URLkhông có ID đính kèm.**

**Nội dung yêu cầu là một tài liệu JSON.**

Chúng tôi gọi điểm cuối \_doc trên chỉ mục không có ID nhưng có phần thân. Yêu cầu POST này cho Elasticsearch biết rằng nó cần gán các ID ngẫu nhiên mới được tạo cho các tài liệu trong quá trình lập chỉ mục.

***5.1 Lập chỉ mục tài liệu***

**GHI CHÚ**Phương pháp POST không mong đợi người dùng cung cấp ID tài liệu. Thay vào đó, nó tự động tạo một ID được tạo ngẫu nhiên cho tài liệu khi nó được lưu trữ. Chúng ta cũng có thể sử dụng POST với một ID. Ví dụ: POSTmyindex/\_doc/1 là một lệnh gọi hợp lệ. Tuy nhiên, PUT không có ID là không hợp lệ; nghĩa là, lệnh gọi PUT myindex/\_doc sẽ gây ra lỗi.

**155**

Ví dụ, hãy xem xét người dùng đăng bài đánh giá phim. Mỗi bài đánh giá phim được ghi lại dưới dạng tài liệu JSON, được gửi đến Elasticsearch từ Kibana. Chúng tôi sẽ không cung cấp ID tài liệu cho yêu cầu này.

 **Liệt kê 5.3 Lập chỉ mục đánh giá phim mà không cần ID**

POST phim\_đánh giá/\_doc

{

"phim":"Bố già", "người dùng":"Peter Piper", "đánh giá":4.5,

**URL không có ID.**

"nhận xét":"Bộ phim bắt đầu bằng một ..."

}

Khi máy chủ thực hiện yêu cầu lập chỉ mục, phản hồi sẽ được gửi trở lại bảng điều khiển Kibana. Hình 5.3 minh họa quá trình này.

{

"\_index" : "đánh giá phim", "\_type" : "\_doc",

"\_id" : " 53NyfXoBW8A1B2amKR5j", "\_version" : 1,

"result" : "đã tạo", "\_shards" : {

**ID là UUID do máy chủ tự động tạo và gán cho bài đánh giá phim.**

**Phản hồi từ máy chủ**

}

"tổng cộng" : 4,

"thành công" : 1,

"thất bại" : 0

},

"\_seq\_no" : 0,

"\_primary\_term" : 1

**Cáckết quảchỉ ra rằng tài liệu đã được lập chỉ mục thành công.**

**Hình 5.3 Máy chủ tạo và gán một ID được tạo tự động ngẫu nhiên chotài liệu.**

Trong phản hồi, trường \_id là dữ liệu được tạo ngẫu nhiên, trong khi phần thông tin còn lại giống với yêu cầu PUT trước đó được hiển thị trong danh sách 5.1. Bạn có thể tự hỏi làm thế nào chúng ta quyết định xem xét động từ nào khi lập chỉ mục tài liệu—phương pháp PUT hay POST. Thanh bên sau đây cung cấp câu trả lời!

**156 CPHẦN5*Làm việc với tài liệu***

**Khi nào sử dụng PUT và/hoặc POST**

Nếu chúng ta muốn kiểm soát ID hoặc chúng ta đã biết ID của tài liệu của mình, chúng ta sử dụngĐẶTphương pháp lập chỉ mục các tài liệu. Chúng có thể là các đối tượng miền của chúng ta với mộtchiến lược nhận dạng được xác định (như khóa chính) mà chúng ta có thể tuân thủ. Việc truy xuất tài liệu bằng ID có thể là lý do để cân nhắc cách học. Nếu chúng ta biết ID tài liệu, chúng ta có thể sử dụng API tài liệu để lấy tài liệu (chúng ta sẽ thảo luận về việc truy xuất sau). Ví dụ, ở đây chúng ta cung cấp ID chobộ phimtài liệu:

POST phim/\_doc/1

{

"phim":"Bố già",

}

Mặt khác, việc sử dụng ID cho các tài liệu có nguồn gốc từ dữ liệu phát trực tuyến hoặc dữ liệu thời gianchuỗi sự kiện không có ý nghĩa (hãy tưởng tượng báo giá bắt nguồn từ máy chủ định giá, biến động giá cổ phiếu, cảnh báo hệ thống từ dịch vụ đám mây, tweet hoặc nhịp tim từ ô tô tự động). Có UUID được tạo ngẫu nhiên là đủ tốt cho các sự kiện và tin nhắn này. Nhưng vì ID được tạo ngẫu nhiên, chúng ta có thể cần viết truy vấn tìm kiếm để truy xuất tài liệu thay vì chỉ truy xuất chúng bằng ID như chúng ta làm đối vớiĐẶTS.

Tóm lại, hãy sử dụng HTTPBƯU KIỆNhành động lập chỉ mục các tài liệu không có doanh nghiệpkhông có danh tính. Ví dụ, chúng tôi không cung cấp ID cho tài liệu đánh giá phim:

POST phim\_đánh giá/\_doc

{

"đánh giá":"Bộ phim Bố già là một kiệt tác..",

}

Khi lập chỉ mục một tài liệu, API lập chỉ mục tài liệu không quan tâm đến việc tài liệu đó có tồn tại hay không. Nếu chúng ta lập chỉ mục lần đầu tiên, tài liệu sẽ được tạo và lưu trữ như mong đợi. Nếu chúng ta lập chỉ mục lại cùng một tài liệu, tài liệu đó sẽ được lưu ngay cả khi nội dung hoàn toàn khác so với tài liệu trước đó. Elasticsearch có chặn hoạt động ghi đè nội dung của tài liệu không? Hãy cùng tìm hiểu.

**TRONG****HÁT\_TẠO ĐỂ TRÁNH GHI ĐÈ MỘT TÀI LIỆU**

Hãy thay đổi hướng đi một chút và xem điều gì xảy ra nếu thực hiện truy vấn sau.

**Liệt kê 5.4 Lập chỉ mục nội dung tài liệu không chính xác**

PUT phim/\_doc/1

{

"tweet":"Elasticsearch in Action 2e đã có mặt!"

**Không phải là một bộ phimnhưng một tweet**

}

Trong ví dụ, chúng ta đang lập chỉ mục một tài liệu có tweet vào một chỉ mục phim có ID tài liệu là 1. Đợi một chút: chúng ta đã có một tài liệu phim (The God-father) có ID đó trong kho lưu trữ của mình chưa? Có chứ. Elasticsearch không kiểm soát được các hoạt động ghi đè như vậy. Trách nhiệm được chuyển xuống cho ứng dụng hoặc người dùng.

***5.1 Lập chỉ mụctài liệu* 157**

Thay vì phụ thuộc vào quyết định của người dùng, có thể dẫn đến dữ liệu không chính xác do ghi đè vô tình, Elasticsearch cung cấp một điểm cuối khác: \_create. Điểm cuối nàyđiểm cuối giải quyết tình huống ghi đè. Chúng ta có thể sử dụng điểm cuối \_create thay vì

\_doc khi lập chỉ mục một tài liệu để tránh ghi đè lên tài liệu hiện có. Hãy cùng xem xét điều này trong thực tế.

**TANH TA\_TẠO NÊNGiao diện lập trình ứng dụng (API)**

Hãy lập chỉ mục cho tài liệu phim có ID là 100, nhưng lần này hãy sử dụng điểm cuối \_create. Danh sách tiếp theo hiển thị lệnh gọi hoạt động.

**Liệt kê 5.5 Lập chỉ mục một bộ phim mới bằng cách sử dụng\_tạo nênđiểm cuối**

PUT phim/\_create/100

{

"title":"Nhiệm vụ bất khả thi", "đạo diễn":"Brian De Palma"

}

Chúng tôi đã lập chỉ mục một bộ phim mới (Nhiệm vụ bất khả thi), lần này sử dụng điểm cuối \_create thay vì \_doc. Sự khác biệt cơ bản giữa hai phương pháp này là

Phương thức \_create không cho phép chúng ta lập chỉ mục lại tài liệu có cùng ID, trong khi \_doc

sẽ không bận tâm.

Tiếp theo, chúng ta hãy thử thay đổi nội dung của tài liệu bằng cách gửi tin nhắn tweet dưới dạng tài liệu phim. Truy vấn được hiển thị ở phần tiếp theo.

**Liệt kê 5.6 Thêm và cập nhật mộtcánh đồng**

PUT phim/\_create/100

{

"tweet":"Xem phim có bỏng ngô thì tuyệt lắm!"

**Lập chỉ mục một tweet thay cho một bộ phim hiện có**

}

Chúng tôi (có thể vô tình) ghi đè nội dung của tài liệu phim. Tuy nhiên, Elasticsearch lại đưa ra lỗi xung đột phiên bản:

{

"loại": "phiên bản\_xung đột\_động cơ\_ngoại lệ",

"lý do":"[100]:xung đột phiên bản,tài liệu đã tồn tại(phiên bản hiện tại[1])"

}

Elasticsearch không cho phép ghi đè dữ liệu. Đây là cách API \_create chỉ ra rằng tài liệu không thể được cập nhật vì phiên bản đã tồn tại.

**GHI CHÚ**Mặc dù điểm cuối \_create không cho phép chúng ta cập nhật tài liệu, chúng ta có thể hoán đổi điểm cuối \_create với \_doc để thực hiện cập nhật nếu cần.

Bài học rút ra cho phần này là nếu chúng ta cần bảo vệ tài liệu của mình bằng cách không cho phép-Để tránh trường hợp ghi đè nhầm, chúng ta nên sử dụng API \_create.

**158 CPHẦN5*Làm việc với tài liệu***

**Vô hiệu hóa việc tự động tạo chỉ mục**

Theo mặc định, Elasticsearch tự động tạo một chỉ mục bắt buộc nếu chỉ mục chưa tồn tại. Nếu chúng ta muốn hạn chế tính năng này, chúng ta cần đặt một cờ có tên làhoạt động

.tự động tạo chỉ mụcĐẾNSAI. Có thể thực hiện điều này theo hai cách:

* Đặt cờ thànhSAItrong tệp cấu hình elasticsearch.yml.
* Đặt cờ một cách rõ ràng bằng cách gọi\_cluster/cài đặtđiểm cuối:

ĐẶT \_cluster/cài đặt

{

"bền bỉ": { "action.auto\_create\_index": "sai"

}

}

Ví dụ, cuộc gọiĐẶT my\_new\_index/\_doc/1thất bại nếuaction.auto\_create\_index chỉ mụcđược thiết lập đểSAI. Bạn có thể muốn thực hiện điều này nếu bạn đã tạo chỉ mụcthủ công (nhiều khả năng là với các thiết lập được xác định trước và lược đồ ánh xạ) và không cần cho phép tạo chỉ mục theo yêu cầu. Chúng tôi sẽ nói thêm về các hoạt động lập chỉ mục trong phần tiếp theo.

Bây giờ chúng ta đã hiểu cách các tài liệu được lưu trữ, hãy cùng xem xét cơ chế lưu trữ của Elasticsearch. Phần tiếp theo tập trung vào cách thức hoạt động của quy trình lập chỉ mục.

* + 1. Cơ chế lập chỉ mục

Chúng tôi đã xem xét sơ qua cách lập chỉ mục hoạt động trong phần 3.2.3. Trong phần này, chúng tôi sẽ xem xét các cơ chế liên quan khi lập chỉ mục một tài liệu (xem hình 5.4). Như chúng tôi đã

**Một chỉ số là một logic**

**bộ sưu tập các mảnh vỡ.**

**Bộ đệm trong bộ nhớ là heap**

**không gian trong nút Lucene (phân đoạn).**

Dữ liệu

Dữ liệu

Trong bộ nhớ

đệm

Hệ thống tập tin

**Tìm kiếm đàn hồi**

**thể hiện (nút)**

**Mảnh vỡ 1**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

Mục lục 1

**Một phiên bản Elasticsearch đang chạy sẽ tạo ra các phiên bản Lucene.**

**Các phân đoạn giữ tài liệuvà được hỗ trợ bởi hệ thống tập tin.**

**Hình 5.4 Cơ chế lập chỉ mục tài liệu**

* 1. ***Lập chỉ mụctài liệu* 159**

biết rằng, các mảnh vỡ là các thể hiện Lucene chứa dữ liệu vật lý được liên kết hợp lý với một chỉ mục.

Khi chúng ta lập chỉ mục một tài liệu, công cụ sẽ quyết định phân đoạn nào mà tài liệu sẽ được lưu trữ dựa trên thuật toán định tuyến (được thảo luận trong phần 3.5). Mỗi phân đoạn đi kèm với bộ nhớ heap và khi một tài liệu được lập chỉ mục, trước tiên tài liệu sẽ được đẩy vào bộ đệm trong bộ nhớ của phân đoạn. Tài liệu được giữ trong bộ đệm trong bộ nhớ này cho đến khi có sự làm mới. Bộ lập lịch của Lucene sẽ phát hành một lần làm mới mỗi giây để thu thập tất cả các tài liệu có sẵn trong bộ đệm trong bộ nhớ, sau đó tạo một phân đoạn mới với các tài liệu này. Phân đoạn bao gồm dữ liệu tài liệu và các chỉ mục đảo ngược. Đầu tiên, dữ liệu được ghi vào bộ đệm hệ thống tệp và sau đó được cam kết vào đĩa vật lý.

Vì các hoạt động I/O tốn kém, Lucene tránh các hoạt động I/O thường xuyên khi ghi dữ liệu vào đĩa. Do đó, nó chờ khoảng thời gian làm mới (một giây), sau đó các tài liệu được đóng gói để đẩy vào các phân đoạn. Khi các tài liệu được chuyển đến các phân đoạn, chúng sẽ có sẵn để tìm kiếm.

Apache Lucene là một thư viện thông minh khi xử lý dữ liệu ghi và đọc. Sau khi đẩy các tài liệu vào một phân đoạn mới (trong quá trình làm mới), nó sẽ đợi cho đến khi ba phân đoạn được hình thành. Nó sử dụng một mẫu hợp nhất ba phân đoạn để hợp nhất các phân đoạn để tạo ra các phân đoạn mới: bất cứ khi nào ba phân đoạn đã sẵn sàng, Lucene sẽ khởi tạo một phân đoạn mới bằng cách hợp nhất chúng. Sau đó, nó sẽ đợi thêm ba phân đoạn nữa được tạo ra để có thể tạo một phân đoạn mới, v.v. Cứ sau ba phân đoạn hợp nhất để tạo ra một phân đoạn khác, như thể hiện trong hình 5.5.

Phân đoạn cấp độ 0

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

Phân đoạn cấp độ 1

Phân đoạn cấp độ 2



**Phân đoạn**

**Hình 5.5 Minh họa cách các phân đoạn được hợp nhấttrong Apache Lucene**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

**Phân đoạn**

**160 CPHẦN5*Làm việc với tài liệu***

Trong khoảng thời gian làm mới một giây, một phân đoạn mới được tạo để chứa bất kỳ số lượng tài liệu nào được thu thập trong bộ đệm trong bộ nhớ. Bộ nhớ heap của phân đoạn (thể hiện Lucene) quyết định số lượng tài liệu mà nó có thể chứa trong bộ đệm trong bộ nhớ trước khi đưa chúng ra kho lưu trữ tệp. Đối với tất cả các mục đích thực tế, Lucene coi các phân đoạn là tài nguyên không thay đổi. Nghĩa là, sau khi một phân đoạn được tạo bằng các tài liệu có sẵn từ bộ đệm, sẽ không có tài liệu mới nào được đưa vào phân đoạn hiện có này. Thay vào đó, chúng được chuyển vào một phân đoạn mới. Tương tự như vậy, các lệnh xóa không được thực hiện vật lý trên các tài liệu trong phân đoạn, nhưng các tài liệu được đánh dấu để xóa sau. Lucene sử dụng chiến lược này để cung cấp hiệu suất và thông lượng cao.

* + 1. Tùy chỉnh quá trình làm mới

Các tài liệu được lập chỉ mục sẽ tồn tại trong bộ nhớ cho đến khi chu kỳ làm mới bắt đầu. Điều này có nghĩa là chúng ta có các tài liệu chưa được cam kết (không bền) cho đến chu kỳ làm mới tiếp theo. Sự cố máy chủ có thể gây ra mất dữ liệu. Rủi ro mất dữ liệu tăng lên khi chu kỳ làm mới tăng lên (thời gian chu kỳ làm mới một giây có nguy cơ mất dữ liệu thấp hơn thời gian chu kỳ làm mới một phút). Mặt khác, việc giảm thời gian làm mới sẽ dẫn đến nhiều hoạt động I/O hơn, có thể gây ra tình trạng tắc nghẽn hiệu suất—do đó, chúng ta phải tìm ra chiến lược làm mới tối ưu cho tổ chức của mình.

Elasticsearch là một ứng dụng nhất quán cuối cùng—có nghĩa là các tài liệu cuối cùng được ghi vào kho lưu trữ bền vững. Các tài liệu được chuyển vào hệ thống tệp dưới dạng các phân đoạn trong quá trình làm mới và do đó có thể tìm kiếm được. Quá trình làm mới tốn kém, đặc biệt là nếu công cụ bị quá tải với số lượng lớn các yêu cầu lập chỉ mục.

**CHÌNH THÀNH CHU KỲ LÀM MỚI**

Tin tốt là chúng ta có thể cấu hình thiết lập làm mới này. Chúng ta có thể thiết lập lại khoảng thời gian từ mặc định 1 giây thành, chẳng hạn, 60 giây bằng cách điều chỉnh các thiết lập trên một chỉ mụcmức độ sử dụng điểm cuối \_settings.

**Liệt kê 5.7 Thiết lập khoảng thời gian làm mới tùy chỉnh**

PUT phim/\_cài đặt

{

"chỉ số":{ "khoảng thời gian làm mới":"60 giây"

}

}

Đây là một thiết lập động, nghĩa là chúng ta có thể thay đổi thiết lập làm mới trên một chỉ mục trực tiếp bất kỳ lúc nào. Để tắt hoàn toàn hoạt động làm mới, hãy đặt giá trị thành -1. Bộ đệm trong bộ nhớ sẽ tích lũy các tài liệu đến nếu hoạt động làm mới bị tắt. Trường hợp sử dụng cho kịch bản này có thể là chúng ta đang di chuyển nhiều tài liệu từ cơ sở dữ liệu vào Elasticsearch và chúng ta không muốn dữ liệu có thể tìm kiếm được cho đến khi

* 1. ***Lấy lạitài liệu* 161**

di chuyển hoàn tất thành công. Để bật lại thủ công làm mới trên chỉ mục, chúng tôi mô phỏngply phát lệnh POST <index>/\_refresh.

**CLIÊN KẾT****-ĐIỀU KHIỂN LÀM MỚI BÊN**

Chúng ta cũng có thể kiểm soát hoạt động làm mới từ phía máy khách đối với các hoạt động CRUD trên tài liệu bằng cách thiết lập tham số truy vấn làm mới. Các API tài liệu (chỉ mục,delete, update và \_bulk) mong đợi làm mới như một tham số truy vấn. Ví dụ, đoạn mã sau đây khuyên công cụ bắt đầu làm mới sau khi tài liệu đã được lập chỉ mục thay vì chờ khoảng thời gian làm mới hết hạn:

PUT phim/\_doc/1?làm mới

Tham số truy vấn làm mới có thể có ba giá trị:

* + - * làm mới = sai*(mặc định)*—Yêu cầu công cụ không ép buộc thao tác làm mới mà thay vào đó áp dụng cài đặt mặc định (một giây). Công cụ chỉ cho phép tìm kiếm tài liệu sau khoảng thời gian làm mới được xác định trước. Chúng tôi cũng không thể cung cấp tham số truy vấn, điều này cũng đạt được hiệu ứng tương tự.

Ví dụ:PUT phim/\_doc/1?refresh=false

* + - * làm mới = đúng*(hoặc chuỗi rỗng)*—Buộc thực hiện thao tác làm mới để tài liệu có thể nhìn thấy để tìm kiếm ngay lập tức. Nếu khoảng thời gian làm mới của chúng tôi được đặt thành 60 giây và chúng tôi lập chỉ mục 1.000 tài liệu vớilàm mới = đúng, tất cả 1.000 tài liệu sẽ có thể được tìm kiếm ngay lập tức thay vì phải chờ khoảng thời gian làm mới 60 giây.

Ví dụ:PUT phim/\_doc/1?refresh=true

* + - * làm mới=chờ\_đợi—Yêu cầu chặn buộc máy khách phải đợi cho đến khi hoạt động làm mới bắt đầu và hoàn tất trước khi yêu cầu được trả về. Ví dụ, nếu khoảng thời gian làm mới của chúng tôi là 60 giây, yêu cầu sẽ bị chặn trong 60 giây cho đến khi thực hiện làm mới. Tuy nhiên, có thể bắt đầu thủ công bằng cách gọiPOST <index>/\_refreshđiểm cuối.

Ví dụ:PUT phim/\_doc/1?refresh=wait\_for

Chúng ta cũng cần hiểu cơ chế truy xuất tài liệu. Elasticsearch cung cấp một API GET để đọc tài liệu, tương tự như các API lập chỉ mục mà chúng ta đã thấy trước đó. Phần tiếp theo sẽ xem xét cơ chế đọc tài liệu từ stash Elasticsearch.

#### Lấy lại tài liệu

Elasticsearch cung cấp hai loại API tài liệu để truy xuất tài liệu:

* + - Một API tài liệu đơn trả về một tài liệu, được cung cấp một ID
    - Một API đa tài liệu trả về nhiều tài liệu, được cung cấp một mảng ID

Nếu tài liệu không khả dụng, chúng tôi sẽ nhận được phản hồi JSON cho biết tài liệu không được tìm thấy. Hãy xem cách truy xuất tài liệu bằng cả hai API.

**162 CPHẦN5*Làm việc với tài liệu***

* + 1. Sử dụng API tài liệu đơn

Elasticsearch cung cấp API RESTful để lấy một tài liệu được cung cấp ID tài liệu, tương tự như API lập chỉ mục mà chúng ta đã thảo luận ở phần trước. Định nghĩa API để lấy một tài liệu duy nhất là

NHẬN <index\_name>/\_doc/<id>

GET là phương thức HTTP cho biết chúng ta đang lấy một tài nguyên. URL cho biếtđiểm cuối của tài nguyên—trong trường hợp này là index\_name theo sau là \_doc và ID tài liệu.

Như bạn có thể nhận thấy, sự khác biệt giữa lập chỉ mục tài liệu và tìm nạp chỉ là động từ HTTP: sửa đổi PUT/POST thành GET. Không có thay đổi nào đối với URL, tuân theo các thông lệ tốt nhất của dịch vụ RESTful.

Hãy lấy lại tài liệu phim có ID là 1 mà chúng ta đã lập chỉ mục trước đó. Thực hiện lệnh GET movies/\_doc/1 trên bảng điều khiển Kibana để lấy tài liệu đã lập chỉ mục trước đó. Phản hồi JSON được hiển thị trong hình 5.6.

**Phản hồi từ máy chủ**

**Lấy lại tài liệu phim có ID1**

NHẬN phim/\_doc/1

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "1",

"\_phiên bản" : 1,

"\_seq\_no" : 0,

"\_primary\_term" : 1, "found" : đúng,

"\_nguồn" : {

**"title": "Bố già",**

**Siêu dữ liệu tài liệu**

**Tài liệu gốc được đính kèm trong\_nguồnsự vật.**

**"tóm tắt": "Người cha già**

**Nguyên bản**

**tài liệu**

**của một tội phạm có tổ chức.."**

**}**

}

**Hình 5.6 Lấy lại một tài liệu bằng cách sử dụngLẤYGọi API**

Phản hồi có hai phần: siêu dữ liệu và tài liệu gốc. Siêu dữ liệu bao gồm \_id, \_type, \_version, v.v. Tài liệu gốc được đính kèm trongThuộc tính \_source. Vậy là xong! Chỉ cần biết ID của tài liệu là có thể lấy được tài liệu đó một cách đơn giản.

Tài liệu có thể không tồn tại trong cửa hàng. Nếu không tìm thấy tài liệu, chúng tôi sẽ nhận đượcphản hồi với thuộc tính found được đặt thành false. Ví dụ, cố gắng tìm một tài liệu có ID 999 (không tồn tại trong hệ thống của chúng tôi) trả về phản hồi sau:

***5.2 Truy xuấttài liệu* 163**

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "999",

"tìm thấy" : sai

}

Tất nhiên chúng ta có thể tìm hiểu xem tài liệu có tồn tại trong kho lưu trữ trước hay không bằng cách sử dụng API đọc nhưng với hành động HTTP HEAD trên URL tài nguyên. Ví dụ, truy vấn sau đây kiểm tra xem phim có ID 1 có tồn tại hay không.

**Liệt kê 5.8 Kiểm tra xem một tài liệutồn tại**

Phim HEAD/\_doc/1

Truy vấn này trả về 200—OK nếu tài liệu tồn tại. Nếu tài liệu không khả dụng trongcửa hàng, lỗi 404 Không tìm thấy được trả về cho khách hàng, như hình 5.7 hiển thị.

Phim HEAD/\_doc/99

{

"mã trạng thái":404, "lỗi":"Không tìm thấy",

"message":"404 - Không tìm thấy"

}

**Kiểm tra xem một tài liệu không tồn tại có tồn tại hay không (một tài liệu có ID99không tồn tại trong cửa hàng của chúng tôi)**

**Chúng tôi nhận được mộtKhông tìm thấythông báo lỗi (phản hồi định dạng JSON) như tài liệu không có sẵn trong cửa hàng.**

**Hình 5.7 Việc lấy một tài liệu không tồn tại sẽ trả về thông báo “Không tìm thấy”.**

Chúng ta có thể gửi yêu cầu HEAD đến máy chủ để xác định xem tài liệu có tồn tại hay không trước khi yêu cầu. Nếu bạn nghĩ rằng điều này sẽ phát sinh thêm một chuyến khứ hồi đến máy chủ, thì thực tế là có. Dựa trên phản hồi của yêu cầu HEAD, chúng ta có thể (hoặc không) phải gửi một yêu cầu khác để lấy đối tượng thực tế. Thay vào đó, chúng ta có thể sử dụng yêu cầu GET để bắt đầu, yêu cầu này sẽ trả về tài liệu nếu nó tồn tại hoặc thông báo Không tìm thấy nếu không. Tất nhiên, tùy bạn lựa chọn.

Cho đến nay, chúng ta chỉ lấy một tài liệu duy nhất cho một chỉ mục duy nhất. Làm thế nào chúng ta có thể đáp ứng yêu cầu lấy nhiều tài liệu có ID từ cùng một chỉ mục hoặc nhiều chỉ mục? Ví dụ, làm thế nào chúng ta có thể lấy hai tài liệu có ID 1 và 2 từ chỉ mục phim? Chúng ta có thể sử dụng API nhiều tài liệu có tên là \_mget, đây là chủ đề của phần tiếp theo.

* + 1. Lấy lại nhiều tài liệu

Trong phần trước, chúng tôi đã sử dụng API một tài liệu để lấy từng tài liệu một. Tuy nhiên, chúng tôi có thể có các yêu cầu như sau:

**164 CPHẦN5*Làm việc với tài liệu***

* Lấy danh sách các tài liệu từ một chỉ mục, dựa trên ID tài liệu.
* Truy xuất danh sách các tài liệu từ nhiều chỉ mục, dựa trên ID tài liệu.

Elasticsearch cung cấp API đa tài liệu (\_mget) để đáp ứng các yêu cầu này. Ví dụ, để lấy danh sách các tài liệu được cung cấp ID, chúng ta có thể sử dụng API \_mget như được hiển thị trong danh sách sau.

**Liệt kê 5.9 Lấy nhiều tài liệu cùng một lúc**

NHẬN phim/\_mget

{

"id": ["1", "12", "19", "34"]

}

Hình 5.8 cho thấy định dạng của lệnh gọi để lấy nhiều tài liệu từ nhiều chỉ mục khác nhau. Như bạn có thể thấy, điểm cuối \_mget được cung cấp với định dạng JSONđối tượng yêu cầu. Khóa docs bên trong yêu cầu mong đợi một mảng các cặp \_index và \_id tài liệu mà chúng ta có thể sử dụng để lấy tài liệu từ nhiều chỉ mục. Mã trong danh sách tiếp theo lấy tài liệu từ ba chỉ mục khác nhau.

**Các\_mgetđiểm cuối tiếp nhận một yêu cầu cơ thể với mộttài liệusự vật.**

NHẬN \_mget

{

"tài liệu":[

{

"\_index":"index\_1", "\_id":88

},

{

"\_index":"index\_2", "\_id":99

**Cáctài liệuđối tượng bao gồm**

**của một mảng các đối tượng yêu cầu.**

**Mỗi đối tượng yêu cầu riêng lẻ bao gồm chỉ mục (\_chỉ mục) và ID (\_nhận dạng) của tài liệu.**

}

]

}

**Hình 5.8 Lấy nhiều tài liệu bằng cách sử dụng\_mgetGiao diện lập trình ứng dụng (API)**

 **Liệt kê 5.10 Lấy tài liệu từ ba chỉ mục khác nhau**

NHẬN \_mget

{

"tài liệu":[

{

**\_mget call không có chỉ mục nào được đề cập trong URL**

"\_index":"phim\_cổ\_điển","\_id":11

},

**Chỉ số đầu tiên**

**được cung cấp ở đây.**

* 1. ***Lấy lạitài liệu* 165**

{

"\_index":"phim\_quốc\_tế","\_id":22

},

{

"\_index":"top100\_phim",  "\_id":33

}

]

}



**Mục lục 3**

**Mục lục 2**

Yêu cầu được hình thành với yêu cầu lấy ba tài liệu từ ba chỉ mục khác nhau:phim kinh điển,phim\_quốc\_tế, Vàtop100\_phim. Lưu ý rằng nếu chỉ mục không tồn tại, chúng ta sẽ nhận được mộtngoại lệ index\_not\_found.

Chúng tôi có thể cung cấp nhiều chỉ mục tùy theo yêu cầu, mặc dù API này có nhược điểm: chúng tôi phải tạo một \_index riêng lẻ/\_id cặp cho mỗi ID. Thật không may, Elasticsearchvẫn chưa cho phép thuộc tính \_id chấp nhận một mảng ID. Chúng ta có thể hy vọng rằng những người của Elastic sẽ triển khai tính năng này trong tương lai gần.

* + 1. Truy vấn id

Chúng tôi đã xem xét việc sử dụng API \_mget để lấy nhiều tài liệu trong phần trước. Tuy nhiên, có một cách khác để lấy nhiều tài liệu: sử dụng truy vấn ids (viết tắt của IDs). Truy vấn tìm kiếm đơn giản này lấy một tập hợp ID tài liệu để trả vềcác tài liệu. Truy vấn id có sẵn như một phần của API tìm kiếm. Chúng tôi thảo luận chi tiết về API này trong các chương từ 8 đến 10, nhưng đây là truy vấn đang hoạt động nếu bạn tò mò.

**Liệt kê 5.11 Sử dụng mộtidtruy vấn để lấy nhiều tài liệu**

NHẬN classic\_movies/\_search

{

"truy vấn": {

"ID": {

"giá trị": [1,2,3,4]

}

}

}

Chúng ta cũng có thể lấy từ nhiều chỉ mục bằng cách thêm các chỉ mục vào URL. Sau đây là một ví dụ:

NHẬN classic\_movies,international\_movies/\_search

{

# thân hình

}

Như vậy là đã xong cách lấy nhiều tài liệu từ một hoặc nhiều chỉ mục. Bây giờ, chúng ta hãy chuyển trọng tâm sang các phản hồi. Bạn có nhận thấy rằng các phản hồi của chúng ta (xem hình 5.6) có siêu dữ liệu cùng với tài liệu nguồn gốc không? Nếu chúng ta muốnchỉ lấy nguồn mà không có siêu dữ liệu? Hoặc nếu chúng ta muốn ẩn một số thông tin nhạy cảm

**166 CPHẦN5*Làm việc với tài liệu***

thông tin trong tài liệu nguồn khi trả lại cho khách hàng? Chúng ta có thể thao tác các phản hồi dựa trên yêu cầu của mình, như chúng ta thảo luận trong phần tiếp theo.

#### Thao tác phản hồi

Phản hồi trả về cho máy khách có thể chứa nhiều thông tin và máy khách có thể không muốn nhận tất cả thông tin đó. Và đôi khi có thể có thông tin nhạy cảm không được tiết lộ trong nguồn được gửi lại dưới dạng phản hồi. Ngoài ra, việc gửi một lượng lớn dữ liệu dưới dạng phản hồi (ví dụ, nếu nguồn có 500 thuộc tính) là lãng phí băng thông! Có nhiều cách để thao tác phản hồi trước khi gửi chúng đến máy khách. Đầu tiên, hãy truy xuất nguồn của tài liệu mà không có siêu dữ liệu.

* + 1. Xóa siêu dữ liệu khỏi phản hồi

Thông thường, đối tượng phản hồi bao gồm siêu dữ liệu và tài liệu gốc (nguồn). Thuộc tính đáng chú ý trong phản hồi là thuộc tính \_source, thuộc tính này bao gồm tài liệu đầu vào gốc. Chúng ta có thể chỉ lấy nguồn (tài liệu gốc) mà không cần siêu dữ liệu bằng cách đưa ra truy vấn như sau:

NHẬN <index\_name>/\_source/<id>

Lưu ý rằng điểm cuối \_doc được thay thế bằng \_source; mọi thứ khác trong lệnh gọi vẫn giữ nguyên. Hãy lấy tài liệu phim bằng cách sử dụng điểm cuối \_source này.

**Liệt kê 5.12 Lấy tài liệu gốc không có siêu dữ liệu**

NHẬN phim/\_nguồn/1

Như phản ứng trong hình 5.9 chỉ ra,tài liệu chúng tôi lập chỉ mục được trả về mà không có thông tin bổ sung nào. Không có trường siêu dữ liệu như \_version, \_id hoặc \_index; chỉ có tài liệu nguồn gốc.

NHẬN phim/\_nguồn/1 



**Các\_nguồnĐiểm cuối cho phép chúng ta lấy tài liệu gốc mà không cần siêu dữ liệu.**

{

"title": "Bố già",

"tóm tắt": "Người cha già đang già đi .."

}

**Chỉ trả lại tài liệu gốc; siêu dữ liệu sẽ bị xóa.**

**Hình 5.9Các\_nguồnĐiểm cuối trả về tài liệu gốc không có siêu dữ liệu.**

Còn nếu chúng ta muốn lấy siêu dữ liệu nhưng không lấy tài liệu nguồn thì sao? Chắc chắn, chúng ta cũng có thể làm được. Hãy xem cách chúng ta có thể xóa dữ liệu nguồn.

***5.3 Thao tácphản hồi* 167**

* + 1. Xóa bỏ tài liệu nguồn

Có thể có những trường hợp tài liệu được tải với hàng trăm trường: ví dụ, một tweet đầy đủ (từ Twitter API) bao gồm nhiều hơn một tweet—nó cóhàng chục thuộc tính như tweet, tác giả, dấu thời gian, cuộc trò chuyện, tệp đính kèm, v.v. Khi truy xuất dữ liệu từ Elasticsearch, đôi khi chúng ta không muốn xem dữ liệu nguồn chút nào—chúng ta muốn ẩn hoàn toàn dữ liệu nguồn và chỉ trả về siêu dữ liệu liên quan đến phản hồi. Trong trường hợp đó, chúng ta có thể đặt trường \_source thành false làm tham số yêu cầu trong truy vấn của mình.

**Liệt kê 5.13 Xóa bỏ nguồn gốc ban đầudữ liệu**

NHẬN phim/\_doc/1?\_source=false

Phản hồi cho truy vấn này được hiển thị trong hình 5.10. Lệnh đặt cờ \_source thành false, chỉ ra cho máy chủ không trả về tài liệu gốc. Như bạn có thể thấy từ phản hồi, chỉ có siêu dữ liệu được trả về, không phải tài liệu nguồn. Không lấy tài liệu gốc cũng giải phóng băng thông.

NHẬN phim/\_doc/1?\_source=false

**Thiết lập\_nguồnĐẾNSAI**

**không trả lại bất kỳ tài liệu gốc nào.**

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "1",

"\_phiên bản" : 1,

"\_seq\_no" : 0,

"\_primary\_term" : 1, "found" : đúng

}

**Siêu dữ liệu của tài liệu được trả về và nguồn gốc ban đầu bị xóa.**

**Hình 5.10 Chỉ trả về siêu dữ liệu**

Bây giờ chúng ta biết cách tránh lấy lại toàn bộ tài liệu, nhưng còn việc trả lại một tài liệu có các trường chọn lọc (bao gồm hoặc loại trừ) thì sao? Ví dụ, chúng ta có thểmuốn tiêu đề và xếp hạng của một bộ phim được trả về, nhưng không phải tóm tắt. Làm thế nào chúng ta có thể tùy chỉnh những trường nào được bao gồm hoặc loại trừ trong danh sách trả về? Chúng ta sẽ xem cách thực hiện điều đó trong phần tiếp theo.

* + 1. Bao gồm và loại trừ các trường

Ngoài việc ngăn chặn\_nguồntrường, chúng ta có thể bao gồm và/hoặc loại trừ các trườngkhi lấy tài liệu. Điều này được thực hiện bằng cách sử dụng\_nguồn\_bao gồmVà\_nguồn\_ loại trừcác thông số, tương tự như\_nguồntham số chúng tôi đã sử dụng trước đó. Chúng tôi có thể cung cấp\_nguồn\_bao gồmthuộc tính với danh sách các trường được phân tách bằng dấu phẩy mà chúng tôi muốn

**168 CPHẦN5*Làm việc với tài liệu***

để trả về. Tương tự như vậy, chúng ta có thể sử dụng \_source\_excludes để loại trừ các trường khỏi phản hồi; không có gì ngạc nhiên ở đây.

Chúng tôicó thể cần phải cải thiện tài liệu phim của chúng ta cho ví dụ này, vì tài liệu hiện tại không có nhiều hơn hai trường. Hãy thêm một vài trường bổ sung cho bộ phim thứ ba (The Shawshank Redemption).

**Danh sách 5.14 Mớibộ phimtài liệu có các thuộc tính bổ sung**

PUT phim/\_doc/3

{

"title":"Nhà tù Shawshank",

**Lập chỉ mục một tài liệu mới**

"synopsis":"Hai người đàn ông bị giam cầm gắn kết với nhau ..", "đánh giá":9.3,

"giấy chứng nhận":"15",

"thể loại":"chính kịch",

"diễn viên":["Morgan Freeman","Tim Robbins"]

}

**Lĩnh vực mới:**

**thuộc tính xếp hạng**

**Trường mới: thuộc tính chứng chỉ**

Sau khi tài liệu này được lập chỉ mục, chúng ta có thể thử nghiệm với các trường có nên hoặc không nên trả về trong phản hồi.

**TÔIBAO GỒM CÁC TRƯỜNG SỬ DỤNG****\_NGUỒN\_BAO GỒM**

Để bao gồm danh sách các trường tùy chỉnh, hãy thêm tham số \_source\_includes vàocác trường được phân cách bằng dấu phẩy. Giả sử chúng ta muốn lấy các trường tiêu đề, xếp hạng và thể loại từ chỉ mục phim của mình trong phản hồi và loại bỏ các trường khác. Chúng ta có thể thực hiện lệnh sau.

**Liệt kê 5.15 Bao gồm một số trường có chọn lọc**

NHẬN phim/\_doc/3?\_source\_includes=tiêu đề,đánh giá,thể loại

Lệnh này trả về tài liệu có ba trường này, lọc bỏ phần còn lại:

{

...

"\_nguồn" : {

"xếp hạng" : 9.3,

"thể loại": "kịch",

"title": "Nhà tù Shawshank"

}

}

Phản hồi này có cả thông tin tài liệu gốc (dưới đối tượng \_source) và siêu dữ liệu liên quan. Chúng ta cũng có thể chạy lại truy vấn bằng cách sử dụng \_sourceđiểm cuối thay vì \_doc để loại bỏ siêu dữ liệu và lấy tài liệu có các trường tùy chỉnh.

**Liệt kê 5.16 Trả về các trường chọn lọc không có siêu dữ liệu**

NHẬN phim/\_source/3?\_source\_includes=tiêu đề,đánh giá,thể loại

* 1. ***Thao tácphản hồi* 169**

Tương tự như vậy, chúng ta có thể loại trừ một số trường trong khi trả về phản hồi bằng cách sử dụngtham số \_source\_excludes.

**VÀ****XCLUDING CÁC TRƯỜNG SỬ DỤNG\_NGUỒN\_KHÔNG BAO GỒM**

Chúng ta có thể loại trừ các trường mà chúng ta không muốn được trả về trong phản hồi bằng cách sử dụng

Tham số \_source\_excludes. Đây là tham số đường dẫn URL chấp nhận các trường được phân cách bằng dấu phẩy. Phản hồi bao gồm tất cả các trường tài liệu trừ các trườngđược đề cập trong tham số \_source\_excludes.

**Liệt kê 5.17 Loại trừ các trường trong phản hồi**

NHẬN phim/\_source/3?\_source\_excludes=diễn viên,tóm tắt

Ở đây, các trường actor và synopsis bị loại trừ khỏi phản hồi. Nếu chúng ta muốn bao gồm một số trường và cũng loại trừ một số trường một cách rõ ràng thì sao? Truy vấn Elasticsearch có thể hỗ trợ chức năng này không? Chắc chắn rồi—chúng ta có thể yêu cầu Elasticsearch đáp ứng các yêu cầu này, như đã thảo luận trong phần tiếp theo.

**TÔICÁC TRƯỜNG BAO GỒM VÀ LOẠI TRỪ**

Chúng ta có thể kết hợp và khớp các thuộc tính trả về mà chúng ta muốn, vì Elasticsearch cho phép chúng tađể tinh chỉnh phản hồi. Để chứng minh, chúng ta hãy tạo một tài liệu phim mới với nhiều xếp hạng khác nhau (amazon, metacritic và rotten\_tomatoes).

**Danh sách 5.18 Mô hình phim mới có xếp hạng**

PUT phim/\_doc/13

{

"title":"Hình đại diện",

"xếp hạng":9.3, "xếp hạng\_amazon":4.5, "xếp hạng\_rotten\_tomatoes":80, "xếp hạng\_metacritic":90

}

Làm thế nào chúng ta có thể trả về tất cả các xếp hạng ngoại trừamazon? Đây là nơi sức mạnh của việc thiết lập tỏa sáng\_nguồn\_bao gồmVà\_nguồn\_loại\_trừvới các thuộc tính phù hợp.

**Liệt kê 5.19 Bỏ qua một số trường nhất định một cách có chọn lọc**

NHẬN phim/\_source/13?\_source\_includes=rating\*&\_source\_excludes=rating\_amazon

Câu hỏi và câu trả lời được thể hiện ở hình 5.11.

Trong truy vấn này, chúng tôi kích hoạt một trường ký tự đại diện,\_source\_includes=xếp hạng\*, để lấy tất cả các thuộc tính được thêm tiền tố bằng từ xếp hạng (xếp hạng,đánh giá\_amazon,xếp hạng

\_siêu phê bình,đánh giá\_cà\_cà\_chà\_thối). Các\_nguồn\_loại\_trừMặt khác, tham số sẽ ngăn chặn một trường (ví dụ:\_source\_excludes=xếp hạng\_amazon). Tài liệu kết quả phải bao gồm tất cả các xếp hạng ngoại trừ xếp hạng của Amazon.

**170 CPHẦN5*Làm việc với tài liệu***

**Các\_nguồn\_loại\_trừtham số không cho phép xếp hạng của Amazon (đánh giá\_amazon).**

NHẬN phim/\_source/13?\_source\_includes=rating\*&\_source\_excludes=rating\_amazon

**Các\_nguồn\_bao gồm**

**tham số cho phép tất cả các xếp hạng do ký tự đại diện (đánh giá\*).**

{

"xếp hạng": "9.3",

"rating\_rotten\_tomatoes" : 80,

"rating\_metacritic" : 90

}

**Đầu ra bao gồm tất cả các xếp hạng nhưng không có trường xếp hạng của Amazon.**

**Hình 5.11 Điều chỉnh các thuộc tính nào là và không phải là một phần của kết quả trả về**

Cho đến nay, chúng ta đã thấy cách tạo và đọc tài liệu, bao gồm cả việc thao tác phản hồi. Bây giờ chúng ta hiểu cơ chế cập nhật tài liệu. Sẽ luôn có nhu cầu cập nhật tài liệu hiện có bằng cách sửa đổi giá trị của trường hiện có hoặc thêm trường mới. Elasticsearch cung cấp một bộ API cập nhật cho mục đích này, được thảo luận trong phần tiếp theo.

#### Đang cập nhật tài liệu

Đôi khi, các tài liệu được lập chỉ mục cần được cập nhật bằng các giá trị đã sửa đổi hoặc các trường bổ sung hoặc toàn bộ tài liệu có thể cần được thay thế. Tương tự như lập chỉ mục tài liệu, Elasticsearch cung cấp hai loại truy vấn cập nhật—một loại để làm việc với các tài liệu đơn lẻ và loại còn lại để làm việc với nhiều tài liệu:

* Các\_cập nhậtAPI cập nhật một tài liệu duy nhất.
* \_cập\_nhật\_theo\_truy\_cậpcho phép chúng ta sửa đổi nhiều tài liệu cùng một lúc.

Trước khi xem một số ví dụ, chúng ta cần hiểu cơ chế liên quan khi cập nhật tài liệu. Chúng ta hãy làm điều đó tiếp theo.

* + 1. Cơ chế cập nhật tài liệu

Elasticsearch yêu cầu một vài bước khi chúng ta cập nhật tài liệu. Hình 5.12 minh họa quy trình: Đầu tiên, Elasticsearch sẽ lấy tài liệu, sửa đổi tài liệu, sau đó lập chỉ mục lại tài liệu. Về cơ bản, nó thay thế tài liệu cũ bằng một tài liệu mới. Đằng sau hậu trường, Elasticsearch tạo một tài liệu mới để cập nhật. Trong quá trình cập nhật này, Elasticsearch sẽ tăng phiên bản của tài liệu sau khi hoạt động cập nhật hoàn tất. Khi phiên bản mới hơn của tài liệu (với các giá trị đã sửa đổi hoặc các trường mới) đã sẵn sàng, nó sẽ đánh dấu phiên bản cũ hơn để xóa.

***5.4 Cập nhậttài liệu* 171**

2. Sửa đổi/thêm trường.

1. Lấy tài liệu đã cho.

**Hình 5.12 Việc cập nhật hoặc sửa đổi tài liệu là một quá trình gồm ba bước.**

3. Lập lại chỉ mục cho tài liệu đã cập nhật.

Nếu bạn nghĩ chúng ta có thể thực hiện cùng một bản cập nhật bằng cách gọi các phương thức GET, UPDATE và POST trên tài liệu riêng lẻ, thì bạn hoàn toàn đúng. Trên thực tế, đây chính là những gì Elasticsearch thực hiện. Như bạn có thể hình dung, đây là ba lệnh gọi khác nhau đến máy chủ, mỗi lệnh dẫn đến một chuyến khứ hồi từ máy khách đến máy chủ. Elasticsearch tránh được chuyến khứ hồi này bằng cách khéo léo thực hiện tập hợp các hoạt động trên cùng một phân đoạn, do đó tiết kiệm lưu lượng mạng giữa máy khách và máy chủ. Bằng cách sử dụng API \_update, Elasticsearch tránh được các lệnh gọi mạng, băng thông và lỗi mã hóa.

* + 1. API \_cập nhật

Khi chúng tôi có kế hoạch cập nhật một tài liệu, chúng tôi thường tập trung vào một hoặc nhiều tình huống sau:

* + - * Thêm nhiều trường hơn vào một tài liệu hiện có
      * Sửa đổi các giá trị trường hiện có
      * Thay thế toàn bộ tài liệu

Tất cả các hoạt động này được thực hiện bằng cách sử dụng API \_update. Định dạng rất đơn giản:

POST <tên\_chỉ\_mục>/\_cập\_nhật/<id>

Khi quản lý tài nguyên của chúng tôi bằng API \_update, chúng tôi sử dụng phương thức POST dựa trên các quy ước API RESTful. ID tài liệu được cung cấp trong URL cùng với tên chỉ mục để tạo thành URL đầy đủ.

**GHI CHÚ** Trước đócập nhật URL sử dụng\_cập nhậtđiểm cuối, nhưng chúng ta cũng có thểsử dụngPOST <tên\_chỉ\_mục>/\_doc/<id>/\_cập\_nhật. Bởi vì các kiểu đã bị loại bỏ và bị xóa khỏi phiên bản 8, hãy tránh sử dụng\_tài liệuđiểm cuối để cập nhật tài liệu.

Bây giờ chúng ta đã biết những điều cơ bản của\_cập nhậtAPI, hãy cập nhậtbộ phimtài liệu (Bố già) có thêm một vài thuộc tính:diễn viênVàgiám đốc.

**MỘT****DDING CÁC LĨNH VỰC MỚI**

Để sửa đổi một tài liệu có các trường mới, chúng ta chuyển một thân yêu cầu bao gồm các trường mới đến API \_update. Các trường mới được gói trong một đối tượng doc như API mong đợi. Mã trong danh sách sau đây sửa đổi tài liệu phim bằng hai trường bổ sung. Lưu ý rằng chúng ta ngầm định giả định rằng dynamic=true.

**172 CPHẦN5*Làm việc với tài liệu***

**Liệt kê 5.20 Thêm các trường vào tài liệu bằng cách sử dụng\_cập nhậtGiao diện lập trình ứng dụng (API)**

POST phim/\_cập nhật/1

{

"tài liệu": {

"diễn viên":["Marlon Brando","Al Pacino","James Caan"], "đạo diễn":"Francis Ford Coppola"

}

}

Truy vấn này thêmdiễn viênVàgiám đốccác lĩnh vực của chúng tôibộ phimtài liệu. Nếu chúng ta lấy tài liệu (NHẬN phim/\_doc/1), các trường bổ sung có sẵn trong tài liệu trả về.

**TôiPHÂN BIỆT CÁC LĨNH VỰC HIỆN CÓ**

Đôi khi chúng ta có thể cần thay đổi các trường hiện có. Làm như vậy không phức tạp; chúng ta chỉ cầnphải cung cấp giá trị mới cho trường trong đối tượng doc như chúng ta đã làm trong truy vấn trước. Ví dụ, để đổi tên trường tiêu đề, chúng ta viết truy vấn như truy vấn được hiển thị trong danh sách tiếp theo.

**Liệt kê 5.21 Cập nhật tiêu đề của một tài liệu hiện có**

POST phim/\_cập nhật/1

{

"tài liệu": {

"title":"Bố già (Bản gốc)"

}

}

Khi cập nhật một phần tử trong một mảng (như thêm một diễn viên mới vào danh sách trong trường diễn viên), chúng ta phải cung cấp cả giá trị mới và cũ. Ví dụ, hãyGiả sử chúng ta muốn thêm một diễn viên khác (Robert Duvall) vào trường diễn viên cho tài liệu Cha đỡ đầu của chúng ta.

**Liệt kê 5.22 Cập nhật trường hiện có với thông tin bổ sung**

POST phim/\_cập nhật/1



{

"tài liệu": {

"diễn viên":["Marlon Brando",

"Al Pacino", "James Caan", "Robert Duvall"]

}

**Chúng tôi đang cập nhậttài liệu có ID 1.**

**Các bản cập nhật phải được bao gồm trong đối tượng doc.**

**Giá trị cũ và mới cùng nhau**

}

Truy vấn cập nhật trường actors để thêm Robert Duvall. Lưu ý rằng chúng ta cung cấp các actors hiện có trong mảng cùng với actor mới của chúng ta. Nếu chúng ta chỉ đưa Robert Duvall vào mảng actors, Elasticsearch sẽ thay thế danh sách chỉ bằng tên của anh ấy.

***5.4 Cập nhậttài liệu* 173**

Chúng ta đã thấy trong vài phần trước cách sửa đổi một tài liệu hiện có. Có những trường hợp chúng ta cần sửa đổi tài liệu dựa trên các điều kiện. Chúng ta thực hiện việc này bằng cách sử dụng các tập lệnh, như đã thảo luận trong phần tiếp theo.

* + 1. Cập nhật theo kịch bản

Chúng tôi đã sử dụng API cập nhật để sửa đổi tài liệu bằng cách thêm trường hoặc cập nhật các trường hiện có. Ngoài việc thực hiện việc này trên cơ sở từng trường, chúng tôi có thể thực thi các bản cập nhật dễ thương bằng cách sử dụng các tập lệnh. Các bản cập nhật theo tập lệnh cho phép chúng tôi cập nhật tài liệu dựa trên các điều kiện: ví dụ, xếp hạng một bộ phim là bom tấn nếu nó vượt qua ngưỡng doanh thu phòng vé nhất định.

Các tập lệnh được cung cấp trong một thân yêu cầu sử dụng cùng một điểm cuối \_update, với các bản cập nhật được gói trong một đối tượng tập lệnh bao gồm nguồn là khóa. Chúng tôi cung cấp các bản cập nhật dưới dạng giá trị cho khóa nguồn này với sự trợ giúp của biến ngữ cảnh ctx, lấy các thuộc tính của tài liệu gốc bằng cách gọi ctx.\_source.<field>.

**TRONGPDATING MẢNG BẰNG CÁCH SỬ DỤNG MỘT Script**

Hãy cập nhật tài liệu phim của chúng ta bằng cách thêm một diễn viên vào mảng hiện có. Lần này, chúng ta không sử dụng phương pháp được hiển thị trong danh sách 5.22, trong đó chúng ta đã đính kèm tất cả các diễn viên hiện códiễn viên vào trường diễn viên cùng với diễn viên mới. Thay vào đó, chúng ta chỉ cần sử dụng một tập lệnh để cập nhật trường diễn viên bằng một diễn viên khác.

**Liệt kê 5.23 Thêm một diễn viên vàodiễn viêndanh sách thông qua một tập lệnh**

POST phim/\_cập nhật/1

{

"kịch bản": {

"nguồn": "ctx.\_source.actors.add('Diane Keaton')"

**Diễn viên bổ sung**

}

}

ctx.\_source.actors lấy mảng actors và gọi phương thức add trên mảng đó để chèn giá trị mới (Diane Keaton) vào danh sách. Tương tự, chúng ta có thể xóa giá trị khỏi danh sách bằng cách sử dụng bản cập nhật theo tập lệnh, mặc dù việc này hơi phức tạp.

**RXÓA MỘT PHẦN TỬ KHỎI MẢNG**

Việc xóa một phần tử khỏi một mảng bằng cách sử dụng một tập lệnh yêu cầu chúng ta phải cung cấp chỉ mục của phần tử. Phương thức remove lấy một số nguyên trỏ đến chỉ mục của tác nhânchúng ta muốn xóa. Để lấy chỉ mục của một diễn viên, chúng ta có thể gọi phương thức indexOf trên đối tượng mảng thích hợp. Hãy xem cách thực hiện, xóa Diane Keaton khỏi danh sách.

**Liệt kê 5.24 Xóa một diễn viên khỏi danh sáchdiễn viên**

POST phim/\_cập nhật/1

{

"kịch bản":{

"nguồn":

**174 CPHẦN5*Làm việc với tài liệu***

"ctx.\_source.actors.remove(ctx.\_source.actors.indexOf('Diane Keaton'))"

}

} **Phương pháp loại bỏ mong đợi**

**vị trí nguyên của diễn viên.**

ctx.\_source.actors.indexOf('Diane Keaton')trả về một chỉ mục của phần tử trongdiễn viênmảng. Điều này là cần thiết chodi dờiphương pháp.

**MỘTDDING MỘT LĨNH VỰC MỚI**

Chúng ta cũng có thể thêm một trường mới vào tài liệu của mình bằng cách sử dụng một tập lệnh như trong danh sách sau. Ở đây chúng ta thêm một trường mới, imdb\_user\_rating, với giá trị là 9,2.

**Liệt kê 5.25 Thêm một trường mới có giá trị bằng cách sử dụngkịch bản**

POST phim/\_cập nhật/1

{

"kịch bản": {

"nguồn": "ctx.\_source.imdb\_user\_rating = 9.2"

}

}

**GHI CHÚ** Để thêm một giá trị mới vào một mảng (như chúng ta đã làm trong danh sách 5.23), chúng ta gọi

thêm vàophương pháp trên mảng:ctx.\_source.<mảng\_đối\_tượng>.add('giá\_trị')

**RDI CHUYỂN MỘT TRƯỜNG**

Xóa một trường cũng là một công việc đơn giản. Danh sách sau đây xóa một trường(imdb\_user\_rating) từ tài liệu phim của chúng tôi.

**Liệt kê 5.26 Xóa một trường khỏi tài liệu nguồn**

POST phim/\_cập nhật/1

{

"kịch bản": {

"nguồn": "ctx.\_source.remove('imdb\_user\_rating')"

}

}

**GHI CHÚ**Nếu chúng ta cố gắng xóa một trường không tồn tại, chúng ta sẽ không nhận được thông báo lỗi cho biết rằng chúng ta đang cố gắng xóa một trường không tồn tại trên lược đồ. Thay vào đó, chúng ta nhận được phản hồi cho biết tài liệu đã được cập nhật và trường đã được tăng lên (mà tôi coi là dương tính giả).

**MỘT****DDING NHIỀU TRƯỜNG**

Chúng ta có thể viết một tập lệnh để thêm nhiều trường cùng một lúc.

**Liệt kê 5.27 Thêm nhiều trường mới bằng cách sử dụng một tập lệnh**

POST phim/\_cập nhật/1

{

"script": { "source": """

***5.4 Cập nhậttài liệu* 175**

ctx.\_source.runtime\_in\_minutes = 175;

ctx.\_source.metacritic\_rating= 100;

ctx.\_source.tomatometer = 97;

ctx.\_source.boxoffice\_gross\_in\_millions = 134,8; """

}

}

Điều đáng chú ý trong danh sách này là các bản cập nhật đa dòng được thực hiện trong khối ba dấu ngoặc kép. Mỗi cặp khóa-giá trị được phân tách bằng dấu chấm phẩy (;).

**MỘTDDING MỘT KỊCH BẢN CẬP NHẬT CÓ ĐIỀU KIỆN**

Chúng ta cũng có thể triển khai logic phức tạp hơn một chút trong khối tập lệnh. Giả sử chúng ta muốn gắn thẻ một bộ phim là bom tấn nếu tổng doanh thu trên 125 triệu đô la. (Tôi đã tạo ra quy tắc này; trên thực tế, các yếu tố khác cũng có liên quan, chẳng hạn như ngân sách, ngôi sao, lợi tức đầu tư, v.v., để biến một bộ phim thành bom tấn.) Để thực hiện điều này, chúng ta hãy tạo một tập lệnh có điều kiện kiểm tra tổng doanh thu của bộ phim; nếu doanh thu vượt ngưỡng, chúng ta sẽ gắn nhãn bộ phim đó là bom tấn. Trong danh sách tiếp theo, chúng ta viếtcâu lệnh if/else đơn giản có logic dựa trên thu nhập và thiết lập cờ bom tấn tương ứng.

**Liệt kê 5.28 Cập nhật tài liệu có điều kiện bằng cách sử dụng khối if/else**

POST phim/\_cập nhật/1

{

"script": { "source": """

nếu(ctx.\_source.boxoffice\_gross\_in\_millions > 125)

{ctx.\_source.blockbuster = đúng} nếu không

{ctx.\_source.blockbuster = sai} """

}

}

Cácnếu nhưmệnh đề kiểm tra giá trị của trườngdoanh thu phòng vé tính bằng triệu đô. Sau đó nó tự động tạo ra một cái mớiphim bom tấntrường (chúng tôi chưa có trường đó trên lược đồ của mình) và đặt cờ thànhĐÚNG VẬYhoặcSAIdựa trên kết quả của điều kiện.

Cho đến nay, chúng ta đã làm việc với các ví dụ đơn giản bằng cách sử dụng các tập lệnh. Tuy nhiên, các tập lệnh cho phép chúng ta làm nhiều hơn thế nữa—từ một bản cập nhật đơn giản đến một sửa đổi có điều kiện phức tạp trên một tập dữ liệu. Hiểu được những điều cốt lõi của tập lệnh nằm ngoài phạm vi của cuốn sách này, nhưng nên tìm hiểu một vài khái niệm, vì vậy hãy thảo luận ngắn gọn về giải phẫu của một tập lệnh.

**MỘTNATOMY CỦA MỘT KỊCH BẢN**

Chúng ta hãy dừng lại để xem xét sơ qua cấu trúc của một tập lệnh. Một tập lệnh có ba phần: nguồn, ngôn ngữ và tham số, như thể hiện trong hình 5.13.

**176 CPHẦN5*Làm việc với tài liệu***

**Cáckịch bảnđối tượng là đối tượng cấp cao nhất bao bọcnguồn,chỉ,Vàtham sốvật thể bên trong.**

**Cácnguồnbao gồm các điều kiện/biểu thức và cài đặt giá trị cho tài liệu.**

"kịch bản" : {

"nguồn": "...",

"lang": "không đau|biểu thức..", "params": {..}

}

**Cáctham sốtrường này giúp truyền dữ liệu vào tập lệnh một cách động.**

**Cácchỉthuộc tính thiết lập ngôn ngữ biểu thức thích hợp: mặc định làkhông đau(chúng ta có thể bỏ quachỉtrường cho các tập lệnh không gây đau đớn).**

**Hình 5.13 Giải phẫu của một kịch bản**

Trường nguồn là nơichúng tôi cung cấp logic, trong khi trường params chứa các tham số mà tập lệnh mong đợi, được phân tách bằng ký tự thanh dọc (hoặc đường ống). Chúng tôi cũng có thể cung cấp ngôn ngữ mà tập lệnh của chúng tôi được viết: ví dụ, một trong các ngôn ngữguages ​​painless, expression, ria mép hoặc java, trong đó painless là mặc định. Tiếp theo, chúng ta hãy xem cách cập nhật tài liệu bằng cách truyền giá trị qua thuộc tính params.

**P****ĐÁNH GIÁ DỮ LIỆU CHO MỘT KỊCH BẢN**

Một vấn đề với mã trong danh sách 5.27 là chúng ta đã mã hóa cứng ngưỡng thu nhập trong tập lệnh (tổng thu nhập là 150 triệu đô la). Thay vào đó, chúng ta có thể đặt giá trị ngưỡng trong tập lệnh bằng cách sử dụng thuộc tính params. Hãy xem lại tập lệnh bom tấn của chúng ta, nhưng lần này chúng ta truyền giá trị của ngưỡng thu nhập gộp vào logic của tập lệnh thông qua params.

**Liệt kê 5.29 Truyền động một tham số cho tập lệnh**

POST phim/\_cập nhật/1

{

"script": { "source": """

**Việc kinh doanhlogic nằm ở đây.**

**Kiểm tra giá trị**



>

nếu(ctx.\_source.boxoffice\_gross\_in\_millions

params.gross\_earnings\_threshold)

{ctx.\_source.blockbuster = đúng} nếu không

{ctx.\_source.blockbuster = sai} """,

**chống lại các tham số**

"params": { "ngưỡng thu nhập gộp":150

}

}

}

**Cung cấp các giá trị tham số**

***5.4 Cập nhậttài liệu* 177**

Bản mã có hai thay đổi đáng chú ý so với phiên bản trước trong danh sách 5.28:

* Cácnếu nhưmệnh đề bây giờ được so sánh với một giá trị đọc từtham sốsự vật (params.gross\_earnings\_threshold).
* Cácngưỡng\_thu\_nhuận\_tổng\_tổngđược thiết lập để150thông quatham sốkhối.

Khi kịch bản làđược thực hiện, Elasticsearch tham khảo đối tượng params và thay thếthuộc tính có giá trị từ đối tượng params. Nếu chúng ta muốn thay đổi giá trị của tổng thu nhập để thiết lập cờ bom tấn (có lẽ ngưỡng params.gross\_earnings\_ cần được cập nhật thành 500 triệu đô la), chúng ta có thể chỉ cần truyền giá trị mới vào cờ params.

Bạn có để ý giá trị params được mã hóa cứng trong tập lệnh không? Bạn có thể tự hỏi tại sao chúng ta lại mã hóa cứng giá trị gross\_earnings\_threshold trong tập lệnh trong đối tượng params. Vâng, chức năng lập trình còn nhiều hơn những gì chúng ta thấy ở đây. Các tập lệnh được biên dịch khi chúng được thực thi lần đầu tiên. Việc biên dịch tập lệnh sẽ tốn kém về hiệu suất, vì vậy nó được coi là một hoạt động tốn kém trong Elasticsearch. Tuy nhiên, các tập lệnh liên quan đến các tham số thay đổi động (sử dụng đối tượng params) tránh được chi phí biên dịch này vì tập lệnh chỉ được biên dịch lần đầu tiên và được cập nhật bằng giá trị của biến (params) khi được gọi trong thời gian còn lại. Đây là một lợi ích đáng kể, vì vậy thông lệ chung là cung cấp các biến động thông qua đối tượng params cho tập lệnh (tham khảo danh sách 5.29).

**SNGÔN NGỮ CRIPTING**

Các tập lệnh được phát triển trong chương này được lấy từ ngôn ngữ kịch bản đặc biệt của Elasticsearch, được gọi là Painless, để giải mã logic và thực thi các tập lệnh. Ngôn ngữ mặc định là Painless (chúng tôi đã không chỉ định rõ ngôn ngữ trong mã của mình trước đó). Chúng ta có thể cắm vào các ngôn ngữ kịch bản khác (Mustache, Expression hoặc thậm chí là Java) bằng cách sử dụngmột tham số ngôn ngữ. Bất kể chúng ta sử dụng ngôn ngữ nào, chúng ta phải tuân theo mẫu thiết lập này:

"kịch bản": {

"lang": "không đau|ria mép|biểu cảm|java", "nguồn": "...",

"tham số": { ... }

}

Cho đến nay, chúng tôi đã cập nhật từng tài liệu bằng lệnh gọi API \_update hoặc tập lệnh. Còn việc cập nhật một loạt tài liệu phù hợp với một tiêu chí thì sao? Đó chính xác là những gì chúng ta sẽ tìm hiểu trong phần tiếp theo.

* + 1. Thay thế tài liệu

Giả sử chúng ta cần thay thế một tài liệu hiện có bằng một tài liệu mới. Điều này rất dễ: chúng ta có thể sử dụng cùng một yêu cầu PUT mà chúng ta đã thực hiện trước đó trong chương khi lập chỉ mục một tài liệu mới. Hãy chèn một tiêu đề phim mới (Avatar) vào tài liệu phim của chúng ta nhưng liên kết nó với một tài liệu hiện có (ID = 1).

**178 CPHẦN5*Làm việc với tài liệu***

**Liệt kê 5.30 Thay thế nội dung của mộttài liệu**

PUT phim/\_doc/1

{

"title":"Hình đại diện"

}

Bộ phim hiện tại, Bố già, được thay thế bằng thuộc tính dữ liệu mới (Avatar) sau khi chúng ta thực hiện lệnh này.

**GHI CHÚ**Nếu mục đích của chúng tôi là thay thế nội dung hiện có bằng nội dung khác, chúng tôi sử dụngAPI \_doc trên cùng một ID với một thân yêu cầu mới. Tuy nhiên, nếu chúng ta không muốn thay thế tài liệu, chúng ta phải sử dụng điểm cuối \_create (được thảo luận trong phần 5.1.1).

Đôi khi, khi chúng ta cố gắng cập nhật một tài liệu không tồn tại, chúng ta muốn Elasticsearch lập chỉ mục nó như một tài liệu mới thay vì đưa ra lỗi. Đây chính là mục đích của hoạt động upsert.

* + 1. Upserts

*Buồn bã*, viết tắt của cập nhật và chèn, là một hoạt động cập nhật tài liệu (nếu có) hoặc lập chỉ mục tài liệu mới bằng dữ liệu được cung cấp (nếu không có). Hình 5.14 minh họa hoạt động này.

Hoạt động Upsert

Tài liệu

có tồn tại không?

Đúng

KHÔNG

Lập chỉ mục một tài liệu mới

Thực hiệnkịch bản

**Hình 5.14 Quy trình hoạt động Upsert**

Giả sử chúng ta muốn cập nhật gross\_earnings cho bộ phim Top Gun. Hãy nhớ rằng chúng ta chưa có bộ phim này trong cửa hàng. Chúng ta có thể phát triển truy vấn để cập nhật trường gross\_earnings và tạo một tài liệu mới với bản cập nhật này.

* 1. ***Đang cập nhậttài liệu* 179**

Truy vấn có hai phần: một tập lệnh và một khối upsert. Phần tập lệnh là nơi chúng ta cập nhật trường trên một tài liệu hiện có, trong khi khối upsert bao gồm thông tin tài liệu phim mới.

**Liệt kê 5.31 Khối Upsertví dụ**

POST phim/\_cập nhật/5

{

"kịch bản": {

"nguồn": "ctx.\_source.gross\_earnings = '357,1 triệu'"

},

"upsert": { "title":"Top Gun",

"gross\_earnings":"357,5 triệu"

}

}

Khi chúng tôi thực hiện truy vấn này, chúng tôi mong đợi tập lệnh chạy và cập nhật trường gross\_ earnings trên một tài liệu có ID 5 (nếu tài liệu khả dụng). Điều gì xảy ra nếu tài liệu đó không có trong chỉ mục của chúng tôi? Vâng, đó là nơi khối upsert phát huy tác dụng. Phần thứ hai của yêu cầu JSON rất thú vị: khối upsert bao gồm các trường tạo thành một tài liệu mới. Bởi vì không có tài liệu nào có ID đó tronglưu trữ, khối upsert tạo một tài liệu mới với các trường được chỉ định. Vì vậy, một hành động upsert cung cấp cho chúng ta khả năng cập nhật một tài liệu hiện có hoặc tạo

tạo một mục mới nếu đây là lần đầu tiên chúng ta lập chỉ mục.

Nếu chúng ta chạy cùng một truy vấn lần thứ hai, phần tập lệnh sẽ được thực thi và thay đổi trường gross\_earnings thành 357,1 triệu (từ 357,5 triệu trong tài liệu gốc). Điều này xảy ra vì tài liệu đã tồn tại.

* + 1. Cập nhật dưới dạng upserts

Trong phần 5.4.2, chúng ta đã xem xét việc cập nhật một phần tài liệu bằng đối tượng doc thông qua API \_update. Trong danh sách sau, chúng ta cập nhật một bộ phim bằng một trường bổ sung, runtime\_in\_minutes. Nếu tài liệu 11 tồn tại, trường này được cập nhật như mong đợi; nếu không, một lỗi sẽ được đưa ra, nêu rằng tài liệu không tồn tại.

**Liệt kê 5.32 Cập nhật một trường không tồn tại (sẽ gây ra lỗi)**

POST phim/\_cập nhật/11

{

"doc": { "runtime\_in\_minutes":110

}

}

Để tránh lỗi—có thể chúng ta muốn tạo một tài liệu mới nếu tài liệu khôngtồn tại—chúng ta có thể sử dụng cờ doc\_as\_upsert. Đặt cờ này thành true cho phép lưu trữ nội dung của đối tượng doc dưới dạng tài liệu mới nếu tài liệu (ID = 11) không tồn tại.

**180 CPHẦN5*Làm việc với tài liệu***

**Liệt kê 5.33 Cập nhật tài liệu nếu trường không tồn tại**

POST phim/\_cập nhật/11

{

"doc": { "runtime\_in\_minutes":110

},

"doc\_as\_upsert":đúng

}

Lần này, nếu chúng ta không có tài liệu có ID 11 trong cửa hàng của mình, thì cũng không thành vấn đề. Công cụ không đưa ra lỗi;thay vào đó, nó tạo ra một tài liệu mới với ID 11 vàcác trường được trích xuất từ ​​đối tượng doc (runtime\_in\_minutes, trong trường hợp này).

* + 1. Cập nhật bằng truy vấn

Đôi khi chúng ta muốn cập nhật một tập hợp các tài liệu khớp với tiêu chí tìm kiếm—ví dụ, cập nhật tất cả các phim có xếp hạng lớn hơn 4 sao để chỉ ra rằng chúng phổ biến. Chúng ta có thể chạy truy vấn để tìm kiếm một tập hợp các tài liệu như vậy và áp dụng các bản cập nhật cho chúng bằng cách sử dụng điểm cuối \_update\_by\_query. Ví dụ, hãy cập nhật tất cả các phim có tên diễn viên khớp với Al Pacino thành Người chiến thắng giải Oscar Al Pacino.

**Liệt kê 5.34 Cập nhật tài liệu bằng cách sử dụngtruy vấnphương pháp**

POST phim/\_cập nhật\_theo\_truy\_cập

{

"truy vấn": {

"cuộc thi đấu": {

"diễn viên": "Al Pacino"

**Tìm kiếm tài liệu và cập nhật bộ**

**Truy vấn tìm kiếm: tìm kiếm tất cả các bộ phim có Al Pacino**

}

}, **Áp dụng logic tập lệnh sau**

"script": { "source": """

**đến các tài liệu phù hợp**

ctx.\_source.actors.add('Người chiến thắng giải Oscar Al Pacino'); ctx.\_source.actors.remove(ctx.\_source.actors.indexOf('Al Pacino')) """,

"lang": "không đau"

}

}

Trong danh sách này, truy vấn khớp lệnh được thực hiện đầu tiên(chúng ta thảo luận về các truy vấn khớp trong chương 10, nhưng hãy coi đó là một loại truy vấn mà chúng ta có thể lấy các tài liệu khớp với một tiêu chí nhất định) để lấy tất cả các bộ phim có diễn viên Al Pacino. Khi truy vấn khớp trả vềkết quả, một tập lệnh được thực thi để đổi Al Pacino thành Người chiến thắng giải Oscar Al Pacino. Vì chúng ta cần xóa tên cũ, chúng ta cũng gọi một thao tác xóa trong tập lệnh.

\_update\_by\_query là một cơ chế tiện dụng để cập nhật nhiều tài liệu dựa trên một tiêu chí. Tuy nhiên, có rất nhiều thứ diễn ra đằng sau hậu trường khi chúng ta sử dụng phương pháp này; hãy xem thanh bên sau để tìm hiểu thêm.

* 1. ***Xóa bỏtài liệu* 181**

**Cơ chế của update\_by\_query**

Elasticsearch đầu tiên phân tích cú pháp truy vấn đầu vào và xác định các phân đoạn chứa các tài liệu có khả năng khớp với truy vấn của chúng tôi. Đối với mỗi phân đoạn này, Elasticsearch thực hiện truy vấn và tìm tất cả các tài liệu khớp với nó.

Elasticsearch sẽ chạy tập lệnh mà chúng tôi đã cung cấp trên mỗi tài liệu khớp này. Trước khi cập nhật các tài liệu, Elasticsearch kiểm tra xem phiên bản của tài liệu hiện tại có tương ứng với phiên bản khi tìm thấy trong giai đoạn truy vấn hay không. Nếu chúng khớp, quá trình cập nhật sẽ tiếp tục; nếu không (có thể tài liệu đã được cập nhật do một hoạt động tạm thời khác), Elasticsearch sẽ thử lại hoạt động. Sau đó, mỗi tài liệu được cập nhật trong bộ nhớ và lập chỉ mục lại, đánh dấu tài liệu cũ là đã xóa trong khi một tài liệu mới được thêm vào chỉ mục.

Khi cập nhật một tài liệu cụ thể không thành công, lỗi sẽ được ghi lại nhưng các tài liệu còn lại sẽ được cập nhật theo tập lệnh. Bất kỳ xung đột nào trong quá trình cập nhật đều có thể được thử lại một số lần cụ thể hoặc bỏ qua hoàn toàn theo các thiết lập được cấu hình trước. Sau khi hoàn tất thao tác và các chỉ mục được làm mới, các thay đổi sẽ có sẵn để tìm kiếm.

Các\_cập\_nhật\_theo\_truy\_cậphoạt động trả về một phản hồi bao gồm số lượng tổng sốtài liệu đã xử lý, xung đột phiên bản, tài liệu đã cập nhật thành công và tài liệu không thành công. Như bạn có thể tưởng tượng,\_cập\_nhật\_theo\_truy\_cậphoạt động là một nguồn lực chuyên sâuhoạt động, do đó cần cân nhắc cẩn thận về tác động của nó đến hiệu suất cụm của bạn. Chúng ta có thể để Elasticsearch xử lý các bản cập nhật theo từng đợt nếu nó sử dụng quá nhiều tài nguyên. Có thể cấu hình kích thước lô và giới hạn điều tiết.

Bây giờ chúng ta đã biết cơ chế cập nhật tài liệu. Đã đến lúc thảo luận về các thao tác xóa—cụ thể là cách xóa một tài liệu hoặc nhiều tài liệu cùng lúc. Phần tiếp theo dành riêng cho việc xóa tài liệu theo nhiều cách.

#### Xóa tài liệu

Khi chúng ta muốn xóa tài liệu, có hai phương pháp: sử dụng ID hoặc truy vấn. Trong trường hợp trước, chúng ta có thể xóa một tài liệu duy nhất; và trong trường hợp sau, chúng ta có thể xóa nhiều tài liệu cùng một lúc. Khi xóa theo truy vấn, chúng ta có thể đặt tiêu chí lọc (ví dụ, xóa tài liệu có trường trạng thái chưa được xuất bản hoặc tài liệu từ tháng trước). Chúng ta hãy xem cả hai phương pháp này hoạt động như thế nào.

* + 1. Xóa bằng ID

Chúng ta có thể xóa một tài liệu duy nhất khỏi Elasticsearch bằng cách gọi lệnh HTTP DELETE

phương pháp trên API lập chỉ mục tài liệu:

XÓA <index\_name>/\_doc/<id>

URL giống với URL chúng tôi sử dụng để lập chỉ mục và truy xuất tài liệu. Với ID của tài liệu cần xóa, chúng tôi xây dựng URL bằng cách chỉ định

**182 CPHẦN5*Làm việc với tài liệu***

index, điểm cuối \_doc và ID tài liệu. Ví dụ, chúng ta có thể gọi truy vấn sau để xóa tài liệu có ID 1 khỏi chỉ mục movies.

**Liệt kê 5.35 Xóa mộtbộ phimtài liệu từ chỉ mục**

XÓA phim/\_doc/1

Phản hồi nhận được từ máy chủ cho biết tài liệu đã được xóa thành công:

{

...

"\_id" : "1",

"\_version" : 2, "result" : "đã xóa"

}

Phản hồi trả về một thuộc tính kết quả được đặt thành đã xóa để cho máy khách biết đối tượng đã được xóa thành công. Nếu tài liệu không bị xóa (ví dụ: tài liệu không tồn tại trong kho lưu trữ của chúng tôi), chúng tôi nhận được phản hồi với giá trị kết quả được đặt thành not\_found (mã trạng thái 404). Điều thú vị là Elasticsearch tăng cờ \_version nếu thao tác xóa thành công.

* + 1. [Xóa theo truy vấn (\_delete\_by\_query)](http://ndjson.org/)

Xóa một tài liệu đơn lẻ thì dễ, như chúng ta vừa thấy. Nhưng nếu chúng ta muốn xóa nhiềutài liệu dựa trên một tiêu chí, chúng ta có thể sử dụng \_delete\_by\_query (tương tự như \_update\_ by\_query trong phần 5.4.7). Ví dụ, nếu chúng ta muốn xóa tất cả các bộ phim do James Cameron đạo diễn, chúng ta có thể viết truy vấn này.

**Liệt kê 5.36 Xóa tất cả phim dựa trên tiêu chí**

POST phim/\_xóa\_bằng\_truy\_cập

{

"truy vấn":{

"cuộc thi đấu":{

"đạo diễn":"James Cameron"

}

}

}

Ở đây chúng tôi tạo truy vấn trong phần thân yêu cầu bằng cách sử dụng tiêu chí là tất cả các bộ phim do James Cameron đạo diễn. Các tài liệu khớp với tiêu chí sau đó được đánh dấu và xóa.

Nội dung của POST này sử dụng cú pháp đặc biệt được gọi là Query DSL (miền cụ thể lan-guage), cho phép chúng ta truyền vào nhiều thuộc tính như term, range và match (như trong danh sách này), tương tự như các truy vấn tìm kiếm cơ bản. Chúng ta sẽ tìm hiểu thêm về các truy vấn tìm kiếm trong các chương sau, nhưng hiện tại, hãy lưu ý rằng \_delete\_by\_query là một điểm cuối mạnh mẽ với các tiêu chí xóa phức tạp. Hãy cùng xem một vài ví dụ trong các phần sau.

***5.5 Xóatài liệu* 183**

* + 1. Xóa bằng truy vấn phạm vi

Chúng tôi có thể muốn xóa các bản ghi nằm trong một phạm vi nhất định: phim có xếp hạng từ

3.5 và 4.5, các chuyến bay bị hủy giữa hai ngày, v.v. Chúng ta có thể sử dụng truy vấn phạm vi để đặt tiêu chí cho phạm vi giá trị cho các yêu cầu như vậy. Danh sách sau sử dụng \_delete\_by\_query để xóa các bộ phim có doanh thu từ 350 triệu đến 400 triệu đô la.

**Liệt kê 5.37 Xóa tất cả các phim có phạm vi tổng doanh thuthu nhập**

POST phim/\_xóa\_bằng\_truy\_cập

{

"truy vấn": {

"phạm vi": { "tổng thu nhập tính bằng triệu": {

"gt": 350,

"lt": 400

}

}

}

}

Tại đây, \_delete\_by\_query chấp nhận truy vấn phạm vi với tiêu chí khớp: tìm tài liệu có tổng thu nhập nằm trong khoảng từ 350 đến 400 triệu đô la. Như chúng tôi mong đợi, tất cả các tài liệu khớp đều bị xóa.

Chúng ta cũng có thể xây dựng các truy vấn phức tạp. Ví dụ, danh sách 5.38 hiển thị một truy vấn xây dựng các tiêu chí để xóa các bộ phim do Steven Spielberg đạo diễn, trong đó các bộ phim đó được xếp hạng từ 9 đến 9,5 và kiếm được dưới 100 triệu đô la. Danh sách sử dụng truy vấn bool làm yêu cầu.

**GHI CHÚ**Đối với truy vấn sau, bạn cần lập chỉ mục tập dữ liệu phim từ các tệp của sách:<http://mng.bz/zXeX>.

**Liệt kê 5.38 Xóa phim có tiêu chí truy vấn phức tạp**

POST phim/\_xóa\_bằng\_truy\_cập

{

"truy vấn": {

"bool": {

"phải": [{

"cuộc thi đấu": {

**Phù hợp với những bộ phim do Spielberg đạo diễn**

"đạo diễn": "Steven Spielberg"

}

}

], **Xếp hạng phải cao hơn**

"must\_not": [{

"phạm vi": { "imdb\_rating": {

"gte": 9,

"lte": 9.5

**lớn hơn 9 nhưng nhỏ hơn 9,5.**

**184 CPHẦN5*Làm việc với tài liệu***

}

}

}

], **Thu nhập không nên giảm**

"lọc": [{

"phạm vi": {

**dưới 100 triệu đô la**

"tổng thu nhập tính bằng triệu": { "lt": 100

}

}

}

]

}

}

}

Truy vấn sử dụng logic truy vấn phức tạp được gọi là truy vấn bool kết hợp nhiều truy vấn nhỏ hơn để hoạt động trên quy mô lớn. Chương 12 dành riêng cho các truy vấn bool và đào sâu hơn vào cách chúng ta có thể xây dựng các truy vấn phức tạp.

* + 1. Xóa tất cả tài liệu

**CẢNH BÁO**Các thao tác xóa không thể đảo ngược! Hãy thận trọng trước khi truy vấn xóa Elasticsearch.

Bạn có thể xóa toàn bộ tập hợp tài liệu khỏi chỉ mục bằng cách sử dụng truy vấn match\_all.

**Liệt kê 5.39 Xóa tất cả các tài liệu tạimột lần**

POST phim/\_xóa\_bằng\_truy\_cập

{

"truy vấn": {

"match\_all": {}

}

}

Mã nàychạy truy vấn match\_all. Nó khớp với tất cả các tài liệu và xóa chúng đồng thời. Đây là một hoạt động phá hủy, vì vậy hãy cẩn thận khi xóa toàn bộ một tập hợptài liệu! Chúng ta cũng có thể xóa toàn bộ chỉ mục bằng cách sử dụng lệnh DELETE movies—nhưng hãy nhớ rằng đây là những lệnh không thể đảo ngược.

Cho đến nay, chúng tôi đã xóa các tài liệu trên một chỉ mục duy nhất. Chúng tôi cũng có thể xóa các tài liệu trên nhiều chỉ mục bằng cách chỉ cần cung cấp danh sách các chỉ mục được phân tách bằng dấu phẩy trong URL API. Định dạng ví dụ được hiển thị ở đây:

POST <index\_1>,<index\_2>,<index\_3>/\_xóa\_bằng\_truy\_cấp

Danh sách sau đây xóa tất cả các tài liệu trên nhiều chỉ mục liên quan đến phim. Lưu ý rằng chúng ta có thể sử dụng GET \_cat/indices để liệt kê tất cả các chỉ mục.

***5.6 Làm việc với các tài liệu trongsố lượng lớn* 185**

**Liệt kê 5.40 Xóa tài liệu khỏi ba chỉ mục phim khác nhau**

POST old\_movies,classics,movie\_reviews/\_delete\_by\_query

{

"truy vấn": {

"match\_all": {}

}

}

Một lần nữa, hãy cẩn thận khi phát hành truy vấn xóa, vì bạn có thể mất toàn bộ tập dữ liệu! Trừ khi bạn muốn xóa toàn bộ tập dữ liệu, hãy thực hiện các thao tác xóa một cách thận trọng trong quá trình sản xuất.

Cho đến nay, chúng tôi đã lập chỉ mục các tài liệu riêng lẻ, nhưng trong thế giới thực, luôn có trường hợp lập chỉ mục các tập tài liệu lớn cùng lúc. Chúng tôi có thể có 100.000 bộ phim để đọc từ tệp CSV hoặc 500.000 tỷ giá hối đoái ngoại tệ được lấy từ dịch vụ của bên thứ ba để lập chỉ mục vào công cụ của chúng tôi. Mặc dù chúng tôi thường có thể sử dụng công cụ ETL (trích xuất-biến đổi-tải) như Logstash để trích xuất, làm giàu và xuất bản dữ liệu; Elasticsearchcung cấp API số lượng lớn (\_bulk) để lập chỉ mục tin nhắn theo số lượng lớn. Chúng tôi thảo luận về API số lượng lớn trong phần tiếp theo.

#### Làm việc với các tài liệu hàng loạt

Cho đến nay, chúng tôi đã lập chỉ mục các tài liệu riêng lẻ bằng Kibana. Việc lập chỉ mục một tài liệu hoặc một số tài liệu bằng các phương pháp API rất đơn giản. Điều này rất tốt cho mục đích phát triển, nhưng chúng tôi hiếm khi thực hiện điều này trong quá trình sản xuất. Nó khá cồng kềnh và dễ xảy ra lỗi đối với một tập dữ liệu lớn hơn (ví dụ: khi trích xuất một số lượng lớn bản ghi từ cơ sở dữ liệu).

May mắn thay, Elasticsearch cung cấp API \_bulk để lập chỉ mục số lượng lớn dữ liệu để lập chỉ mục các tập dữ liệu lớn cùng một lúc. Chúng ta cũng có thể sử dụng API \_bulk để thao tác tài liệu, bao gồm xóa chúng.

API \_bulk chấp nhận yêu cầu POST, có thể thực hiện các hành động lập chỉ mục, tạo, xóa và cập nhật đồng thời. Điều này giúp tiết kiệm băng thông bằng cách tránh nhiều lần truy cập đến máy chủ. Có một định dạng đặc biệt cho API \_bulk, bạn có thể thấy hơi lạ nhưng không khó để hiểu. Trước tiên, hãy xem định dạng.

* + 1. Định dạng của API \_bulk

API \_bulk bao gồm một cú pháp cụ thể với phương thức POST gọi lệnh gọi API (xem hình 5.15). Nội dung yêu cầu bao gồm hai dòng cho mọi tài liệu cần được lưu trữ. Dòng đầu tiên chỉ ra một trong các hành động cần thực hiện trên tài liệu: lập chỉ mục, tạo, xóa hoặc cập nhật. Tài liệu được mô tả ở dòng thứ hai, chúng tôi sẽ đề cập ngắn gọn.

Sau khi chọn hành động, chúng ta cần cung cấp giá trị cho khóa hành động này với siêu dữ liệu. Siêu dữ liệu thường là tên của chỉ mục tài liệu và ID tài liệu. Ví dụ, siêu dữ liệu cho tài liệu có ID 100 trong chỉ mục phim là "\_index":"movies","\_id":"100".

**186 CPHẦN5*Làm việc với tài liệu***

**Chúng ta có thể lập chỉ mục, cập nhật hoặc xóa tài liệu bằng một trong những hành động này.**

**Dòng đầu tiên: hành động và siêu dữ liệu**

**Tên chỉ mục và ID tài liệu**

POST \_số lượng lớn

{"index|tạo|xóa|cập nhật":{"\_index":"<index\_name>","\_id":<identifier>

{"field1": "giá trị","field2": "giá trị","field3": "giá trị"}

**Dòng thứ hai:**

**dữ liệu nguồn**

**Các nguồn tài liệu được cung cấp ở dòng mới dưới dạng cặp trường-giá trị.**

**Hình 5.15 Các\_số lượng lớnĐịnh dạng chung của API**

Dòng thứ hai trong hình 5.15 là nguồn của tài liệu, là thứ chúng ta muốn lưu trữ tài liệu. Như mong đợi, tài liệu được định dạng trong JSON và được thêm vào yêu cầu trong dòng mới. Cả siêu dữ liệu và dòng nguồn đều được phân định bằng dấu phân cách dòng mới (\n) được thể hiện trong JSON (tức là JSON phân định bằng dòng mới [NDJSON,[http://ndjson.org](http://ndjson.org/)]—một định dạng thuận tiện để lưu trữ các bản ghi được sử dụng từng cái một).

**GHI CHÚ**Yêu cầu đính kèm vào API \_bulk phải tuân thủ nghiêm ngặt định dạng NDJSON; nếu không, các tài liệu sẽ không được lập chỉ mục hàng loạt. Mỗi dòng phải kết thúc bằng dấu phân cách dòng mới vì yêu cầu hàng loạt nhạy cảm với dòng mới. Đảm bảo tài liệu của bạn được định dạng là NDJSON.

Với khái niệm này trong đầu, chúng ta hãy tạo một yêu cầu hàng loạt để lập chỉ mục tài liệu phim

*Nhiệm vụ bất khả thi*.

* + 1. Tài liệu lập chỉ mục hàng loạt

Chúng tôi muốn lập chỉ mục tài liệu bằng API \_bulk, vì vậy hãy xem ví dụ sau. Yêu cầu tương tự được hiển thị trong hình 5.16 với chú thích.

**Liệt kê 5.41 Lập chỉ mục hàng loạt tronghoạt động**

POST \_số lượng lớn

**\_Số lượng lớn**

**Giao diện lập trình ứng dụng (API)Địa chỉ URL Chúng tôi muốn "lập chỉ mục"**

{"index":{"\_index":"phim","\_id":"100"}}

**tài liệu này.**

{"title": "Nhiệm vụ bất khả thi","release\_date": "1996-07-05"}

**Tài liệu của chúng tôi**

***5.6 Làm việc với các tài liệu trongsố lượng lớn* 187**

POST \_số lượng lớn

**Bộ phim được lập chỉ mục vàophimmục lục.**

**Bộ phimID là100.**

{"**chỉ số**":{"\_index":"phim","\_id":"100"}}

{"title": "Nhiệm vụ bất khả thi","release\_date": "1996-07-05"}

**Cácchỉ sốhành động (được tô sáng) cho phép công cụ lập chỉ mục tài liệu mới.**

**Cáctiêu đềVàngày phát hànhCác trường của phim được cung cấp ở dòng thứ hai của yêu cầu.**

**Hình 5.16 Lập chỉ mục cho bộ phim mới Nhiệm vụ bất khả thi bằng cách sử dụng\_số lượng lớnGiao diện lập trình ứng dụng (API)**

Nếu chúng ta thực hiện truy vấn này, chúng ta có một tài liệu có ID 100 được lập chỉ mục vào tài liệu phim với các trường được cung cấp ở dòng thứ hai. Nghĩa là, bộ phim Mission Impossible được lập chỉ mục vào kho lưu trữ của chúng ta. Hai dòng tạo ra một yêu cầu—chúng ta phải mã hóa hai dòng cho mỗi tài liệu mà chúng ta muốn được xử lý.

Chúng ta cũng có thể rút ngắn dòng siêu dữ liệu. Ví dụ, chúng ta có thể xóa chỉ mục và đính kèm nó vào URL.

**Danh sách 5.42 API hàng loạt với chỉ mục được nhúng trong URL yêu cầu**

BƯU KIỆNphim/\_số lượng lớn 

{"chỉ mục":{"\_id":"100"}}

**URL bao gồm**

**chỉ số. Trường \_index đã bị xóa.**

{"title": "Nhiệm vụ bất khả thi","release\_date": "1996-07-05"}

Chúng ta cũng có thể loại bỏ trường \_id nếu muốn hệ thống tạo ID ngẫu nhiên cho phim của chúng ta. Đoạn mã sau đây minh họa cách tiếp cận này.

**Liệt kê 5.43 Cho phép hệ thống tạo tài liệuID**

POST phim/\_số lượng lớn

{"chỉ mục":{}}

**Cả \_index và**

**\_id đã bị xóa**

{"title": "Nhiệm vụ bất khả thi","release\_date": "1996-07-05"}

Hệ thống sẽ gán ID cho tài liệu này, một UUID ngẫu nhiên.

Tất nhiên, bạn có thể tự hỏi tại sao chúng ta cần phải theo cách tiếp cận số lượng lớn này khi chúng ta có thể lập chỉ mục cùng một tài liệu bằng API chỉ mục tài liệu (PUT movies/\_doc/100). Đó là một câu hỏi hợp lý, nhưng chúng ta vẫn chưa giải phóng toàn bộ sức mạnh của API \_bulk. Giả sử chúng ta có yêu cầu lập chỉ mục các bộ phim của Tom Cruise vào chỉ mục phim của mình. Yêu cầu được viết trong danh sách tiếp theo.

**188 CPHẦN5*Làm việc với tài liệu***

**Danh sách 5.44 Lập chỉ mục hàng loạt Tom Cruisephim**

POST phim/\_số lượng lớn

{"chỉ mục":{}}

{"title": "Nhiệm vụ bất khả thi","release\_date": "1996-07-05"}

{"chỉ mục":{}}

{"title": "Nhiệm vụ bất khả thi II","release\_date": "2000-05-24"}

{"chỉ mục":{}}

{"title": "Nhiệm vụ bất khả thi III","release\_date": "2006-05-03"}

{"chỉ mục":{}}

{"title": "Nhiệm vụ bất khả thi - Giao thức bóng ma","release\_date": "2011-12-26"}

Vì chúng tôi đã đính kèm tên chỉ mục vào URL (POST movies/\_bulk) và không quan tâm đến ID được xác định trước nên truy vấn đã lập chỉ mục thành công bốn bộ phim của Tom Cruise.

* + 1. Các thực thể độc lập và nhiều hành động

Chúng tôi chỉ lập chỉ mục cho các phim Nhiệm vụ bất khả thi, nhưng chúng tôi có thể lập chỉ mục cho các thực thể khác trong cùng một yêu cầu. Điều đáng chú ý là API \_bulk giúp gói gọn nhiều thực thể—chúng tôi có thể đóng gói không chỉ các bộ phim mà còn bất kỳ loại nào, chẳng hạn như sách, chuyến bay, nhật ký, v.v., trong một yêu cầu. Danh sách sau đây có danh sách các yêu cầu loại trừ lẫn nhau.

**Danh sách 5.45 Yêu cầu hàng loạt với túi hỗn hợpyêu cầu**

POST \_số lượng lớn

{"index":{"\_index":"sách"}}

**Lập chỉ mục một cuốn sách**

{"title": "Elasticsearch trong hành động"}

{"create":{"\_index":"chuyến bay", "\_id":"101"}}

{"title": "Từ London đến Bucharest"}

{"index":{"\_index":"thú cưng"}}

{"tên": "Milly","tuổi\_tháng": 18}

{"xóa":{"\_index":"phim", "\_id":"101"}}

{ "cập nhật" : {"\_index":"phim", "\_id":"1"} }

 **Tạo một chuyến bay Lập chỉ mục một con vật cưng vào**

**một chỉ số vật nuôi**

**Xóa một bộ phim**

**Cập nhật tiêu đề**



}

{ "doc" : {"title" : "Bố già (Bản gốc)"}

**của một bộ phim**

Yêu cầu hàng loạt này bao gồm hầu hết các hành động có thể thực hiện khi sử dụng API \_bulk. Hãy cùng tìm hiểu sâu hơn về các hành động riêng lẻ này.

**TANH TA TẠO RA HÀNH ĐỘNG**

Chúng ta có thể hoán đổi hành động index với create để không thay thế document nếu nó không tồn tại trong khi lập chỉ mục (xem phần 5.1.3). Mã sau đây cho thấy hoạt động của create.

**Danh sách5,46 \_số lượng lớnAPI vớitạo nênhoạt động**

POST \_số lượng lớn

{"create":{"\_index":"phim","\_id":"101"}}

{"title": "Nhiệm vụ bất khả thi II","release\_date": "2000-05-24"}

**Tránh ghi đè ngẫu nhiên**

* 1. ***Làm việc với các tài liệu trongsố lượng lớn* 189**

**THÀNH ĐỘNG CẬP NHẬT**

Việc cập nhật tài liệu tuân theo một mô hình tương tự, nhưng chúng ta phải gói các trường cần cập nhật trong một đối tượng doc, như đã học ở phần 5.4.2. Ví dụ này cập nhật phim có ID 200 (Giờ cao điểm) với các trường bổ sung (đạo diễn và diễn viên).

**Liệt kê 5.47 Cập nhật hàng loạt di chuyển Giờ cao điểm**

POST \_số lượng lớn

{"cập nhật":{"\_index":"phim","\_id":"200"}}

{"doc": {"đạo diễn":"Brett Ratner", "diễn viên":["Jackie Chan","Chris Tucker"]}}

**TAnh ấy xóa hành động**

Cuối cùng, hãy sử dụng API \_bulk để xóa một tài liệu. Định dạng hơi khác một chút, như được hiển thị bên dưới.

**Liệt kê 5.48 Cuộc gọi hàng loạt vớixóa bỏhoạt động**

POST \_số lượng lớn

{"xóa":{"\_index":"đánh giá phim","\_id":"111"}}

Như bạn có thể thấy, chúng ta không cần dòng thứ hai cho hoạt động này. Truy vấn xóaĐánh giá phim có ID 111 từ chỉ mục movie\_reviews. Hãy nhớ rằng, bạn có thể mất toàn bộ tập dữ liệu nếu không cẩn thận khi đưa ra truy vấn xóa.

* + 1. Yêu cầu hàng loạt sử dụng cURL

Chúng tôi đã sử dụng Kibana để thực hiện các thao tác trên tài liệu bằng API \_bulk. Chúng tôi cũng có thể thực hiện các hành động này bằng cURL. Trên thực tế, sử dụng cURL có thể là phương pháp được ưu tiên nếu chúng tôi có khối lượng bản ghi lớn hơn để xử lý.

Để sử dụng cURL, chúng ta cần tạo một tệp JSON chứa tất cả dữ liệu và chuyển tệp đó đến cURL với cờ --data-binary. Dữ liệu được sử dụng trong danh sách 5.49 có sẵn trong tệp movie\_bulk\_data.json có sẵn với các tệp của sách; bạn có thể chuyển tệp đó đến cURL.

**Liệt kê 5.49 Sử dụng cURL để thực thi dữ liệu hàng loạthoạt động**

curl -H "Loại nội dung: application/x-ndjson"

-XPOST localhost:9200/\_bulk

--data-binary "@movie\_bulk\_data.json"

Máy chủ cục bộ là địa chỉ của phiên bản Elasticsearch đang chạy trên máy cục bộ của chúng tôi.

**GHI CHÚ**Hãy đảm bảo đặt cờ --data-binary tên tệp của bạn (không có phần mở rộng) với tiền tố @.

Bây giờ chúng ta biết cách làm việc với các yêu cầu hàng loạt. Tuy nhiên, đôi khi chúng ta muốn di chuyển các tài liệu từ chỉ mục này sang chỉ mục khác (ví dụ: từ phim bom tấn sang

**190 CPHẦN5*Làm việc với tài liệu***

classic\_movies). API \_bulk có thể không phù hợp để di chuyển (hoặc di chuyển) dữ liệu giữa các chỉ mục. Elasticsearch cung cấp một tính năng phổ biến: API lập chỉ mục lại (\_reindex). Điều này được thảo luận chi tiết trong phần tiếp theo.

#### Lập chỉ mục lại tài liệu

Tùy thuộc vào ứng dụng và nhu cầu kinh doanh của chúng tôi, chúng tôi có thể cần di chuyển tài liệu từ chỉ mục này sang chỉ mục khác theo thời gian. Điều này đặc biệt đúng khi chúng tôi cần di chuyển chỉ mục cũ sang chỉ mục mới hơn do thay đổi trong lược đồ ánh xạ hoặc tập hợp của chúng tôi.tings. Chúng ta có thể sử dụng API \_reindex cho các yêu cầu như vậy, vì định dạng này hiển thị:

POST \_reindex

{

"nguồn": {"index": "<source\_index>"}, "đích": {"index": "<destination\_index>"}

}

Bạn có thể hỏi khi nào chúng ta sẽ đưa việc lập chỉ mục lại vào hoạt động? Giả sử chúng ta muốn cập nhật chỉ mục phim của mình bằng các sửa đổi lược đồ có thể phá vỡ chỉ mục hiện tại nếu chúng ta triển khai chúng trực tiếp trên đó. Trong trường hợp này, ý tưởng là tạo một chỉ mục mới vớithiết lập được cập nhật (ví dụ, chỉ mục movies\_new với lược đồ được cập nhật) và di chuyển dữ liệu từ chỉ mục movies cũ sang chỉ mục mới. Truy vấn trong danh sách sau đây thực hiện chính xác điều đó.

**Liệt kê 5.50 Di chuyển dữ liệu giữa các chỉ mục bằng cách sử dụng\_reindexGiao diện lập trình ứng dụng (API)**

POST \_reindex

{

"nguồn": {"index": "phim"},

"đích": {"index": "phim\_mới"}

}

Truy vấn này lấy ảnh chụp nhanh của chỉ mục phim và đẩy các bản ghi vào chỉ mục mới. Dữ liệu được di chuyển giữa các chỉ mục này như mong đợi.

Một trong những trường hợp sử dụng quan trọng của việc lập chỉ mục lại là di chuyển không có thời gian chết trong sản xuất nếu chúng ta sử dụng phương pháp này với các bí danh. Chúng ta sẽ xem xét điều này trong chương 6.

Đây là một chương dài và chúng ta đã đề cập đến rất nhiều nội dung. Hãy kết thúc chương này và chuyển sang chương tiếp theo, nơi chúng ta thảo luận chi tiết về các hoạt động lập chỉ mục.

#### Bản tóm tắt

* Elasticsearch cung cấp một bộ API hoạt động trên các tài liệu. Chúng ta có thể sử dụng các API này để thực hiện các hành động CRUD (tạo, đọc, cập nhật và xóa) trên từng tài liệu.
* Tài liệu của chúng tôi được lưu trong bộ đệm trong bộ nhớ của mảnh và được đẩy vào một phân đoạn trong quá trình làm mới. Lucene sử dụng chiến lược tạo một phân đoạn mới với các tài liệu trong quá trình làm mới. Sau đó, nó tích lũy hợp nhất ba phân đoạn để tạo thành một phân đoạn mới và quá trình lặp lại.

***Bản tóm tắt* 191**

* + Các tài liệu có mã định danh (ID) sử dụng HTTPĐẶThành động khi lập chỉ mục (ví dụ,ĐẶT <index>/\_doc/<ID>), trong khi các tài liệu không có ID sẽ gọiBƯU KIỆNphương pháp.
  + Elasticsearch tạo ra các mã định danh duy nhất ngẫu nhiên (UUID) và gán chúng cho các tài liệu trong quá trình lập chỉ mục.
  + Để tránh ghi đè lên tài liệu, chúng ta có thể đưa ra các lệnh sau:
    - \_tạo nên—API này sẽ báo lỗi nếu tài liệu đã tồn tại.
    - \_mget—API này cho phép chúng tôi truy xuất nhiều tài liệu cùng một lúc, dựa trên ID của chúng.
    - \_số lượng lớn—API này thực hiện các hoạt động liên quan đến tài liệu như lập chỉ mục, xóa và cập nhật nhiều tài liệu trong một lệnh gọi.
  + Chúng ta có thể điều chỉnh nguồn và siêu dữ liệu được truy xuất do kết quả của các lệnh gọi truy vấn và sau đó tùy chỉnh nguồn tài liệu được trả về để bao gồm và/hoặc loại trừ các trường bằng cách đặt\_nguồn\_bao gồmVà\_nguồn\_loại\_trừtính chất tương ứng.
  + Các\_cập nhậtAPI cho phép chúng ta sửa đổi một tài liệu hiện có bằng cách cập nhật và thêm các trường. Các bản cập nhật dự kiến ​​được gói trong mộttài liệuđối tượng và được truyền dưới dạng nội dung yêu cầu.
  + Nhiều tài liệu có thể được sửa đổi bằng cách xây dựng một\_cập\_nhật\_theo\_truy\_cập

truy vấn.

* + Các bản cập nhật theo kịch bản cho phép chúng tôi sửa đổi các tài liệu dựa trên các điều kiện. Nếu mệnh đề điều kiện được đề cập trong nội dung yêu cầu được đánh giá đểĐÚNG VẬY, kịch bản được đưa vào thực hiện.
  + Tài liệu có thể được xóa bằng HTTPXÓA BỎcho một tài liệu duy nhất hoặc bằng cách chạy một\_xóa\_bởi\_truy\_cậpphương pháp trên nhiều tài liệu.
  + Việc di chuyển dữ liệu giữa các chỉ mục được thực hiện bởi API lập chỉ mục lại.

Lệnh gọi API \_reindex mong muốn được cung cấp chỉ mục nguồn và đích để truyền dữ liệu.

*Hoạt động lập chỉ mục*

***Chương này bao gồm***

* Các hoạt động lập chỉ mục cơ bản
* Mẫu chỉ mục
* Quản lý và giám sát trạng thái
* Quản lý vòng đời chỉ mục (ILM)

Trong vài chương trước, chúng ta đã làm việc với các chỉ mục mà không đi sâu vào các chi tiết phức tạp. Mặc dù điều đó đủ để bắt đầu với Elasticsearch, nhưng nó còn lâu mới lý tưởng. Việc định cấu hình một chỉ mục với các thiết lập phù hợp không chỉ giúp Elasticsearch chạy hiệu quả mà còn tăng khả năng phục hồi. Một chiến lược lập chỉ mục tổ chức hợp lý sẽ tạo ra một công cụ tìm kiếm có khả năng thích ứng với tương lai và do đó, mang lại trải nghiệm người dùng mượt mà hơn.

Vìmột cụm Elasticsearch lành mạnh và hiệu suất cao, chúng ta cần làm việc với các chỉ mục ở cấp độ thấp hơn. Hiểu sâu về quản lý chỉ mục sẽ giúp ích khi thiết lập hệ thống tìm kiếm linh hoạt và mạch lạc. Chương này dành riêng cho các hoạt động lập chỉ mục và hiểu chi tiết về cơ chế, API lập chỉ mục và hoạt động bên trong.

Chúng ta bắt đầu bằng cách khám phá các thiết lập cấu hình trên một chỉ mục. Các chỉ mục đi kèm với ba bộ cấu hình: thiết lập, ánh xạ và bí danh. Mỗi cấu hình

**192**

***6.2 Tạochỉ số* 193**

sửa đổi chỉ mục theo cách này hay cách khác. Ví dụ, chúng tôi sử dụng các thiết lập để điều chỉnh số lượng phân đoạn và bản sao và để thay đổi các thuộc tính chỉ mục khác. Phân đoạn và bản sao cho phép mở rộng quy mô và tính khả dụng cao của dữ liệu. Ánh xạ xác định một lược đồ hiệu quả cho dữ liệu của chúng tôi để lập chỉ mục và truy vấn dữ liệu hiệu quả. Biệt danh (tên thay thế được đặt cho chỉ mục) cho phép chúng tôi truy vấn dễ dàng trên nhiều chỉ mục và lập chỉ mục lại dữ liệu mà không có thời gian chết. Chúng tôi sẽ trình bày chi tiết tất cả các cấu hình này trong phần đầu tiên của chương.

Mặc dù có thể khởi tạo các chỉ mục theo cách thủ công, nhưng đây là một quá trình tẻ nhạt, không hiệu quả và đôi khi sai sót. Thay vào đó, các tổ chức nên phấn đấu cho một chiến lược phát triển các chỉ mục bằng cách sử dụng các mẫu chỉ mục. Với các mẫu chỉ mục, chúng ta có thể tạo các chỉ mục với cấu hình được xác định trước; và hiểu được cơ chế mẫu cho phép chúng ta phát triển các chỉ mục cho các hoạt động nâng cao như chuyển tiếp. Phần 6.6 xem xét các mẫu này và các quy trình tạo mẫu trong hành động.

Các chỉ mục tăng theo thời gian cùng với dữ liệu, nếu không được kiểm tra có thể khiến hệ thống không phản hồi. Elasticsearch cung cấp một cơ chế để tạo các định nghĩa chính sách vòng đời nhằm giúp quản lý và giám sát các chỉ mục một cách hiệu quả. Khi một chỉ mục già đi hoặc đạt đến một kích thước nhất định, nó có thể được chuyển sang một chỉ mục mới, do đó ngăn ngừa các trường hợp ngoại lệ không thể tránh khỏi.

Tương tự như vậy, các chỉ mục lớn hơn được tạo ra để dự đoán nhiều dữ liệu hơn có thể được tự động hủy sau một khoảng thời gian nhất định. Mặc dù quản lý vòng đời chỉ mục là một chủ đề nâng cao, nhưng nó vừa thú vị vừa hấp dẫn. Chúng tôi sẽ tìm hiểu các tùy chọn quản lý vòng đời chỉ mục trong phần 6.9. Chương này đào sâu vào quản lý lập chỉ mục, giám sát chỉ mục và vòng đời chỉ mục, vì vậy chúng ta hãy bắt đầu ngay.

**GHI CHÚ**Mã cho chương này có sẵn trên GitHub (<http://mng.bz/Pzyg>) và trên trang web của cuốn sách ([https://www.manning.com/books/elasticsearch](https://www.manning.com/books/elasticsearch-in-action-second-edition)

[-in-action-second-edition](https://www.manning.com/books/elasticsearch-in-action-second-edition)).

#### Hoạt động lập chỉ mục

Chúng ta hãy nhanh chóng tóm tắt lại index là gì: đó là một tập hợp hợp lý dữ liệu của chúng ta được sao lưu bởi các shard (chính và bản sao). Các tài liệu được biểu diễn dưới dạng JSON và có các thuộc tính tương tự (nhân viên, đơn hàng, dữ liệu kiểm tra đăng nhập, tin tức theo khu vực, v.v.) được lưu trữ trong các index tương ứng. Bất kỳ index nào bao gồm các shard đều được phân phối trên nhiều nút khác nhau trong cụm. Một index mới được tạo sẽ được liên kết với một số lượng shard, bản sao và các thuộc tính khác theo mặc định.

Chúng tôi đưa các chỉ mục vào cuộc sống với các cấu hình tùy chỉnh. Chúng tôi có thể thực hiện nhiều thao tác khi phát triển một chỉ mục, từ việc tạo chỉ mục đến đóng, thu nhỏ, sao chép, đóng băng, xóa và các thao tác khác. Hiểu các thao tác này cho phép chúng tôi thiết lập hệ thống để lưu trữ dữ liệu hiệu quả và tìm kiếm truy xuất. Hãy bắt đầu bằng cách xem xét việc tạo chỉ mục và các thao tác liên quan đến việc khởi tạo chúng.

#### Tạo chỉ mục

Khi chúng ta lập chỉ mục một tài liệu lần đầu tiên trong các chương trước, Elasticsearch cũng tạo chỉ mục một cách ngầm định. Đây là một trong những cách chúng ta có thể tạo chỉ mục. Một

**194 CPHẦN6*Hoạt động lập chỉ mục***

giải pháp thay thế là tạo chỉ mục một cách rõ ràng; chúng ta có nhiều quyền kiểm soát hơn đối với việc tùy chỉnh chỉ mục khi chúng ta tạo chúng thông qua tuyến đường đó. Hãy xem xét cả hai cách tiếp cận:

* *Tạo ngầm định (tự động)*—Khi lập chỉ mục một tài liệu lần đầu tiên, nếu chỉ mục không tồn tại, Elasticsearch sẽ tạo chỉ mục đó một cách ngầm định với các thiết lập mặc định. Phương pháp tạo chỉ mục này thường hoạt động tốt, nhưng cần cẩn thận khi sử dụng phương pháp này trong sản xuất vì các chỉ mục không chính xác hoặc không được tối ưu hóa sẽ gây ra hậu quả không mong muốn cho hệ thống đang chạy.

Elasticsearch sử dụng ánh xạ động để suy ra các kiểu trường khi tạo lược đồ ánh xạ bằng phương pháp này. Thật không may, các định nghĩa ánh xạ được tạo ra không phải là hoàn hảo; ví dụ, dữ liệu ở định dạng ngày không phải ISO (dd-MM-yyyy hoặc MM-dd-yyyy) được xác định là trường văn bản chứ không phải là kiểu dữ liệu ngày tháng.

* *Tạo rõ ràng (thủ công)*—Chọn cách tiếp cận này cho phép chúng ta kiểm soát việc tạo chỉ mục để chúng ta có thể tùy chỉnh chỉ mục khi cần. Chúng ta có thể định cấu hình chỉ mục bằng lược đồ ánh xạ do các kiến ​​trúc sư dữ liệu thường trú chỉnh sửa, phân bổ các phân đoạn dựa trên kỳ vọng lưu trữ hiện tại và dự kiến, v.v.

Elasticsearch cung cấp một bộ API tạo chỉ mục giúp chúng ta tạo chỉ mục với cấu hình được cá nhân hóa. Chúng ta có thể tận dụng các API này khi tạo chỉ mục trước để các chỉ mục được tối ưu hóa cho việc lưu trữ và truy xuất dữ liệu. Các API cung cấp tính linh hoạt tuyệt vời; ví dụ, chúng ta có thể tạo một chỉ mục riêng lẻ với các tính năng như phân đoạn có kích thước phù hợp, định nghĩa ánh xạ áp dụng, nhiều bí danh, v.v.

**GHI CHÚ** Để kiểm soát việc tạo chỉ mục tự động, chúng ta có thể tắt việc tạo chỉ mụcbằng cách đặt cờ action.auto\_create\_index thành false thông qua API cài đặt cụm hoặc đặt thuộc tính này trong config/elasticsearch.yml. Theo mặc định, cờ này được đặt thành true. Chúng tôi sẽ sử dụng tính năng này trong thời gian ngắn.

* + 1. Tạo chỉ mục ngầm định (tạo tự động)

Khi chúng ta lập chỉ mục một tài liệu lần đầu tiên, Elasticsearch không phàn nàn về một chỉ mục không tồn tại; thay vào đó, nó vui vẻ tạo một chỉ mục cho chúng ta. Khi một chỉ mục được tạo theo cách này, Elasticsearch sử dụng các thiết lập mặc định như thiết lập số lượng phân đoạn chính và phân đoạn bản sao thành một. Để chứng minh, chúng ta hãy nhanh chóng lập chỉ mục một tài liệu có thông tin về ô tô bằng API tài liệu. (Chúng ta đã tạo chỉ mục ô tô trong chương 4. Chúng ta xóa và tạo lại chỉ mục đó để bắt đầu lại trong chương này.)

**Liệt kê 6.1 Tài liệu đầu tiên có chứa ô tôdữ liệu**

XÓA xe ô tô

ĐẶT xe/\_doc/1

{

"làm":"Maserati",

**Xóa bỏ xe ô tô chỉ số vì vậy chúng ta bắt đầu từ đầu**

**Chỉ mục không tồn tại nhưng được tạo ra lần đầu tiên khi một tài liệu được lập chỉ mục. Bằng cách lập chỉ mục một tài liệu, chúng ta ngầm định tạo ra chỉ mục cars.**

"model":"GranTurismo Sport",

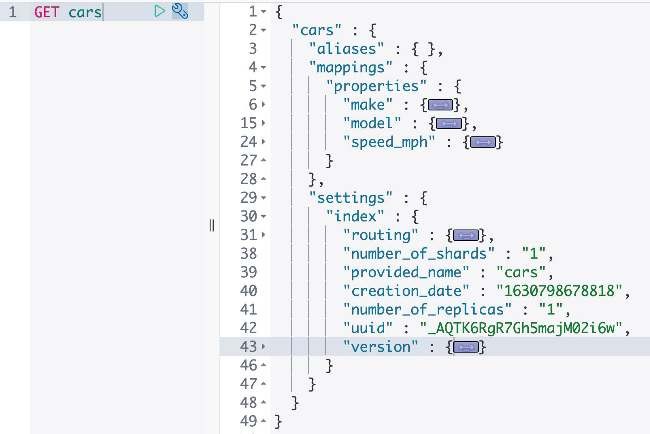
***6.2 Tạochỉ số* 195**

"tốc độ\_mph":186

}

Vì đây là tài liệu đầu tiên được lưu trữ trong chỉ mục cars, khi chúng ta gửi yêu cầu này đến Elasticsearch, máy chủ sẽ ngay lập tức tạo một chỉ mục có tên cars vì chỉ mục đó không tồn tại trong kho. Chỉ mục được cấu hình với các thiết lập mặc định và ID tài liệu là 1. Chúng ta có thể lấy thông tin chi tiết về chỉ mục mới được tạo bằng cách gọi lệnh GET cars, như hình 6.1 cho thấy.

**CácNHẬN xe ô tôlệnh lấy thông tin chi tiết về chỉ mục.**



**Mỗi chỉ mục được tạo thành từ ba thành phần: bí danh, ánh xạ và cài đặt.**

**số\_mảnh\_shardsVàsố\_bản\_sao\_lạiđược thiết lập để1theo mặc định.**

**Hình 6.1 Lấy thông tin chi tiết củaxe ô tôchỉ số**

Chúng ta nên lưu ý một vài điều quan trọng từ phản hồi: mỗi chỉ mục được tạo thành từ các lệnh ánh xạ, cài đặt và bí danh. Elasticsearch tự động tạo lược đồ ánh xạ bằng cách xác định kiểu dữ liệu của từng trường từ giá trị của trường. Ví dụ, vìcác kiểu make và model dường như chứa thông tin dạng văn bản, các trường này được tạo dưới dạng trường văn bản. Ngoài ra, Elasticsearch tự do phân bổ một phân đoạn chính và một phân đoạn bản sao theo mặc định.

**Không thể thay đổi cài đặt tĩnh trên chỉ mục hoạt động**

Không phải tất cả các thiết lập mặc định (số\_mảnh\_shards,số\_bản\_sao\_lại, v.v.) được áp dụng bởi động cơ có thể được thay đổi trên một chỉ số đang hoạt động. Ví dụ,số\_bản\_sao\_lạicài đặt có thể được sửa đổi trên một chỉ mục trực tiếp, nhưngsố\_mảnh\_shardskhông thể.Chúng ta cần đưa phân đoạn đó ngoại tuyến để thay đổi số phân đoạn chính và các thiết lập tĩnh khác.

**196 CPHẦN6*Hoạt động lập chỉ mục***

**D****BẬT TỰ ĐỘNG-TẠO CHO MỘT TÍNH NĂNG CHỈ MỤC**

Như đã đề cập trước đó, Elasticsearch cho phép chúng ta chặn việc tự động tạo chỉ mục bằngđặt thuộc tính action.auto\_create\_index thành false (mặc định là true). Chúng ta có thể điều chỉnh cài đặt này để sửa đổi giá trị của cờ bằng cách gọi thay đổi thuộc tính trên toàn cụm bằng API cài đặt cụm. Danh sách sau đây vô hiệu hóa tính năng này.

**Liệt kê 6.2 Vô hiệu hóa việc tạo tự độngchỉ số**

ĐẶT \_cluster/cài đặt

{

"bền bỉ": {

**Cập nhật các thiết lập trên toàn bộ cụm**

**Những thay đổi có thể là**



"action.auto\_create\_index":sai

}

} **Tắt chức năng tự động tạo**

**dai dẳng hoặc tạm thời.**

Sự dai dẳngthuộc tính chỉ ra rằng các thiết lập sẽ là vĩnh viễn. Mặt khácMặt khác, việc sử dụng thuộc tính tạm thời chỉ lưu các thiết lập cho đến lần khởi động lại máy chủ Elasticsearch tiếp theo.

Mặc dù việc vô hiệu hóa tính năng này nghe có vẻ hay, nhưng trên thực tế, việc làm như vậy là không nên. Chúng tôi đang hạn chế việc tạo tự động trên bất kỳ chỉ mục nào, nhưng một ứng dụng hoặc Kibana có thể cần tạo chỉ mục cho mục đích quản trị; Kibana thường tạo các chỉ mục ẩn. (Một dấu chấm được đặt trước tên chỉ mục được coi là chỉ mục ẩn:

.hồ sơ người dùng,.quản trị viênvân vân.)

Thay vì tắt hoàn toàn việc tạo chỉ mục tự động, có một cách để điều chỉnh thuộc tính này ngoài các tùy chọn nhị phân. Thay vào đó, chúng ta có thể cung cấp một tập hợp các chỉ mục được chọn lọc thông qua các biểu thức chính quy được phân tách bằng dấu phẩy cho phép (hoặc không cho phép) thay đổi này. Sau đây là một ví dụ:

action.auto\_create\_index: [".admin\*, cars\*, "\*books\*", "-x\*","-y\*","+z\*"]

Thiết lập này cho phép tự động tạo các chỉ mục ẩn với tiền tố là admin, cũng như bất kỳ chỉ mục nào có tiền tố là cars hoặc books và những chỉ mục theo sau dấu cộng (+).Tuy nhiên, thiết lập này không cho phép bất kỳ chỉ mục nào bắt đầu bằng x hoặc y được tự động tạo ra vì dấu trừ (-) chỉ ra rằng việc tạo chỉ mục tự động bị vô hiệu hóa. Nó cũng có nghĩa là bất kỳ chỉ mục nào khác không khớp với mẫu này sẽ không được tự động khởi tạo. Ví dụ, nếu chúng ta cố gắng lập chỉ mục một tài liệu vào chỉ mục các chuyến bay, việc tạo chỉ mục sẽ không thành công vì tên chỉ mục không khớp với các biểu thức chính quy mà chúng ta vừa định nghĩa (sẽ nói thêm về điều này sau). Sau đây là ngoại lệ do công cụ đưa ra:

không có chỉ mục nào như vậy [flights] và [action.auto\_create\_index] ([.admin\*, cars\*, \*books\*, -x\*,-y\*,+z\*]) không khớp

Cho phép máy chủ tạo chỉ mục thúc đẩy quá trình phát triển nhanh chóng. Tuy nhiên, chúng ta hiếm khi đưa vào sản xuất mà không điều chỉnh một số thuộc tính này. Ví dụ, chúng ta có thể quyết định có chiến lược gồm 10 phân đoạn chính với 2 bản sao cho mỗi phân đoạn, trong trường hợp đó, chúng ta phải thay đổi cài đặt (sẽ là thảm họa nếu thiết kế dịch vụ tìm kiếm chỉ với một phân đoạn chính). Ngoài ra, như chúng ta đã học trong chương 4, Elasticsearch có thể

***6.2 Tạochỉ số* 197**

không thể suy ra đúng các kiểu dữ liệu dựa trên các giá trị trường của tài liệu. Các kiểu dữ liệu không chính xác sẽ dẫn đến thất bại trong quá trình tìm kiếm.

May mắn thay, Elasticsearch cho phép chúng ta tạo các chỉ mục để đáp ứng các yêu cầu của mình bằng cách cho phép chúng ta cấu hình và khởi tạo chúng một cách rõ ràng theo yêu cầu của chúng ta. Trước khi bắt đầu phát triển các chỉ mục tùy chỉnh, chúng ta cần biết về các cấu hình chỉ mục, được thảo luận trong phần tiếp theo.

**TÔICẤU HÌNH NDEX**

Mỗi chỉ mục được tạo thành từ một cấu hình bao gồm các ánh xạ, cài đặt và bí danh, bất kể nó được tạo tự động hay rõ ràng. Chúng tôi đã đề cập đến ánh xạ trong chương 4; chúng tôi tóm tắt lại ở đây và trình bày hai cấu hình khác:

* *Ánh xạ*—Ánh xạ là quá trình tạo ra định nghĩa lược đồ. Dữ liệu được lưu trữ thường có nhiều kiểu dữ liệu liên kết với các trường của nó, chẳng hạn nhưchữ,từ khóa,dài,ngày, v.v. Elasticsearch tham khảo các định nghĩa ánh xạ để áp dụng các quy tắc thích hợp để phân tích dữ liệu đến trước khi lưu trữ dữ liệu đó để tìm kiếm hiệu quả và hiệu suất. Ví dụ: đoạn mã sau đây đặt ánh xạ chocars\_index\_with\_sample\_mappingmục lục:

ĐẶT cars\_index\_with\_sample\_mapping

{

"ánh xạ": { "thuộc tính": {

"làm":{

"loại": "văn bản"

}

}

}

}

Chúng tôi có thể phát hànhNHẬN cars\_index\_with\_sample\_mapping/\_mappinglệnh để lấy lược đồ cho mục mới tạo của chúng tacars\_index\_with\_sample\_mappingmục lục.

* *Cài đặt*—Mỗi chỉ mục đi kèm với một tập hợp các thiết lập cấu hình, chẳng hạn như số lượng phân đoạn và bản sao, tốc độ làm mới, codec nén và các thiết lập khác. Một số thiết lập (gọi là thiết lập động) có thể được điều chỉnh trên một chỉ mục trực tiếp khi chạy. Các thiết lập khác (thiết lập tĩnh) được áp dụng cho một chỉ mục ở chế độ không hoạt động. Chúng ta sẽ xem xét hai loại này trong thời gian ngắn. Danh sách tiếp theo sẽ cấu hình một chỉ mục với một số thiết lập.

**Liệt kê 6.3 Tạo chỉ mục với tùy chỉnhcài đặt**

ĐẶT cars\_index\_with\_sample\_settings

{

**Tạo chỉ mục**

"cài đặt": {

"số\_bản\_sao\_lại": 3, "codec": "nén\_tốt\_nhất"

**Áp dụng**

**các thiết lập**

}

}

**198 CPHẦN6*Hoạt động lập chỉ mục***

Gọi đếnNHẬN cars\_index\_with\_sample\_settings/\_settingslấy các thiết lập cho chỉ mục này.

* *Bí danh*—Bí danh là tên thay thế được đặt cho các chỉ mục. Một bí danh có thể trỏ đến một hoặc nhiều chỉ mục. Ví dụ, một bí danh có tên bí danh xe của tôicó thể trỏ đến tất cả các chỉ mục xe. Chúng ta cũng có thể thực hiện các truy vấn trên các bí danh như thể chúng ta đang chạy chúng trên các bí danh riêng lẻ. Mã này cho thấy cách tạo một bí danh.

**Liệt kê 6.4 Tạo một chỉ mục vớibí danh**

ĐẶT cars\_index\_with\_sample\_alias

{

"bí danh": {

**Tạo chỉ mục**

**Khai báo đối tượng bí danh**

"alias\_for\_cars\_index\_with\_sample\_alias": {}

}

} **Bản thân bí danh**

**để cấu hình bí danh**

Chúng tôi có thể phát hànhNHẬN cars\_index\_with\_sample\_alias/\_aliasđể lấy bí danh cho chỉ mục này.

Khi chúng ta tạo một chỉ mục một cách rõ ràng, chúng ta có thể thiết lập các ánh xạ, thiết lập và bí danh ngay từ đầu. Theo cách này, chỉ mục được khởi tạo với tất cả các cấu hình cần thiết tại chỗ. Tất nhiên, chúng ta có thể sửa đổi một số cấu hình này khi chạy (chúng ta cũng có thể điều chỉnh các chỉ mục đã đóng [không hoạt động]. Trong phần tiếp theo, chúng ta sẽ xem cách thiết lập các cấu hình này trên các chỉ mục được tạo một cách rõ ràng.

* + 1. Tạo chỉ mục một cách rõ ràng

Các chỉ mục được tạo ngầm định hiếm khi sẵn sàng cho cấu hình sản xuất. Tạo chỉ mục một cách rõ ràng có nghĩa là chúng ta phải thiết lập các cấu hình tùy chỉnh. Chúng ta có thể chỉ đạo Elasticsearch cấu hình chỉ mục với các ánh xạ và cài đặt cần thiết, cũng như các bí danh, thay vì phụ thuộc vào các mặc định.

Chúng ta đã biết rằng việc tạo một chỉ mục rất dễ dàng: chỉ cần phát hành PUT <index\_name>. Lệnh này tạo một chỉ mục mới với cấu hình mặc định (tương tự như chỉ mục được tạo khi một tài liệu được lập chỉ mục lần đầu tiên). Ví dụ: PUT carstạo một chỉ mục cars và phát hành GET cars trả về chỉ mục. Hãy xem cách chúng ta có thể quản lý các chỉ mục này bằng các cấu hình tùy chỉnh.

* + 1. Chỉ mục với các thiết lập tùy chỉnh

Mỗi chỉ mục có thể được khởi tạo bằng các thiết lập (mặc định hoặc tùy chỉnh) trong quá trình tạo. Chúng ta cũng có thể thay đổi một số thiết lập khi chỉ mục vẫn đang hoạt động. Vì mục đích này, Elasticsearch sẽ đưa API \_settings vào để cập nhật các thiết lập trên chỉ mục trực tiếp. Tuy nhiên, như chúng tôi đã đề cập, không phải tất cả các thuộc tính đều có thể thay đổi trên chỉ mục trực tiếp—chỉ có các thuộc tính động. Điều đó đưa chúng ta đến một cuộc thảo luận ngắn gọn về hai loại thiết lập chỉ mục, tĩnh và động:

* + - * *Cài đặt tĩnh*—Thiết lập tĩnh là những thiết lập được (và có thể được) áp dụng trong quá trình tạo chỉ mục và không thể thay đổi trong khi chỉ mục đang hoạt động. Những thiết lập này

***6.2 Tạochỉ số* 199**

là các thuộc tính như số lượng phân đoạn, codec nén, kiểm tra dữ liệu khi khởi động, v.v. Nếu chúng ta muốn thay đổi các thiết lập tĩnh của một chỉ mục trực tiếp, chúng ta phải đóng chỉ mục để áp dụng lại các thiết lập hoặc tạo lại chỉ mục với các thiết lập mới.

Luôn tốt nhất là khởi tạo một chỉ mục với các thiết lập tĩnh cần thiết vì việc áp dụng chúng như một suy nghĩ sau đó đòi hỏi phải tắt chỉ mục. Tuy nhiên, có nhiều cách để quản lý việc nâng cấp chỉ mục (reindexing là một hình thức nâng cấp) mà không có thời gian chết (chúng ta đã xem xét tính năng reindexing trong chương trước—phần 5.7).

* + - * + *Thiết lập động*—Thiết lập động có thể được áp dụng trên một chỉ mục trực tiếp (hoạt động). Ví dụ, chúng ta có thể thay đổi các thuộc tính như số lượng bản sao, cho phép hoặc không cho phép ghi, khoảng thời gian làm mới và các thuộc tính khác trên các chỉ mục đang hoạt động.

Một số ít các thiết lập nằm trong cả hai trường hợp, do đó, hiểu biết sâu sắc về từng loại sẽ giúp ích về lâu dài. Hãy cùng xem cách chúng ta có thể khởi tạo một chỉ mục với một số thiết lập tĩnh.

**GHI CHÚ**Bạn có thể tham khảo tài liệu chính thức để tìm hiểu thêm về các thiết lập tĩnh và động mà Elasticsearch hỗ trợ:<http://mng.bz/1qXR>.

Chúng tôi muốn tạo một chỉ mục với các thuộc tính sau: ba phân đoạn với năm bản sao cho mỗi phân đoạn, codec nén, số lượng trường tập lệnh tối đa và làm mớikhoảng thời gian. Để áp dụng các thiết lập cấu hình này vào chỉ mục của chúng tôi, chúng tôi sử dụng một đối tượng thiết lập.

**Liệt kê 6.5 Tạo chỉ mục với tùy chỉnhcài đặt**

**Đặt số lượng mảnh vỡ thành 3**

ĐẶT xe\_có\_cài\_đặt\_tùy\_chỉnh

{

"cài đặt":{ "số\_phân\_mảnh":3, "số\_bản\_sao\_lại":5, "codec": "nén\_tốt\_nhất", "trường\_script\_tối\_đa":128, "khoảng\_thời\_gian\_làm\_mới": "60 giây"

**Tạo một chỉ mụcvới các thiết lập tùy chỉnh**

**Đối tượng cài đặt bao gồm các thuộc tính cần thiết.**

**Thay đổi độ nén từ giá trị mặc định của nó**

**Đặt số lượng bản sao thành 5**

}

} **Thay đổi khoảng thời gian làm mớitừ mặc định của nó là 1 giây**

**Tăng số lượng tối đa**

**trường tập lệnh từ mặc định của nó là 32**

Chúng tôi hướng dẫn Elasticsearch tạo một chỉ mục với các thiết lập mà chúng tôi cho là cần thiết dựa trên yêu cầu của chúng tôi. Việc phát hành lệnh GET cars\_with\_custom\_settings sẽ lấy thông tin chi tiết về chỉ mục, phản ánh các thiết lập tùy chỉnh cho các phân đoạn và bản sao mà chúng tôi đã thiết lập trước đó.

Như đã đề cập, một số thiết lập được ghi trên đá sau khi chỉ mục hoạt động (thiết lập tĩnh) và một số khác (thiết lập động) có thể được thay đổi trên chỉ mục đang chạy. Nếu chúng ta thửthay đổi bất kỳ thuộc tính tĩnh nào (ví dụ: thuộc tính number\_of\_shards) trên một chỉ mục trực tiếp, Elasticsearch sẽ đưa ra một ngoại lệ cho biết rằng nó không thể cập nhật các thuộc tính không động

**200 CPHẦN6*Hoạt động lập chỉ mục***

cài đặt. Chúng ta có thể cập nhật cài đặt động bằng cách sử dụng điểm cuối \_settings, như được hiển thị trong danh sách sau.

**Liệt kê 6.6 Cập nhật thuộc tính động trên một chỉ mục**

ĐẶT xe\_với\_cài\_đặt\_tùy\_chỉnh/\_cài\_đặt

{

"cài đặt": { "số\_bản\_sao\_lẻ": 2

}

}

Thuộc tính number\_of\_replicas là thuộc tính động, do đó không quan trọng liệu chỉ mục có hoạt động hay không; thuộc tính này được áp dụng ngay lập tức.

**GHI CHÚ** Elasticsearch không cho phép chúng ta thay đổi số lượng phân mảnh sau khi chỉ mục hoạt động. Có một lý do đơn giản nhưng hợp lệ cho việc này: trang chủ của tài liệu được lấy từ hàm định tuyến này: shard\_home = hash(doc\_ID)

% number\_of\_shards. Chức năng định tuyến phụ thuộc vào số lượng shards, do đó việc sửa đổi số lượng shard (tức là thay đổi number\_of\_shards) sẽ làm hỏng chức năng định tuyến vì các tài liệu hiện có có thể bị đặt sai vị trí hoặc được gán không đúng cho một shard.

Nếu chúng ta muốn cấu hình lại các chỉ mục với các thiết lập mới, chúng ta phải thực hiện một số bước sau:

**1** Đóng chỉ mục hiện tại (chỉ mục không hỗ trợ hoạt động đọc/ghi).

**2** Tạo một chỉ mục mới với các thiết lập mới.

**3**Di chuyển dữ liệu từ chỉ mục cũ sang chỉ mục mới (hoạt động lập chỉ mục lại).

**4** Trỏ lại bí danh tới chỉ mục mới (giả sử chỉ mục đã có bí danh hiện tại).

Chúng ta đã thấy các hoạt động lập chỉ mục lại trong chương trước (Phần 5.7) đang diễn ra. Chúng ta cũng có một phần về lập chỉ mục lại dữ liệu với thời gian chết bằng không được thảo luận trong phần 6.2.3.

Việc lấy các thiết lập cấu hình là một công việc đơn giản: chỉ cần đưa ra lệnh GET

lời yêu cầu.

**Liệt kê 6.7 Lấy các thiết lập của mộtchỉ số**

NHẬN cars\_with\_custom\_settings/\_settings

Chúng ta cũng có thể lấy các thiết lập của nhiều chỉ mục bằng cách sử dụng các chỉ mục được phân tách bằng dấu phẩy hoặc các mẫu ký tự đại diện trên tên chỉ mục. Danh sách sau đây cho biết cách thực hiện việc này (mã có sẵn trong các tệp của sách).

**Liệt kê 6.8 Lấy các thiết lập của nhiều chỉ mục cùng một lúc**

NHẬN cars1,cars2,cars3/\_settings NHẬN cars\*/\_settings

**Lấy nhiều thiết lập chỉ mục**

**Lấy các thiết lập cho chỉ mục được xác định bằng ký tự đại diện (\*)**

***6.2 Tạochỉ số* 201**

Chúng ta cũng có thể lấy một thuộc tính duy nhất. Ví dụ, danh sách này hiển thị yêu cầu lấy số lượng phân đoạn.

**Liệt kê 6.9 Lấy mộtthuộc tính**

NHẬN cars\_with\_custom\_settings/\_settings/index.number\_of\_shards

Ở đây, các thuộc tính được bao bọc trong một chỉ mục đối tượng bên trong, vì vậy chúng ta phải thêm tiền tố thuộc tính bằng đối tượng cấp cao nhất như thế này: index.<tên\_thuộc\_tính>.

* + 1. Chỉ mục với ánh xạ

Ngoài các thiết lập, chúng ta có thể cung cấp ánh xạ trường khi tạo chỉ mục. Đây là cách chúng ta tạo lược đồ cho mô hình dữ liệu của mình. Danh sách sau đây cho thấy cơ chế tạo chỉ mục với định nghĩa ánh xạ cho loại xe (thực tế làchỉ mục cars\_with\_mappings, nhưng để ngắn gọn, chúng ta có thể cho rằng chỉ mục này xem xét các thực thể car (xe hơi) với các thuộc tính make, model và registration\_year.

**Liệt kê 6.10 Tạo chỉ mục với các ánh xạ trường cho mộtxe hơitài liệu**

ĐẶT xe\_có\_ánh\_sáng

{

"ánh xạ": { "thuộc tính": {

"làm":{

"loại": "văn bản"

},

"người mẫu":{

"loại": "văn bản"

},

**Đối tượng ánh xạ**

**bao gồmcủa cải. Các trường có dữ liệu xe hơi**

**các kiểu được khai báo ở đây.**

**Khai báo make như một kiểu văn bản**

**Khai báo registration\_year là**

"năm đăng ký":{"loại": "ngày",

"định dạng": "dd-MM-yyyy"

}

}

}

}

**một loại ngày**

**Định dạng tùy chỉnh của trường**

Tất nhiên, chúng ta cũng có thể kết hợp cài đặt và ánh xạ. Danh sách tiếp theo cho thấy cách tiếp cận này.

 **Liệt kê 6.11 Tạo chỉ mục với cả cài đặt vàánh xạ**

PUT cars\_with\_settings\_and\_mappings

{

**Chỉ mục với cả cài đặt và ánh xạ**

"cài đặt": {

"số\_bản\_sao\_lại": 3

**Cài đặttrên chỉ số**

},

"ánh xạ": { "thuộc tính": {

"làm":{

**Định nghĩa lược đồ ánh xạ**

**202 CPHẦN6*Hoạt động lập chỉ mục***

"loại": "văn bản"

},

"người mẫu":{

"loại": "văn bản"

},

"năm đăng ký":{"loại": "ngày",

"định dạng": "dd-MM-yyyy"

}

}

}

}

Bây giờ chúng ta đã biết cách thiết lập chỉ mục với các cài đặt và ánh xạ, phần cuối cùng trong bức tranh ghép hình là tạo bí danh.

* + 1. Chỉ mục với bí danh

Bí danh là tên thay thế được đặt cho các chỉ mục cho nhiều mục đích khác nhau như tìm kiếm hoặc tổng hợp dữ liệu từ nhiều chỉ mục (như một bí danh duy nhất) hoặc cho phép không có thời gian chết trong quá trình lập chỉ mục lại. Sau khi tạo bí danh, chúng ta có thể sử dụng bí danh đó để lập chỉ mục, truy vấn và mọi mục đích khác như thể đó là một chỉ mục. Bí danh rất tiện lợi trong quá trình phát triển cũng như trong quá trình sản xuất. Chúng ta cũng có thể nhóm nhiều chỉ mục và gán bí danh cho chúng để có thể viết các truy vấn đối với một bí danh duy nhất thay vì hàng chục chỉ mục.

Để tạo một chỉ mục tương tự như chỉ mục mà chúng ta đã thấy khi cấu hình chỉ mục bằng các thiết lập và ánh xạ (liệt kê 6.11), chúng ta có thể đặt thông tin bí danh trong một đối tượng bí danh.

**Liệt kê 6.12 Tạo bí danh bằng cách sử dụngbí danhsự vật**

ĐẶT xe\_cho\_biệt\_diện

{

"bí danh": { "my\_new\_cars\_alias": {}

}

**Tạo một chỉ mục với một bí danh**

**Điểm bí danhđến chỉ mục**

}

Tuy nhiên, có một cách khác để tạo bí danh, thay vì sử dụng API chỉ mục: sử dụng API bí danh. Elasticsearch hiển thị API bí danh và cú pháp như sau (kiểm tra

Điểm cuối \_alias được tô đậm):

PUT|POST <tên\_chỉ\_mục>/\_bí\_mật/<tên\_bí\_mật>

Hãy tạo một bí danh có tên làbí danh xe của tôiđiều đó chỉ ra chúng taxe\_cho\_bí\_danh

mục lục.

**Liệt kê 6.13 Tạo bí danh bằng cách sử dụng\_bí danhđiểm cuối**

ĐẶT cars\_for\_aliases/\_alias/my\_cars\_alias

***6.2 Tạochỉ số* 203**

Như thể hiện trong hình 6.2, my\_cars\_alias là tên thay thế (thứ hai) của chỉ mục cars\_ for\_aliases. Tất cả các hoạt động truy vấn được thực hiện trên chỉ mục cho đến nay có thể được chuyển hướng đến my\_cars\_alias trong tương lai. Ví dụ, chúng ta có thể lập chỉ mục một tài liệu hoặc tìm kiếm trên bí danh này, như thể hiện trong hình.

**Tất cả các truy vấn tìm kiếm và lập chỉ mục có thể được xử lý**

MỤC LỤC

**Tạo một bí danh bằng cách sử dụng\_bí danhGiao diện lập trình ứng dụng (API)**

**bằng bí danh.**

BIỆT DANH

ĐẶT cars\_for\_aliases /\_alias/my\_cars\_alias

xe\_cho\_bí\_danh

bí danh xe của tôi

ĐẶT my\_cars\_alias/\_doc/1

{

"làm":"BMW":

}

NHẬN my\_cars\_alias/\_search

{

"truy vấn": {

"cuộc thi đấu": {

"làm": "BMW"



**Lập chỉ mục một tài liệu mới bằng cách sử dụng bí danh**

}

}

}

**Tìm kiếm trên bí danh**

**Hình 6.2 Tạo bí danh cho chỉ mục hiện có**

Chúng ta cũng có thể tạo một alias duy nhất trỏ đến nhiều index (xem hình 6.3), bao gồm các index được cung cấp với một ký tự đại diện. Liệt kê 6.14 hiển thị mã tạo alias (multi\_cars\_alias). Đến lượt mình, alias này trỏ đến nhiều index (cars1,cars2,cars3). Lưu ý rằng một trong những index này phải là write index.

MỤC LỤC

xe ô tô1

BIỆT DANH

xe ô tô2

biệt danh nhiều xe

ĐẶT cars1,cars2,car3/\_alias/multi\_cars\_alias

xe ô tô3

**Hình 6.3 Tạo một bí danh trỏ đến nhiều chỉ mục**

**Liệt kê 6.14 Tạo một bí danh duy nhất trỏ đến nhiềuchỉ số**

ĐẶT cars1,cars2,cars3/\_alias/multi\_cars\_alias

**Phân cách bằng dấu phẩydanh sách chỉ số**

**204 CPHẦN6*Hoạt động lập chỉ mục***

Khi tạo một bí danh trỏ đến nhiều chỉ mục, ít nhất một chỉ mục phải cólà\_ghi\_chỉ\_sốthuộc tính được thiết lập thànhĐÚNG VẬY. Ví dụ, đoạn mã này tạo ra mộtxe ô tô3chỉ số vớilà\_ghi\_chỉ\_sốđã bật:

ĐẶT xe ô tô3

{

"bí danh": { "bí danh xe hơi": { "is\_write\_index":đúng

}

}

}

Tương tự như vậy, chúng ta có thể tạo một bí danh trỏ đến nhiều chỉ mục bằng cách sử dụng ký tự đại diện (chúng ta phải đảm bảo rằng một trong các chỉ mục có thuộc tính is\_write\_index được đặt thành true).

**Liệt kê 6.15 Tạo một bí danh vớiký tự đại diện**

PUT xe hơi\*/\_alias/wildcard\_cars\_alias

**Tất cả các chỉ mục có tiền tố là ô tô**

Sau khi các bí danh được tạo, việc lấy thông tin chi tiết về chỉ mục (GET <tên\_bí\_bí\_bí\_bí>) sẽ phản ánh các bí danh được xác định trên chỉ mục. GET cars trả về chỉ mục có tất cả các bí danh được tạo trên chỉ mục đó (ngoài tất cả các ánh xạ và cài đặt).

**Liệt kê 6.16 Lấy các bí danh, cài đặt và ánh xạ cho mộtchỉ số**

NHẬN xe ô tô

Bây giờ chúng ta đã hiểu cơ chế tạo bí danh, hãy xem cách lấychi tiết bí danh. Tương tự như cài đặt và ánh xạ, chúng ta có thể gửi yêu cầu GET đến điểm cuối \_alias để lấy thông tin chi tiết về bí danh.

**Liệt kê 6.17 Nhận một bí danh trên mộtchỉ số**

NHẬN my\_cars\_alias/\_alias

Tất nhiên, chúng ta cũng có thể mở rộng lệnh này thành nhiều bí danh.

**Liệt kê 6.18 Lấy các bí danh liên kết với nhiềuchỉ số**

NHẬN all\_cars\_alias,my\_cars\_alias/\_alias

**TôiIGRATING DỮ LIỆU KHÔNG CÓ THỜI GIAN CHẶN LẠI BẰNG CÁCH SỬ DỤNG BÍ DANH**

Thiết lập cấu hình cho các chỉ mục trong sản xuất có thể phải được cập nhật với các thuộc tính mới hơn, có thể là do yêu cầu kinh doanh mới hoặc cải tiến kỹ thuật (hoặc để sửa lỗi). Các thuộc tính mới có thể không tương thích với dữ liệu hiện có trong chỉ mục đó, trong trường hợp đó, chúng ta có thể tạo một chỉ mục với các thiết lập mới và di chuyển dữ liệu từ chỉ mục cũ sang chỉ mục hoàn toàn mới.

* 1. ***Tạochỉ số* 205**

Điều này có vẻ tốt và hay, nhưng một vấn đề tiềm ẩn là các truy vấn đóđược viết theo chỉ mục cũ (ví dụ: GET cars/\_search { .. }) cần được cập nhật vì giờ đây chúng cần chạy theo chỉ mục mới (cars\_new). Nếu các truy vấn này được mã hóa cứng trong mã ứng dụng, chúng ta có thể cần bản phát hành hotfix để đưa vào sản xuất.

Giả sử chúng ta có một chỉ mục có tên là vintage\_cars với dữ liệu về xe cổ và chúng ta được yêu cầu cập nhật chỉ mục. Đây là nơi chúng ta có thể dựa vào bí danh; chúng ta có thể đưa ra một chiến lược với một bí danh trong đầu. Bằng cách thực hiện các bước sau (cũng được minh họa trong hình 6.4), chúng ta có thể đạt được quá trình di chuyển mà không có thời gian chết:

**1** Tạo một bí danh có tên là biệt danh xe cổ điểnđể tham khảo chỉ số hiện tại

xe cổ.

**2**Vì các thuộc tính mới không tương thích với chỉ mục hiện tại, hãy tạo một chỉ mục mới, vintage\_cars\_new, với các thiết lập mới.

**3** Sao chép (tức là lập lại chỉ mục) dữ liệu từ chỉ mục cũ (xe cổ) vào chỉ mục mới (xe\_cổ\_trang\_mới).

**4**Tạo lại bí danh hiện có (biệt danh xe cổ điển), trỏ đến chỉ mục cũ để tham chiếu đến chỉ mục mới. (Xem phần tiếp theo.) Do đó,cổ điển\_xe\_biệt\_diệnbây giờ chỉ vàoxe\_cổ\_trang\_mới.

**5**Tất cả các truy vấn được thực hiện đối vớibiệt danh xe cổ điểnhiện đang được thực hiện trên chỉ mục mới.

**6**Xóa chỉ mục cũ (vintage\_cars) khi việc lập chỉ mục lại và phát hành đã được chứng minh là có hiệu quả.

**Tạo một bí danh trỏ đến một chỉ mục hiện có**



xe cổ

biệt danh xe cổ điển

**Tạo một chỉ mục mới vì chỉ mục hiện tại cần được nâng cấp. Sau đó di chuyển dữ liệu từ chỉ mục cũ sang chỉ mục mới.**

xe cổ

xe cổ

biệt danh xe cổ điển

biệt danh xe cổ điển

xe\_cổ\_trang\_mới

**Trỏ bí danh đến chỉ mục mới và xóa chỉ mục cũ. Tất cả các truy vấn được viết theo bí danh vẫn hợp lệ và sẽ lấy dữ liệu từ chỉ mục mới.**

xe\_cổ\_trang\_mới

**Hình 6.4 Đạt được sự di chuyển với thời gian chết bằng không**

**206 CPHẦN6*Hoạt động lập chỉ mục***

Các truy vấn được thực hiện trên alias hiện sẽ lấy dữ liệu từ chỉ mục mới mà không cần ứng dụng bị trả lại. Do đó, chúng tôi đã đạt được thời gian chết bằng không.

**Tôi****CÁC HOẠT ĐỘNG BIASING ĐA PHƯƠNG THỨC SỬ DỤNG\_BIỆT DANHGiao diện lập trình ứng dụng (API)**

Ngoài việc làm việc với các bí danh bằng API \_alias, chúng ta có thể sử dụng một API khác để làm việc trên nhiều hành động đặt bí danh: API \_aliases. API này kết hợp một số hành động, chẳng hạn như thêm và xóa các bí danh và xóa các chỉ mục. Trong khi API \_alias được sử dụng để tạo bí danh, API \_aliases giúp tạo nhiều hành động trên các chỉ mục liên quan đến việc đặt bí danh.

Mã trong danh sách 6.19 thực hiện hai hoạt động đặt bí danh riêng biệt trên hai chỉ mục: nó xóa một bí danh trỏ đến một chỉ mục cũ và trỏ lại (thông qua hành động thêm) cùng một bí danh đó đến một chỉ mục mới (xem phần trước để hiểu lý do cần thực hiện thao tác này).

**Liệt kê 6.19 Thực hiện nhiều hoạt động bí danh**

POST \_bí danh

{

"hành động": [

{

"di dời": {

**Sử dụng API \_aliases để thực hiện nhiều hành động**

**Liệt kê các hành động riêng lẻ**



**Xóa bỏ bí danh đó**

"chỉ mục": "xe\_cổ\_bản",

"bí danh": "bí danh xe cổ điển"

}

},

{ **Thêmmột bí danh mà**

**trỏ đến một chỉ mục cũ**

"thêm vào": {

**trỏ đến một chỉ mục mới**

"index": "vintage\_cars\_new", "alias": "vintage\_cars\_alias"

}

}

]

}

Chúng tôi xóa một bí danh, vintage\_cars\_alias, ban đầu được tạo cho

xe cổchỉ mục. Sau đó chúng tôi gán lại nó choxe\_cổ\_trang\_mớimục lục.

Với API \_aliases, chúng ta cũng có thể xóa alias trỏ đến index hiện tại và gán nó cho index mới khi index mới đã sẵn sàng sau khi di chuyển dữ liệu. Và chúng ta có thể tạo alias cho nhiều index bằng cùng một API \_aliases bằng cách sử dụngtham số chỉ mục để thiết lập danh sách chỉ mục. Danh sách sau đây cho biết cách thực hiện điều đó.

**Liệt kê 6.20 Tạo một bí danh trỏ đến nhiều chỉ mục**

POST \_bí danh

{

"hành động": [

{

"thêm vào": {

* 1. ***Đọcchỉ số* 207**

"chỉ số": ["xe\_cổ\_đại","xe\_điện","xe\_hiếm","xe\_sang\_trọng"], "bí danh": "bí danh xe\_cao\_cấp"

}

}

]

}

Đây,hành độngtạo ra một bí danh được gọi làbí danh xe hơi cao cấpđiều đó chỉ ra bốn chỉ số xe hơi (xe cổ,xe điện,xe hiếm, Vàxe\_sang\_trọng). Bây giờ chúng ta đã thành thạo nghệ thuật tạo và đặt bí danh cho chỉ mục, hãy cùng xem cách đọc chúng.

#### Đọc chỉ số

Cho đến nay, các chỉ mục chúng ta đã xem xét là các chỉ mục công khai—thường được tạo bởi người dùng hoặc ứng dụng để lưu trữ dữ liệu. Chúng ta sẽ thảo luận về các chỉ mục công khai trong phần tiếp theo và sau đó xem xét một loại chỉ mục khác: chỉ mục ẩn.

* + 1. Đọc chỉ mục công khai

Chúng ta có thể lấy thông tin chi tiết của một chỉ mục bằng cách chỉ cần phát lệnh GET (như GET cars), như chúng ta đã thấy. Phản hồi cung cấp các ánh xạ, thiết lập và bí danh dưới dạng các đối tượng JSON. Phản hồi cũng có thể trả về thông tin chi tiết của nhiều chỉ mục. Giả sử chúng ta muốn trả về thông tin chi tiết của ba chỉ mục: cars1, cars2 và cars3. Danh sách sau đây cho thấy cách thực hiện.

**Liệt kê 6.21 Nhận cấu hình chỉ mục cho nhiều chỉ mục**

NHẬN xe1,xe2,xe3

**Lấy thông tin chi tiết của ba chỉ mục**

**GHI CHÚ**Kibana phàn nàn nếu URL nhiều chỉ mục có khoảng cách giữachỉ mục. Ví dụ, GET cars1, cars2 sẽ không thành công vì có một khoảng trắng sau dấu phẩy. Hãy đảm bảo nhiều chỉ mục được phân tách bằng dấu phẩy.

Lệnh này trả về thông tin có liên quan cho cả ba chỉ mục, nhưng việc cung cấp danh sách dài các chỉ mục được phân tách bằng dấu phẩy không nhất thiết sẽ phù hợp với các nhà phát triển. Thay vào đó, chúng ta có thể sử dụng ký tự đại diện nếu chỉ mục của chúng ta có một mẫu. Ví dụ, mã trong danh sách sau đây sẽ lấy tất cả các chỉ mục bắt đầu bằng các chữ cái ca.

**Liệt kê 6.22 Nhận nhiều cấu hình chỉ mục với ký tự đại diện**

NHẬN ca\*

**Trả về tất cả các chỉ mục có tiền tố là ca**

Sử dụng cùng nguyên tắc với các chỉ mục phân cách bằng dấu phẩy và ký tự đại diện, chúng ta có thể lấy cấu hình cho các chỉ mục cụ thể. Ví dụ, danh sách sau đây lấy tất cả các chỉ mục có tiền tố là mov và stu (viết tắt của movies và students).

**208 CPHẦN6*Hoạt động lập chỉ mục***

**Liệt kê 6.23 Lấy cấu hình cho các chỉ mục cụ thể**

NHẬN mov\*,stu\*

Mặc dù tất cả các lệnh GET này đều lấy các bí danh, ánh xạ và cài đặt của các chỉ mục đã chỉ định, vẫn còn một cách khác để trả về thông tin đó. Giả sử chúng ta muốn lấy một cấu hình riêng lẻ cho một chỉ mục cụ thể. Chúng ta có thể sử dụng các API có liên quan để thực hiện việc đó.

**Liệt kê 6.24 Nhận cấu hình riêng lẻ cho một chỉ mục**

NHẬN xe/\_cài đặt NHẬN xe/\_bản đồ NHẬN xe/\_bí danh

**Nhận được các ánh xạtừ điểm cuối \_mapping**

**Lấy các bí danh từ điểm cuối \_alias**

**Nhận các thiết lập từ**

**\_cài đặtđiểm cuối**

Các lệnh GET này trả về các cấu hình được chỉ định cho một chỉ mục công khai. Tiếp theo, chúng ta hãy xem xét việc truy xuất thông tin này cho một chỉ mục ẩn.

**GHI CHÚ**Để xác định xem chỉ mục có tồn tại trong cụm hay không, chúng ta có thể sử dụng lệnhĐẦU <tên\_chỉ\_mục>. Ví dụ,Xe ô tô HEADtrả về mã lỗi200—Đượcnếu chỉ số tồn tại và lỗi404—Không tìm thấynếu chỉ mục không tồn tại.

* + 1. Đọc các chỉ mục ẩn

Như đã đề cập, có hai loại chỉ mục: chỉ mục bình thường (public) mà chúng ta đã làm việc cho đến nay và chỉ mục ẩn. Tương tự như các thư mục ẩn trong hệ thống tệp máy tính, được thêm tiền tố là dấu chấm (.etc, .git, .bash, v.v.), chỉ mục ẩn thường được dành riêng cho quản lý hệ thống. Chúng được sử dụng cho Kibana, tình trạng ứng dụng, v.v. Ví dụ, chúng ta có thể tạo chỉ mục ẩn bằng cách thực hiện lệnh PUT

.old\_cars (lưu ý dấu chấm ở phía trước tên chỉ mục).

**GHI CHÚ**Mặc dù chúng ta có thể sử dụng các chỉ mục ẩn mô phỏng các chỉ mục công khai để sử dụng cho mục đích hoạt động (vì Elasticsearch không có kiểm tra và cân bằng trong phiên bản 8 trở về trước), nhưng điều này sẽ thay đổi trong các phiên bản tương lai. Tất cả các chỉ mục ẩn sẽ được dành riêng cho công việc liên quan đến hệ thống. Với sự thay đổi này, lời khuyên của tôi là hãy lưu ý khi tạo các chỉ mục ẩn cho dữ liệu liên quan đến doanh nghiệp.

Lệnh gọi GET \_all hoặc GET \* sẽ lấy tất cả các chỉ mục, bao gồm cả các chỉ mục ẩn. Để kiểm tra-Ví dụ, hình 6.5 hiển thị kết quả của lệnh GET \_all: lệnh này trả về toàn bộ danh sách chỉ mục, bao gồm cả các chỉ mục ẩn (chỉ mục có dấu chấm trước tên).

Bây giờ chúng ta đã biết cách tạo và đọc các chỉ mục, nếu cần, chúng ta có thể thực hiện các thao tác xóa trên chúng. Điều này sẽ được thảo luận chi tiết trong phần tiếp theo.

* 1. ***Xóa bỏchỉ số* 209**

**Các chỉ mục có tiền tố là dấu chấm là các chỉ mục ẩn.**



NHẬN \_tất cả

**Lệnh để lấy toàn bộ danh sách chỉ mục**

**Chỉ mục bình thường (liên quan đến doanh nghiệp hoặc ứng dụng)**

**Hình 6.5 Lấy danh sách tất cả các chỉ mục công khai và ẩn**

#### Xóa chỉ mục

Xóa một chỉ mục hiện có rất đơn giản: hành động DELETE trên chỉ mục (như DELETE <index\_name>) sẽ xóa chỉ mục vĩnh viễn. Ví dụ, phát lệnh DELETE cars sẽ xóa chỉ mục cars khi lệnh được phát, nghĩa là tất cả các tài liệu trong chỉ mục đó sẽ biến mất—mãi mãi—bao gồm mọi thiết lập, lược đồ ánh xạ và bí danh!

**DXÓA NHIỀU CHỈ MỤC**

Chúng ta cũng có thể xóa nhiều chỉ mục. Thêm danh sách các chỉ mục được phân tách bằng dấu phẩy để xóa chúng cùng lúc.

**Liệt kê 6.25 Xóa nhiều chỉ mục**

XÓA xe,phim,đặt hàng

Chúng ta cũng có thể xóa các chỉ mục bằng cách sử dụng mẫu ký tự đại diện:XÓA BỎ \*. Tuy nhiên, nếu chúng ta cũng muốn xóa các chỉ mục ẩn, chúng ta phải sử dụng\_tất cảđiểm cuối:XÓA \_tất cả. Lưu ý rằng chúng ta phải thiết lậphành động.phá hủy\_yêu\_cầu\_têntài sản đểSAIkhi cố gắng xóa các chỉ mục bằng ký tự đại diện hoặc\_tất cả:

ĐẶT \_cluster/cài đặt

{

"tạm thời": { "action.destructive\_requires\_name":false

}

}

**210 CPHẦN6*Hoạt động lập chỉ mục***

Cáchành động.phá hủy\_yêu\_cầu\_tênthuộc tính được thiết lập thànhĐÚNG VẬYtheo mặc định, do đó bạn có thể nhận được lỗi "Biểu thức ký tự đại diện hoặc tất cả các chỉ mục không được phép" khi cố gắng xóa các chỉ mục bằng ký tự đại diện hoặc\_tất cả.

**CẢNH BÁO**Việc xóa chỉ mục một cách vô tình có thể dẫn đến mất dữ liệu vĩnh viễn.Khi làm việc với API DELETE, bạn nên thận trọng vì việc vô tình gọi lệnh có thể làm mất ổn định hệ thống.

**D****CHỈ XÓA BÍ DANH**

Ngoài việc xóa toàn bộ chỉ mục, xóa nội bộ các ánh xạ, cài đặt, bí danh và dữ liệu, còn có một cơ chế để chỉ xóa bí danh. Chúng tôi sử dụng API \_alias cho mục đích này.

**Liệt kê 6.26 Xóa bí danhrõ ràng**

XÓA cars/\_alias/cars\_alias

**Xóa cars\_alias**

Xóa một chỉ mục là một tác vụ phá hoại vì dữ liệu bị xóa vĩnh viễn. Không cần phải nói, trước khi thực hiện thao tác này, hãy chắc chắn rằng bạn thực sự muốn xóa chỉ mục và tất cả các cấu hình và dữ liệu của nó. Tiếp theo, hãy xem một số thao tác ít phá hoại hơn: đóng và mở chỉ mục.

#### Chỉ số đóng và mở

Tùy thuộc vào trường hợp sử dụng, chúng ta có thể đóng hoặc mở một chỉ mục—chỉ mục sẽ được giữ lại để lập chỉ mục hoặc tìm kiếm thêm khi nó bị đóng. Hãy xem các tùy chọn để đóng một chỉ mục.

* + 1. Chỉ số đóng

*Đóng cửa*một chỉ mục có nghĩa chính xác là: nó đã đóng cửa kinh doanh và mọi hoạt động trên đó sẽ chấm dứt. Sẽ không có việc lập chỉ mục tài liệu hoặc tìm kiếm và truy vấn phân tích.

**GHI CHÚ**Vì các chỉ mục đóng không khả dụng cho các hoạt động kinh doanh, bạn phải cẩn thận trước khi đóng một chỉ mục. Làm như vậy có thể phá vỡ hệ thống nếu các chỉ mục đóng nhưng được tham chiếu trong mã của bạn. Đây là một lý do chính đáng để phụ thuộc vào các bí danh thay vì các chỉ mục thực!

API chỉ mục đóng (\_close) đóng chỉ mục. Cú pháp là POST <index\_name>/\_close. Ví dụ, danh sách sau đây đóng chỉ mục cars (cho đến khi có thông báo mới) và do đó bất kỳ thao tác nào trên đó đều dẫn đến lỗi.

**Liệt kê 6.27 Đóngxe ô tôchỉ số vô thời hạn**

POST xe ô tô/\_đóng

Mã này đóng chỉ mục để kinh doanh. Không được phép thực hiện thêm thao tác đọc/ghi nào nữa.

* 1. ***Đóng và mởchỉ số* 211**

**CMẤT TẤT CẢ HOẶC NHIỀU CHỈ SỐ**

Chúng ta có thể đóng nhiều chỉ mục bằng cách sử dụng các chỉ mục được phân tách bằng dấu phẩy (bao gồm cả ký tự đại diện).

**Liệt kê 6.28 Đóng nhiềuchỉ số**

POST cars1,\*mov\*,students\*/\_close

Cuối cùng, nếu chúng ta muốn dừng tất cả các hoạt động trực tiếp trên các chỉ mục trong một cụm, chúng ta có thể đưa ra lệnh gọi API chỉ mục đóng (\_close) với \_all hoặc \*, như được hiển thị trong danh sách tiếp theo (hãy đảm bảoaction.destructive\_requires\_name propertyliases được đặt thành false để tránh lỗi "không được phép sử dụng biểu thức ký tự đại diện").

**Liệt kê 6.29 Đóng tất cảchỉ số**

POST \*/\_close

**Đóng tất cả các chỉ mục trong cụm**

**MỘT****VÔ HIỆU LÀM MẤT ỔN ĐỊNH HỆ THỐNG**

Như bạn có thể tưởng tượng, việc đóng (hoặc mở) tất cả các chỉ mục có thể làm mất ổn định hệ thống. Đây là một trong những khả năng quản trị siêu cấp, nếu thực hiện mà không cân nhắc trước, có thể dẫn đến hậu quả thảm khốc, bao gồm cả việc hạ gục hệ thống hoặc khiến hệ thống không thể sửa chữa được. Đóng chỉ mục sẽ chặn các hoạt động đọc/ghi và do đó giảm thiểu chi phí duy trì các phân đoạn cụm. Các tài nguyên sẽ bị xóa và bộ nhớ sẽ được khôi phục trên các chỉ mục đã đóng.

Chúng ta có thể vô hiệu hóa khả năng đóng chỉ mục không? Có, chúng ta có thể vô hiệu hóa tính năng đóng nếu chúng ta không bao giờ muốn đóng bất kỳ chỉ mục nào. Bằng cách vô hiệu hóa chức năng này, chúng ta cho phép các chỉ mục hoạt động mãi mãi (trừ khi chúng ta xóa chúng). Để thực hiện việc này, chúng ta đặt cụm

.chỉ số.đóng.cho phéptài sản đểSAI(nó làĐÚNG VẬYtheo mặc định) trên cụm bằng cách sử dụng cài đặt cấu hình.

**Liệt kê 6.30 Vô hiệu hóa các chỉ mục đóngtính năng**

ĐẶT \_cluster/cài đặt

{

"dai dẳng": { "cluster.indices.close.enable":false

}

}

* + 1. Mở chỉ mục

Việc mở một chỉ mục sẽ khởi động lại các phân đoạn; khi chúng đã sẵn sàng, chúng sẽ được mở để lập chỉ mục và tìm kiếm. Chúng ta có thể mở một chỉ mục đã đóng chỉ bằng cách gọi API \_open.

**Liệt kê 6.31 Đưa chỉ mục trở lại hoạt động**

POST xe/\_mở

**212 CPHẦN6*Hoạt động lập chỉ mục***

Sau khi lệnh được thực thi thành công, chỉ mục cars sẽ có sẵn ngay lập tức. Tương tự như API \_close, API \_open có thể được gọi trên nhiều chỉ mục, bao gồm cả việc chỉ định chỉ mục bằng ký tự đại diện.

Cho đến nay, chúng tôi đã làm việc với một số hoạt động chỉ mục và tạo các chỉ mục với các ánh xạ và thiết lập tùy chỉnh, mặc dù riêng lẻ. Mặc dù phương pháp này hiệu quả trong quá trình phát triển, nhưng nó không lý tưởng trong môi trường sản xuất—có thể rất cồng kềnh khi tạo một tập hợp các chỉ mục nhiều lần. Ngoài ra, chúng tôi có thể không muốn các kỹ sư tạo các chỉ mục với số lượng phân đoạn hoặc bản sao không mong muốn, điều này có thể làm mất ổn định cụm của chúng tôi. Đây là nơi chúng tôi có thể sử dụng các mẫu chỉ mục.

Kiểm soát chặt chẽ hơn và các tiêu chuẩn kinh doanh được mong đợi như một phần của chiến lược lập chỉ mục tổng thể, do đó Elasticsearch cung cấp các mẫu chỉ mục để hỗ trợ phát triển các chỉ mục với chiến lược tổ chức trong đầu. Chúng ta có thể sử dụng tính năng này để áp dụng các cấu hình ở quy mô lớn, như đã thảo luận trong phần tiếp theo.

#### Mẫu chỉ mục

Sao chép cùng một thiết lập trên nhiều chỉ mục khác nhau, đặc biệt là từng cái một, rất tẻ nhạt và đôi khi dễ xảy ra lỗi. Nếu chúng ta định nghĩa một mẫu lập chỉ mục với một lược đồ ở phía trước, một chỉ mục mới sẽ được đúc ngầm từ lược đồ này nếu tên chỉ mục khớp với mẫu. Do đó, bất kỳ chỉ mục mới nào cũng sẽ tuân theo cùng một thiết lập và đồng nhất trên toàn tổ chức, và DevOps sẽ không cần phải liên tục ủng hộ các thiết lập tối ưu với từng nhóm trong tổ chức.

Một trường hợp sử dụng cho mẫu lập chỉ mục là tạo một tập hợp các mẫu dựa trên môi trường. Ví dụ, các chỉ mục cho môi trường phát triển có thể có 3 phân đoạn và 2 bản sao, trong khi các chỉ mục cho môi trường sản xuất có 10 phân đoạn với 5 bản sao cho mỗi phân đoạn.

Với index template, chúng ta có thể tạo một template với các mẫu và cấu hình được xác định trước. Chúng ta có thể có một tập hợp các ánh xạ, thiết lập và bí danh được đóng gói trong template này, cùng với một tên index. Sau đó, khi chúng ta tạo một index mới, nếu tên index khớp với tên pattern, template sẽ được áp dụng. Ngoài ra, chúng ta có thể tạo một template dựa trên một mẫu glob (lệnh toàn cục), chẳng hạn như ký tự đại diện, tiền tố, v.v.

**GHI CHÚ**Mẫu glob thường được sử dụng trong phần mềm máy tính để biểu diễn phần mở rộng tên tệp (\*.txt, \*.cmd, \*.bat, v.v.).

Elasticsearch đã nâng cấp chức năng mẫu của mình bắt đầu từ phiên bản 7.8 và phiên bản mới hơn này trừu tượng hơn và có thể tái sử dụng nhiều hơn. (Nếu bạn quan tâm đến các mẫu chỉ mục từ các phiên bản trước, hãy xem tài liệu chính thức để biết chi tiết.) Các mẫu chỉ mục là được phân loại vào trong hai Thể loại: *có thể ghép lại* chỉ số mẫu (hoặc đơn giản *chỉ số mẫu*)và các mẫu thành phần. Như tên gọi cho thấy, các mẫu chỉ mục có thể cấu thành được tạo thành từ không hoặc nhiều mẫu thành phần. Một mẫu chỉ mục cũng có thể tồn tại độc lập mà không cần liên kết với bất kỳ mẫu thành phần nào. Các mẫu chỉ mục như vậy có thể chứa tất cả các tính năng mẫu cần thiết (ánh xạ, cài đặt và bí danh). Chúng được sử dụng như các mẫu độc lập khi tạo chỉ mục có mẫu.

***6.6 Mục lụcmẫu* 213**

Mặc dù bản thân một mẫu thành phần là một mẫu, nhưng nó không hữu ích lắm nếu nó không được liên kết với một mẫu chỉ mục. Tuy nhiên, nó có thể được liên kết với nhiều mẫu chỉ mục. Thông thường, chúng tôi phát triển một mẫu thành phần (ví dụ, chỉ định codec cho một môi trường phát triển) và đính kèm nó vào nhiều chỉ mục khác nhau thông qua các mẫu chỉ mục có thể cấu hình cùng một lúc. Điều này được thể hiện trong hình 6.6.

**Gồm ba mẫu thành phần**

**Các mẫu thành phần được chia sẻ giữa các mẫu chỉ mục có thể cấu hình**

**Gồm hai mẫu thành phần được chia sẻ với các mẫu lập chỉ mục khác**



Mẫu lập chỉ mục C

Mẫu thành phần 1

Mẫu lập chỉ mục A

Mẫu thành phần 2

Mẫu lập chỉ mục B



Mẫu lập chỉ mục D

Mẫu thành phần 4

Mẫu thành phần 3

**Không bao gồm bất kỳ mẫu thành phần nào**

**Gồm ba mẫu thành phần được chia sẻ với các mẫu lập chỉ mục khác**

**Hình 6.6 Các mẫu có thể cấu thành (chỉ mục) được tạo thành từ các mẫu thành phần.**



Trong hình, các mẫu lập chỉ mục A, C và D chia sẻ các mẫu thành phần với nhau (ví dụ, mẫu thành phần 2 được cả ba mẫu lập chỉ mục sử dụng). Mẫu lập chỉ mục B là mẫu độc lập không có mẫu thành phần nào.

Chúng ta có thể tạo một mẫu chỉ mục mà không có mẫu thành phần nào. Điều này đưa chúng ta đến một số quy tắc nhất định khi tạo mẫu:

* Một chỉ mục được tạo bằng cấu hình rõ ràng sẽ được ưu tiên hơn các cấu hình được xác định trong các mẫu chỉ mục hoặc thành phần. Điều này có nghĩa là nếu bạn tạo một chỉ mục với các thiết lập rõ ràng, đừng mong đợi chúng bị ghi đè bởi các mẫu.
* Các mẫu cũ có mức độ ưu tiên thấp hơn các mẫu có thể cấu hình.
  + 1. Tạo mẫu có thể cấu hình (chỉ mục)

Để tạo một chỉ mụcmẫu, chúng tôi sử dụng điểm cuối \_index\_template, cung cấp tất cả các ánh xạ, thiết lập và bí danh cần thiết dưới dạng mẫu chỉ mục trong mẫu này. Giả sử yêu cầu của chúng tôi là tạo mẫu cho ô tô, được biểu diễn bằng mẫu có ký tự đại diện: \*ô tô\*, như được hiển thị trong danh sách 6.32 Mẫu này có một số thuộc tính vàcài đặt, chẳng hạn như thuộc tính created\_by và created\_at, cũng như các phân đoạn và số bản sao. Bất kỳ chỉ mục nào khớp với mẫu này trong quá trình tạo của nó đều kế thừa

**214 CPHẦN6*Hoạt động lập chỉ mục***

cấu hình được xác định trong mẫu. Ví dụ,xe\_mới,xe\_thể\_thao, Vàxe cổchỉ số cóđược tạo bởiVàđã tạo\_tạithuộc tính, cũng như các phân đoạn và bản sao 1 và 3, tương ứng, nếu chúng được định nghĩa trong mẫu.

**Liệt kê 6.32 Tạo chỉ mụcbản mẫu**

ĐẶT \_index\_template/cars\_template

{

"index\_patterns": ["\*xe hơi\*"], "ưu tiên": 1,

"bản mẫu": {

"ánh xạ": { "thuộc tính":{

"created\_at":{ "type":"ngày"

},

"created\_by":{ "type":"văn bản"

}

}

}

}

}

Khi chúng ta thực hiện lệnh này (bạn có thể nhận được cảnh báo lỗi thời—bỏ qua ngay bây giờ), mẫu được tạo bằng mẫu chỉ mục \*cars\*. Khi chúng ta tạo chỉ mục mới, nếu tên của chỉ mục mới (ví dụ: vintage\_cars) khớp với mẫu (\*cars\*), chỉ mục được tạo bằng cấu hình được xác định trong mẫu. Khi chỉ mục khớp với mẫu đã cho, các cấu hình mẫu được áp dụng tự động.



**Mẫu ưu tiên**

Mẫu chỉ mục có một mức độ ưu tiên: một số dương được xác định khi tạo mẫu.plate. Mức độ ưu tiên càng cao thì mức độ ưu tiên càng cao. Mức độ ưu tiên hữu ích khi chúng ta có các thiết lập tương tự hoặc giống nhau trong hai mẫu khác nhau. Nếu một chỉ mục khớp với nhiều hơn một mẫu chỉ mục, mẫu có mức độ ưu tiên cao hơn sẽ được sử dụng. Ví dụ:xe\_mẫu\_tháng\_21ghi đèxe\_mẫu\_mar21trong đoạn mã sau (mặc dù tên gợi ý ngược lại!):

ĐĂNG \_index\_template/cars\_template\_mar21

{

"index\_patterns": ["\*xe hơi\*"],

"ưu tiên": 20,

"bản mẫu": { ... }

**Ưu tiên thấp hơn**

**mẫu chỉ mục**

}

ĐĂNG \_index\_template/cars\_template\_feb21

{

"index\_patterns": ["\*xe hơi\*"],

* 1. ***Mục lụcmẫu* 215**



"ưu tiên": 30,

"bản mẫu": { ... }

**Phù hợp với mẫu, nhưng**

**với mức độ ưu tiên cao hơn**

}

Khi nhiều mẫu khớp với các chỉ mục đang được tạo, Elasticsearch áp dụng tất cả các thiết lập từ tất cả các mẫu khớp, nhưng mẫu có mức độ ưu tiên cao hơn sẽ được đưa lên đầu. Trong ví dụ trước, nếuxe\_mẫu\_mar21có một codec xác địnhnén tốt nhất, nó bị ghi đè bởi một codec có giá trị mặc định (được xác định trongxe\_mẫu\_tháng\_21bản mẫu).Bởi vì mẫu sau có mức độ ưu tiên cao hơn.

Bây giờ chúng ta đã biết nhiều hơn về mẫu chỉ mục, hãy cùng xem xét các mẫu thành phần có thể tái sử dụng.

* + 1. Tạo mẫu thành phần

Nếu bạn có nền tảng DevOps, tại một thời điểm nào đó, bạn có thể có yêu cầu tạo chỉ mục với cấu hình cài đặt sẵn cho từng môi trường. Thay vì áp dụng thủ công từng cấu hình, bạn có thể tạo mẫu thành phần cho từng môi trường.

Một mẫu thành phần không gì khác ngoài một khối cấu hình có thể tái sử dụng mà chúng ta có thể sử dụng để tạo thêm nhiều mẫu chỉ mục. Tuy nhiên, các mẫu thành phần không có giá trị trừ khi chúng được liên kết với các mẫu chỉ mục. Chúng được hiển thị thông qua \_component\_điểm cuối mẫu. Chúng ta hãy xem điều này phù hợp với nhau như thế nào.

Giả sử chúng ta cần tạo một mẫu cho môi trường phát triển có ba phân đoạn chính với ba bản sao cho mỗi phân đoạn. Bước đầu tiên là khai báo và thực thi một mẫu thành phần với cấu hình này.

**Liệt kê 6.33 Phát triển mẫu thành phần**

POST \_component\_template/dev\_settings\_component\_template

{

"bản mẫu":{

"cài đặt":{ "số\_mảnh\_shards":3, "số\_bản\_sao\_lại":3

}

}

}

Trong danh sách này, chúng tôi sử dụng điểm cuối \_component\_template để tạo một mẫu. Nội dung của yêu cầu lưu giữ thông tin mẫu trong một đối tượng mẫu. Sau khi thực hiện, dev\_settings\_component\_template sẽ có sẵn để sử dụng ở nơi khác trong các mẫu chỉ mục. Lưu ý rằng mẫu này không xác định mẫu chỉ mục; nó chỉ đơn giản là một khối mã cấu hình một số thuộc tính cho chúng tôi.

Tương tự như vậy, chúng ta hãy tạo một mẫu khác. Lần này, chúng ta hãy định nghĩa một lược đồ ánh xạ.

**216 CPHẦN6*Hoạt động lập chỉ mục***

**Danh sách 6.34 Mẫu thành phần có lược đồ ánh xạ**

POST \_component\_template/dev\_mapping\_component\_template

{

"bản mẫu": {

"ánh xạ": { "thuộc tính": {

"created\_by": { "type": "văn bản"

},

"phiên bản": {

"kiểu": "float"

}

}

}

}

}

Cácmẫu thành phần lập bản đồ devbao gồm một lược đồ ánh xạ được xác định trước với hai thuộc tính,được tạo bởiVàphiên bản.

Bây giờ chúng ta đã có hai mẫu thành phần, bước tiếp theo là đưa chúng vào sử dụng.

Chúng ta có thể thực hiện điều này bằng cách cho phép một mẫu chỉ mục, chẳng hạn như ô tô, sử dụng chúng.

**Danh sách 6.35 Mẫu chỉ mục bao gồm các mẫu thành phần**

ĐĂNG \_index\_template/composed\_cars\_template

{

"index\_patterns": ["\*xe hơi\*"], "ưu tiên": 200,

"composed\_of": ["dev\_settings\_component\_template",

"dev\_mapping\_component\_template"]

}

Thẻ combined\_of được tô sáng là tập hợp tất cả các mẫu thành phần mà chúng ta muốn áp dụng. Trong trường hợp này, chúng ta chọn các mẫu thành phần cài đặt và ánh xạ.

Sau khi tập lệnh được thực thi, nếu chúng ta tạo một chỉ mục có cars trong tên chỉ mục (vintage\_cars, my\_cars\_old, cars\_sold\_in\_feb, v.v.), chỉ mục được tạo bằng cấu hình bắt nguồn từ cả hai mẫu thành phần. Ví dụ, để tạo một mẫu tương tự trong môi trường sản xuất, chúng ta có thể tạo một mẫu có thể cấu hình bằngphiên bản prod\_\* của các mẫu thành phần.

Cho đến nay, chúng tôi đã làm việc với các hoạt động CRUD trên các chỉ mục và khởi tạo các chỉ mục đó bằng các mẫu. Tuy nhiên, chúng tôi không có khả năng hiển thị hiệu suất chỉ mục. Như phần tiếp theo sẽ thảo luận, Elasticsearch cung cấp số liệu thống kê về dữ liệu đã được lập chỉ mục, xóa và truy vấn.

#### Theo dõi và quản lý chỉ số

Elasticsearch cung cấp số liệu thống kê chi tiết về dữ liệu được đưa vào chỉ mục cũng như dữ liệu được rút ra. Nó cung cấp API để tạo báo cáo như số lượng tài liệu

***6.7 Giám sát và quản lýchỉ số* 217**

ments mà index lưu giữ, tài liệu đã xóa, hợp nhất và xóa số liệu thống kê, v.v. Trong vài phần tiếp theo, chúng ta sẽ xem xét các API để lấy các số liệu thống kê này.

* + 1. Thống kê chỉ số

Mỗi chỉ mục tạo ra các số liệu thống kê như tổng số tài liệu mà nó có, số lượng tài liệu đã xóa, bộ nhớ của phân đoạn, dữ liệu yêu cầu tìm kiếm và lấy, v.v. API \_stats giúp chúng ta truy xuất số liệu thống kê chỉ mục cho cả phân đoạn chính và phân đoạn bản sao.

Danh sách sau đây cho thấy cơ chế để lấy số liệu thống kê cho chỉ mục cars bằng cách gọi điểm cuối \_stats. Hình 6.7 cho thấy số liệu thống kê được trả về bởi lệnh gọi này.

**Liệt kê 6.36 Lấy số liệu thống kê của mộtchỉ số**

NHẬN xe/\_thống kê

NHẬN xe/\_thống kê

**Tổng số mảnh vỡ cho chỉ mục này**

**Các\_tất cảkhối cung cấp số liệu thống kê kết hợp của tất cả các chỉ mục trên cả phân đoạn chính và phân đoạn sao chép.**



**Cácchỉ sốkhối cung cấp số liệu thống kê của một chỉ mục cụ thể.**

**Hình 6.7 Thống kê cho mộtxe ô tôchỉ số**

Phản hồi bao gồm một thuộc tính tổng, là tổng số phân đoạn (chính và bản sao) được liên kết với chỉ mục này. Vì chúng ta chỉ có một phân đoạn chính,thuộc tính thành công trỏ đến số lượng phân mảnh này.

Câu trả lời bao gồm hai khối:

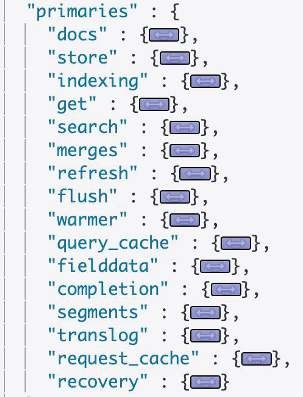
* Các\_tất cảkhối, với số liệu thống kê tổng hợp cho tất cả các chỉ mục được kết hợp
* Cácchỉ sốkhối, với số liệu thống kê cho từng chỉ mục (mỗi chỉ mục trong cụm đó)

Các khối này bao gồm hai nhóm thống kê: nhóm chínhbucket chứa các số liệu thống kê chỉ liên quan đến các mảnh chính, trong khi bucket tổng thể chỉ ra các số liệu thống kê cho cả các mảnh chính và bản sao. Elasticsearch trả về hơn một chục số liệu thống kê (xem

**218 CPHẦN6*Hoạt động lập chỉ mục***

hình 6.8) mà chúng ta có thể tìm thấy trong các nhóm phản hồi chính và tổng số. Bảng 6.1 mô tả một số thống kê này.

**Hơn một chục số liệu thống kê được trả về bởi\_thống kêgọi.**



**Các số liệu thống kê này có thông tin chi tiết về từng số liệu riêng lẻ.**

**Hình 6.8Nhiều số liệu thống kê cho một chỉ mục**

**Bảng 6.1 Thống kê chỉ mục được lấy bằng lệnh gọi đến\_thống kêđiểm cuối (danh sách được rút gọn để ngắn gọn)**

|  |  |
| --- | --- |
| **Thống kê** | **Sự miêu tả** |
| tài liệu | Số lượng tài liệu trong chỉ mục và số lượng tài liệu đã xóa |
| cửa hàng | Kích thước của chỉ mục (tính bằng byte) |
| lấy | Số lượng các hoạt động GET trên chỉ mục |
| tìm kiếm | Các thao tác tìm kiếm bao gồm truy vấn, cuộn và gợi ý thời gian. |
| làm cho khỏe lại | Số lượng hoạt động làm mới |

Có nhiều số liệu thống kê hơn, chẳng hạn như lập chỉ mục, hợp nhất, hoàn thành, dữ liệu trường, phân đoạn và các số liệu khác, nhưng chúng tôi đã bỏ qua chúng ở đây do hạn chế về không gian. Xem[http://](http://mng.bz/Pzd5) [mng.bz/Pzd5](http://mng.bz/Pzd5)để biết danh sách đầy đủ các số liệu thống kê mà Elasticsearch đưa ra thông qua API thống kê chỉ mục.

* + 1. Nhiều chỉ số và số liệu thống kê

Cũng giống như chúng ta lấy dữ liệu thống kê trên một chỉ mục riêng lẻ, chúng ta cũng có thể lấy dữ liệu thống kê trên nhiều chỉ mục bằng cách cung cấp tên chỉ mục được phân tách bằng dấu phẩy. Danh sách sau đây hiển thị lệnh.

* 1. ***Giám sát và quản lýchỉ số* 219**

**Liệt kê 6.37 Lấy số liệu thống kê cho nhiều chỉ mục**

NHẬN cars1,cars2,cars3/\_stats

Chúng ta cũng có thể sử dụng ký tự đại diện trên các chỉ mục, như được hiển thị bên dưới.

**Liệt kê 6.38 Lấy số liệu thống kê bằng cách sử dụngký tự đại diện**

NHẬN xe\*/\_stats

Và đây là cách lấy số liệu thống kê cho tất cả các chỉ mục trong cụm (bao gồm cả các chỉ mục ẩn).

**Liệt kê 6.39 Lấy số liệu thống kê cho tất cả các chỉ mục trongcụm**

NHẬN \*/\_stats

Chúng ta có thể không cần tìm tất cả các số liệu thống kê mọi lúc, vì vậy chúng ta có thể, ví dụ, chỉ tìm một số số liệu thống kê nhất định về các phân đoạn như trường được lưu trữ và bộ nhớ thuật ngữ, kích thước tệp, số lượng tài liệu, v.v. Danh sách sau đây trả về số liệu thống kê theo phân đoạn.

**Liệt kê 6.40 Phân đoạnthống kê**

NHẬN xe/\_thống kê/phân khúc

Lệnh này trả về dữ liệu sau:

"phân đoạn" : {

"đếm" : 1,

"memory\_in\_bytes" : 1564,

"terms\_memory\_in\_bytes" : 736,

"stored\_fields\_memory\_in\_bytes" : 488,

"term\_vectors\_memory\_in\_bytes" : 0,

"norms\_memory\_in\_bytes" : 64,

"points\_memory\_in\_bytes" : 0,

"doc\_values\_memory\_in\_bytes" : 276,

"index\_writer\_memory\_in\_byte" : 0,

"version\_map\_memory\_in\_bytes" : 0,

"fixed\_bit\_set\_memory\_in\_bytes" : 0,

"max\_unsafe\_auto\_id\_timestamp" : -1, "file\_sizes" : { }

}

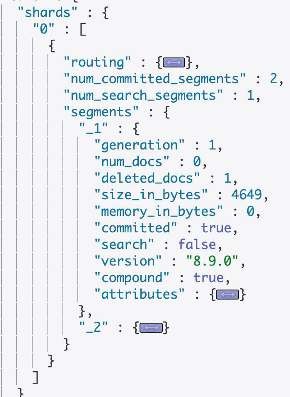
Khi chúng ta đang nói về chủ đề này, Elasticsearch cung cấp API phân đoạn chỉ mục để xem chi tiết cấp thấp hơn của các phân đoạn. Gọi mã GET cars/\_segments cung cấp chế độ xem chi tiết về các phân đoạn do Apache Lucene quản lý, chẳng hạn như danh sách các phân đoạn và số lượng tài liệu (bao gồm cả tài liệu đã xóa) mà phân đoạn nắm giữ, dung lượng đĩa, phân đoạn có thể tìm kiếm được không, v.v. (xem hình 6.9). Chúng ta cũng có thể lấy thông tin phân đoạn trên toàn bộ chỉ mục bằng cách gọi GET

\_phân đoạngọi.

**220 CPHẦN6*Hoạt động lập chỉ mục***

**Nhiều phân đoạn của Shard**

**Thông tin chi tiết của phân khúc**



**Hình 6.9 Thông tin chi tiết về mọi phân đoạn trong một phân đoạn**

Đôi khi chúng ta có thể cần quản lý một chỉ mục quá lớn để tiết kiệm không gian hoặc giảm cơ sở hạ tầng đằng sau một chỉ mục chủ yếu không hoạt động. Chúng ta có thể thực hiện các hoạt động như vậy bằng các thủ thuật nâng cao, được thảo luận chi tiết trong phần tiếp theo.

#### Các hoạt động nâng cao

Trước đó, chúng ta đã xem xét các hoạt động CRUD như tạo, đọc và xóa chỉ mục. Ngoài các hoạt động cơ bản như vậy, chúng ta có thể thực hiện các hoạt động nâng cao như chia tách chỉ mục để thêm nhiều phân đoạn hơn hoặc giảm quy mô chỉ mục bằng cách thu nhỏ hoặc chuyển đổi định kỳ (chẳng hạn như hàng ngày). Hãy cùng xem xét một số hoạt động nâng cao mà chúng ta có thể thực hiện trên các chỉ mục của mình.

* + 1. Chia tách một chỉ mục

Đôi khi các chỉ mục bị quá tải dữ liệu. Để tránh nguy cơ mất dữ liệu hoặc giảm thiểu phản hồi truy vấn tìm kiếm chậm hơn, chúng ta có thể phân phối lại dữ liệu vào các phân đoạn bổ sung. Việc thêm các phân đoạn vào chỉ mục sẽ tối ưu hóa bộ nhớ và phân phối tài liệu đồng đều.

Ví dụ, nếu một chỉ mục (cars) có 5 phân đoạn chính bị quá tải, chúng ta có thể chia chỉ mục thành một chỉ mục mới có nhiều phân đoạn chính hơn: ví dụ, 15. Việc mở rộng một chỉ mục từ kích thước nhỏ thành kích thước lớn hơn được gọi là chia tách chỉ mục. Chia tách không gì khác hơn là tạo một chỉ mục mới có nhiều phân đoạn hơn và sao chép dữ liệu từ chỉ mục cũ vào chỉ mục mới.

Elasticsearch cung cấp API \_split để chia tách một chỉ mục. Có một số quy tắc, chẳng hạn như số lượng phân đoạn mà một chỉ mục mới có thể được tạo ra, nhưng trước tiên hãy xem cách chúng ta có thể chia tách một chỉ mục.

***6.8 Nâng caohoạt động* 221**

Giả sử chỉ mục all\_cars của chúng ta được tạo bằng hai phân đoạn và vì dữ liệu đang tăng theo cấp số nhân nên chỉ mục hiện đã quá tải. Để giảm thiểu rủi ro truy vấn chậm và hiệu suất giảm, chúng ta muốn tạo một chỉ mục mới có nhiều không gian hơn. Để thực hiện việc này, chúng ta có thể chia chỉ mục thành một chỉ mục mới có nhiều không gian hơn và các phân đoạn chính bổ sung.

Trước khi chúng ta gọi thao tác chia tách trên chỉ mục all\_cars, chúng ta phải đảm bảo chỉ mục bị vô hiệu hóa cho mục đích lập chỉ mục—tức là chúng ta cần thay đổi chỉ mục thành chỉ đọc. Mã sau đây giúp ích bằng cách gọi API \_settings.

**Liệt kê 6.41 Đảm bảo chỉ mục chỉ đọc**

ĐẶT tất cả các xe/cài đặt

{

"cài đặt":{

**Sử dụng**

**\_cài đặt API**

"index.blocks.write":"đúng"

}

}

**Đóng chỉ mục cho các hoạt động ghi**

Bây giờ điều kiện tiên quyết để làm cho chỉ mục không hoạt động đã hoàn tất, chúng ta có thể chia nhỏ chỉ mục bằng cách gọi API \_split. API này mong đợi các chỉ mục nguồn và đích:

POST <source\_index>/\_tách ra/<chỉ mục mục tiêu>

Bây giờ, chúng ta hãy chia chỉ mục thành một chỉ mục mới (tất cả xe mới).

**Liệt kê 6.42 Chia táchtất cả các xechỉ số**

POST tất cả các xe/\_split/tất cả các xe mới

{

"cài đặt": { "index.number\_of\_shards": 12

}

**\_split mong đợi một chỉ mục mục tiêu.**

**Thiết lập số lượng mảnh vỡtrên chỉ số mới**

}

Yêu cầu này khởi động quá trình chia tách. Hoạt động chia tách là đồng bộ, nghĩa là yêu cầu của máy khách sẽ chờ phản hồi cho đến khi quá trình hoàn tất. Khi quá trình chia tách hoàn tất, chỉ mục mới (all\_cars\_new) sẽ có tất cả dữ liệu cũng như không gian bổ sung vì có thêm nhiều phân đoạn được thêm vào.

Như đã đề cập trước đó, hoạt động chia tách đi kèm với một số quy tắc và điều kiện nhất định. Chúng ta hãy xem xét một số quy tắc và điều kiện đó:

* *Chỉ mục mục tiêu không được tồn tại trước thao tác này.*Ngoài cấu hình mà chúng tôi cung cấp trong đối tượng yêu cầu khi phân tách (liệt kê 6.42), một bản sao chính xác của chỉ mục nguồn sẽ được chuyển đến chỉ mục đích.
* *Số lượng phân đoạn trong chỉ mục đích phải là bội số của số lượng phân đoạn trong chỉ mục nguồn.*Nếu chỉ mục nguồn có ba phân đoạn chính, chỉ mục đích có thể được xác định bằng số phân đoạn là bội số của ba (ba, sáu, chín, v.v.).

**222 CPHẦN6*Hoạt động lập chỉ mục***

* *Chỉ mục đích không bao giờ có thể có ít phân đoạn chính hơn chỉ mục nguồn.*

Hãy nhớ rằng việc chia nhỏ sẽ tạo ra nhiều không gian hơn cho chỉ mục.

* *Nút của chỉ mục mục tiêu phải có đủ không gian.*Đảm bảo các mảnh vỡ được phân bổvới không gian thích hợp.

Trong quá trình phân tách, tất cả các cấu hình (cài đặt, ánh xạ và bí danh) được sao chép từ chỉ mục nguồn vào chỉ mục đích mới được tạo. Sau đó, Elasticsearch di chuyển các liên kết cứng của phân đoạn nguồn đến chỉ mục đích. Cuối cùng, tất cả các tài liệu được băm lại, vì trang chủ của tài liệu đã thay đổi.

**Số chỉ mục của mục tiêu phải là bội số của số chỉ mục của nguồn**

Số lượng phân đoạn chính của chỉ mục mục tiêu phải là bội số của số phân đoạn chính.những mảnh vỡ của mary trong chỉ mục nguồn.Để xem điều gì xảy ra, hãy cung cấp một số không phải số nhiều để thiết lập lạiindex.số\_mảnh\_shards(ĐẾN14: một số không phải là bội số của số chính của nguồnsố mảnh vỡ là 3) như truy vấn sau đây hiển thị:

POST tất cả các xe/\_split/tất cả các xe mới

{

"cài đặt": { "index.number\_of\_shards": 14

}

}

Thật không may, truy vấn này đưa ra một ngoại lệ vì chúng tôi đã vi phạm quy tắc thứ hai trongdanh sách trước. Đây là ngoại lệ:

"lý do": "số lượng mảnh nguồn [3] phải là một hệ số của [14]"

Việc chia tách các chỉ mục cũng giúp thay đổi kích thước một cụm bằng cách thêm các chỉ mục chính vào số lượng ban đầu. Các cấu hình được sao chép qua các phân đoạn nguồn đến các phân đoạn đích. Vì vậy, ngoài việc thêm nhiều phân đoạn hơn, API chia tách không thể thay đổi bất kỳ cài đặt nào trên chỉ mục đích. Nếu chúng ta chỉ cần tăng số lượng phân đoạn để dữ liệu được phân bổ trên các chỉ mục mới tạo, thì chia tách là cách tốt nhất. Ngoài ra, hãy nhớ rằng các chỉ mục đích không được tồn tại trước khi gọi thao tác chia tách. Bây giờ chúng ta đã làm việc trên các chỉ mục chia tách, hãy xem xét một thao tác khác: thu nhỏ một chỉ mục.

* + 1. Thu hẹpmột chỉ số

Trong khi việc chia tách một chỉ mục mở rộng chỉ mục bằng cách thêm các phân đoạn bổ sung để có thêm không gian, thì việc thu hẹp lại ngược lại: nó làm giảm số lượng các phân đoạn. Việc thu hẹp giúp hợp nhất tất cả các tài liệu trải rộng trên nhiều phân đoạn khác nhau thành ít phân đoạn hơn. Làm như vậy có thể giúp ích trong các trường hợp sử dụng như sau:

* + - * Chúng tôi đã có một hoặc nhiều chỉ mục di chuyển trong thời gian nghỉ lễ, với dữ liệu trải rộng trên nhiều phân đoạn. Bây giờ khi thời gian nghỉ lễ đã kết thúc, chúng tôi muốn giảm số lượng phân đoạn.
      * Để tăng tốc độ đọc (thông lượng tìm kiếm), chúng tôi đã thêm các nút dữ liệu. Khi nhu cầu giảm xuống, không có lý do gì để giữ tất cả các nút đó hoạt động.

***6.8 Nâng caohoạt động* 223**

Giả sử chúng ta có một chỉ mục (all\_cars) được phân phối giữa 50 mảnh và muốn thay đổi kích thước của nó thành số lượng mảnh một chữ số: ví dụ, 5 mảnh. Tương tự như những gì chúng ta đã làm với một hoạt động chia tách, bước đầu tiên là đảm bảo rằng chỉ mục all\_cars của chúng ta chỉ đọc, vì vậy chúng ta đặt thuộc tính index.blocks.write thành true. Sau đó, chúng ta có thể điều chỉnh lại các mảnh thành một nút duy nhất. Mã trong danh sách sau đây hiển thị các hành động này là điều kiện tiên quyết trước khi thu nhỏ chỉ mục.

**Liệt kê 6.43 Điều kiện tiên quyết trước khi thu hẹp chỉ mục**

ĐẶT tất cả các xe/cài đặt

{

"cài đặt": { "index.blocks.write": đúng,

"index.routing.allocation.require.\_name": "node1"

}

}

Bây giờ chỉ mục nguồn đã được thiết lập để thu nhỏ, chúng ta có thể sử dụng thao tác thu nhỏ. Định dạng như sau:ĐẶT <source\_index>/\_shrink/<target\_index>.Hãy đưa ra lệnh thu nhỏ để thu nhỏtất cả các xemục lục.

**Liệt kê 6.44 Thu nhỏchỉ số**

ĐẶT tất cả các xe/\_shrink/tất cả các xe mới

{

"cài đặt":{ "index.blocks.write":null,

**Thu hẹp chỉ mục nguồn**

**Loại bỏ**

**Đặt**

"index.routing.allocation.require.\_name":null,

**chỉ đọc chỉ dẫn**

**tên nút**

**đến null**

"index.number\_of\_shards":1,

"index.number\_of\_replicas":5

}

}

**Giảm số lượng mảnh vỡ**

Chúng ta cần chỉ ra một vài điều trong tập lệnh này. Chỉ mục nguồn được thiết lập với hai thuộc tính: chỉ đọc và tên nút chỉ mục phân bổ. Các thiết lập này sẽ được chuyển sang chỉ mục đích mới nếu chúng ta không đặt lại chúng. Trong tập lệnh, chúng ta vô hiệu hóa các thuộc tính này để chỉ mục đích sẽ không bị áp đặt các hạn chế này khi nó được tạo. Chúng ta cũng thiết lập số lượng phân đoạn và bản sao trên chỉ mục đích mới khởi tạo và tạo liên kết cứng cho chỉ mục đích, trỏ đến các phân đoạn tệp chỉ mục nguồn.

**GHI CHÚ**Hãy nhớ rằng số lượng mảnh vỡ phải nhỏ hơn (hoặc bằng) số lượng mảnh vỡ của chỉ mục nguồn (xét cho cùng, chúng ta đang thu hẹp chỉ mục!). Và đúng vậy, như bạn có thể hình dung, số lượng mảnh vỡ của chỉ mục đích phải là một hệ số của số lượng mảnh vỡ của chỉ mục nguồn.

Trong khi chúng ta ở đây, chúng ta cũng có thể xóa tất cả các bản sao khỏi chỉ mục nguồn để hoạt động thu nhỏ dễ quản lý hơn. Chúng ta chỉ cần đặt thuộc tính index.number\_of\_replicas thành 0. Hãy nhớ rằng thuộc tính number\_of\_replicas là động, nghĩa là nó có thể được điều chỉnh trên chỉ mục trực tiếp.

**224 CPHẦN6*Hoạt động lập chỉ mục***

Chúng ta cũng phải thực hiện các hành động sau trước khi thu hẹp chỉ mục:

* *Chỉ mục nguồn phải được tắt (chuyển sang chế độ chỉ đọc) để lập chỉ mục.*Mặc dù không phải là con người-Ngày nay, chúng tôi khuyến cáo bạn nên tắt các bản sao trước khi thu nhỏ.
* *Chỉ mục mục tiêu không được tạo hoặc tồn tại trước hành động thu hẹp.*
* *Tất cả các phân đoạn chỉ mục phải nằm trên cùng một nút.*Chúng ta phải thiết lậpindex.routing

.allocation.require.<tên\_nút>thuộc tính trên chỉ mục có tên nút để đạt được điều này.

* *Số lượng phân mảnh của chỉ mục đích phải là một hệ số của số phân mảnh của chỉ mục nguồn.*Của chúng tôi

chỉ mục all\_cars có 50 mảnh chỉ có thể được thu nhỏ thành 25, 10, 5 hoặc 2 mảnh.

* *Nút của chỉ mục mục tiêu phải đáp ứng các yêu cầu về bộ nhớ.*

Chúng ta có thể sử dụng các hoạt động thu nhỏ khi chúng ta có nhiều mảnh vỡ nhưng dữ liệu được phân bố thưa thớt. Như tên gọi của nó, ý tưởng là giảm số lượng mảnh vỡ.

Tách hoặc thu hẹp chỉ mục là một cách hay để quản lý chúng khi dữ liệu của chúng ta tăng lên. Một hoạt động nâng cao khác là tạo chỉ mục trên một mẫu thiết lập với sự trợ giúp của cơ chế chuyển đổi.

* + 1. Lăn qua một bí danh chỉ mục

Index tích lũy dữ liệu theo thời gian. Đúng vậy, chúng ta có thể chia một index để xử lý dữ liệu bổ sung, như chúng ta đã thấy trước đó. Tuy nhiên, việc chia chỉ đơn giản là điều chỉnh lại dữ liệu thành các phân đoạn bổ sung. Elasticsearch cung cấp một cơ chế khác gọi là rollover, trong đó index hiện tại được tự động chuyển sang index trống mới.

Không giống như trong thao tác chia tách, trong thao tác rollover, các tài liệu không được sao chép vào chỉ mục mới. Chỉ mục cũ trở thành chỉ đọc và mọi tài liệu mới đều được lập chỉ mục vào chỉ mục rollover này từ bây giờ. Ví dụ, nếu chúng ta có chỉ mục app-000001, thao tác rollover sẽ tạo ra chỉ mục mới app-000002. Nếu chúng ta rollover thêm một lần nữa, một chỉ mục mới khác, app-000003, sẽ được khởi tạo, v.v.

Hoạt động rollover thường được sử dụng khi xử lý dữ liệu chuỗi thời gian. Dữ liệu chuỗi thời gian—dữ liệu được tạo cho một khoảng thời gian cụ thể như mỗi ngày, tuần hoặc tháng—thường được lưu giữ trong một chỉ mục được tạo cho một khoảng thời gian cụ thể. Ví dụ, nhật ký ứng dụng được tạo dựa trên ngày: logs-18-sept-21, logs-19-sept-21, v.v.

Điều này sẽ dễ hiểu khi bạn thấy nó hoạt động. Giả sử chúng ta có một chỉ mục cho xe hơi: cars\_2021-000001. Elasticsearch thực hiện một vài bước để chuyển qua chỉ mục này:

**1**Elasticsearch tạo một alias trỏ đến index (trong trường hợp này là cars\_2021- 000001). Trước khi tạo alias, chúng ta phải đảm bảo index có thể ghi bằng cách đặt is\_write\_index thành true. Ý tưởng là alias phải có ít nhất một index hỗ trợ có thể ghi.

**2**Elasticsearch gọi lệnh rollover trên alias bằng cách sử dụng API \_rollover. Lệnh này tạo ra một chỉ mục rollover mới (ví dụ: cars\_2021-000002).

**GHI CHÚ**Hậu tố theo sau (chẳng hạn như 000001) là một số dương và Elasticsearch mong đợi chỉ mục được tạo bằng hậu tố này. Elasticsearch chỉ có thể tăng từ một số dương; không quan trọng số bắt đầu là bao nhiêu.

* 1. ***Trình độ caohoạt động* 225**

Miễn là chúng ta có một số nguyên dương, Elasticsearch sẽ tăng số đó và tiến về phía trước. Ví dụ, nếu chúng ta cung cấp my-index-04 hoặc my-index-0004, chỉ mục rollover tiếp theo sẽ là my-index-000005. Elasticsearch tự động đệm hậu tố bằng số không.

Chúng tôi sẽ trình bày các bước này ở phần tiếp theo.

**CĐỌC MỘT BIỂU DANH CHO CÁC HOẠT ĐỘNG LĂN KHOẢN**

Điều đầu tiên chúng ta cần làm trước khi thực hiện thao tác rollover là tạo một alias trỏ đến index mà chúng ta muốn rollover. Chúng ta có thể sử dụng API rollover cho index hoặc data stream alias. Ví dụ, danh sách sau đây gọi API \_aliases để tạo aliasđược gọi là latest\_cars\_a cho chỉ mục cars\_2021-000001 (hãy đảm bảo bạn tạo chỉ mục này trước).

**Liệt kê 6.45 Tạo bí danh cho mộtchỉ số**

**Bộcủa hành động**

POST \_bí danh

{

"hành động": [

{

"thêm vào": {

**Sử dụng API \_aliases để gọi hành động thêm**

**Thêm vào**

**hoạt động Chỉ mục chúng tôi muốn tạo**

"index": "cars\_2021-000001", "alias": "latest\_cars\_a", "is\_write\_index": đúng

}

}

]

}

**cho dữ liệu lật xe**

**Tạo bí danh cho chỉ mục xe hơi**



**Làm cho chỉ mục có thể ghi được**

Các\_bí danhCơ quan yêu cầu API mong đợithêm vàohành động với một chỉ mục và bí danh của nó được xác định. Nó tạo ra bí danhxe\_mới\_nhất\_atrỏ đến một chỉ mục hiện có,xe hơi\_2021-000001, vớiBƯU KIỆNyêu cầu.

Một điểm quan trọng cần lưu ý: bí danh phải trỏ đến một chỉ mục có thể ghi—do đó chúng tôiđặt is\_write\_index thành true trong danh sách. Nếu alias trỏ đến nhiều index, ít nhất một index phải là một writeable index. Bước tiếp theo là chuyển index.

**TÔI****PHÁT HÀNH HOẠT ĐỘNG LĂN KHOẢN**

Bây giờ chúng ta đã tạo một bí danh, bước tiếp theo là gọi điểm cuối API rollover. Elasticsearch đã định nghĩa một API \_rollover cho mục đích này.

**Liệt kê 6.46 Lăn qua chỉ mục**

ĐĂNG latest\_cars\_a/\_rollover

Điểm cuối \_rollover được gọi trên alias, không phải index. Khi cuộc gọi thành công, một index mới, cars\_2021-000002, được tạo (\*-000001 tăng thêm 1). Sau đây là phản hồi cho cuộc gọi.

**226 CPHẦN6*Hoạt động lập chỉ mục***

**Liệt kê 6.47 Phản hồi từ một\_lăn quagọi**

{

"đã xác nhận": đúng, "shards\_acknowledged": đúng, "old\_index": "latest\_cars-000001", "new\_index": "latest\_cars-000002", "rolled\_over": đúng,

"dry\_run" : false, "điều kiện" : { }

}

**Tên chỉ mục cũ**

**Mớitên chỉ mục**

Như phản hồi cho biết, một chỉ mục mới (latest\_cars-000002) đã được tạo như một chỉ mục chuyển tiếp. Chỉ mục cũ đã được đưa vào chế độ chỉ đọc để mở đường cho việc lập chỉ mục các tài liệu trên chỉ mục chuyển tiếp mới được tạo.

**GHI CHÚ** API chuyển tiếp được áp dụng cho bí danh, nhưng chỉ mục đằng sau bí danh này là chỉ mục được chuyển tiếp.

Đằng sau hậu trường, việc gọi lệnh \_rollover trên bí danh sẽ thực hiện một số việc:

* Tạo một chỉ mục mới (xe\_2021-000002) có cấu hình giống như cấu hình cũ (tiền tố tên vẫn giữ nguyên, nhưng hậu tố sau dấu gạch ngang sẽ tăng lên).
* Ánh xạ lại bí danh để trỏ đến chỉ mục mới được tạo mới (xe\_2021-000002(trong trường hợp này). Các truy vấn của chúng tôi không bị ảnh hưởng vì tất cả các truy vấn đều được viết theo một bí danh (không phải chỉ mục vật lý).
* Xóa bí danh trên chỉ mục hiện tại và trỏ lại vào chỉ mục chuyển tiếp mới được tạo.

Khi chúng ta gọi lệnh \_rollover, Elasticsearch thực hiện một loạt các hành động (hãy nhớ rằng chỉ mục hiện tại phải có một bí danh trỏ đến chỉ mục như một điều kiện tiên quyết):

* Làm cho chỉ mục hiện tại chỉ đọc (do đó chỉ các truy vấn được thực thi)
* Tạo một chỉ mục mới với quy ước đặt tên phù hợp
* Trỏ lại bí danh vào chỉ mục mới này

Nếu chúng ta gọi lại lệnh gọi trong danh sách 6.47, một chỉ mục mới cars\_2021-000003 được tạo ra và bí danh được gán lại cho chỉ mục mới này thay vì chỉ mục cũ cars\_2021-000002. Khi chúng ta cần chuyển dữ liệu sang chỉ mục mới, chỉ cần gọi \_rollover trên bí danh là đủ.

**Quy ước đặt tên**

Chúng ta hãy cùng tìm hiểu về các quy ước đặt tên mà chúng ta sử dụng khi chuyển qua các chỉ mục.

\_lăn quaAPI có hai định dạng—một định dạng mà chúng ta có thể cung cấp tên chỉ mục

ĐĂNG <bí danh chỉ mục>/\_lăn qua

* 1. ***Quản lý vòng đời chỉ mục(ILM)* 227**

và một cái khác do hệ thống xác định:

ĐĂNG <bí danh chỉ mục>/\_rollover/<tên\_chỉ\_mục\_tiêu>

Chỉ định tên chỉ mục đích, như được đưa ra trong tùy chọn thứ hai, cho phép API rollover tạo chỉ mục với tham số được đưa ra làm tên chỉ mục đích. Tuy nhiên, tùy chọn đầu tiên, trong đó chúng tôi không cung cấp tên chỉ mục, có một quy ước đặc biệt:<tên\_chỉ\_số>-00000*N*. Số (sau dấu gạch ngang) luôn là sáu chữ số với số không được đệm. Nếu chỉ mục của chúng ta theo định dạng này, việc lăn qua sẽ tạo ra một chỉ mục mới có cùng tiền tố, nhưng hậu tố sẽ tự động tăng lên số tiếp theo:<tên\_chỉ\_mục>

-00000*N*+1. Sự gia tăng bắt đầu từ bất cứ nơi nào số chỉ mục ban đầu là: ví dụ,my\_cars-000034được tăng lênmy\_cars-000035.

Bạn có thể tự hỏi khi nào bạn muốn chuyển qua một chỉ mục. Điều đó tùy thuộc vào bạn. Khi bạn nghĩ rằng chỉ mục bị tắc nghẽn hoặc bạn cần (di chuyển lại) dữ liệu cũ hơn, bạn có thể gọi chuyển qua. Tuy nhiên, trước tiên hãy tự hỏi:

* + Bạn có thể tự động chuyển chỉ mục khi kích thước của phân đoạn vượt quá ngưỡng nhất định không?
  + Bạn có thể tạo một chỉ mục mới cho nhật ký hàng ngày không?

Mặc dù chúng ta đã thấy cơ chế chuyển đổi trong phần này, chúng ta có thể giải đáp những câu hỏi này bằng cách sử dụng tính năng quản lý vòng đời chỉ mục (ILM) tương đối mới, được thảo luận chi tiết trong phần tiếp theo.

#### 6.9 Quản lý vòng đời chỉ mục(ILM)

Các chỉ mục được kỳ vọng sẽ tăng kích thước khi dữ liệu đổ vào theo thời gian. Đôi khi một chỉ mục được ghi quá thường xuyên và các mảnh vỡ cơ bản hết bộ nhớ; những lần khác, hầu hết các mảnh vỡ được điền thưa thớt. Sẽ không lý tưởng sao nếu tự động cuộn qua chỉ mục trong trường hợp trước và thu nhỏ nó trong trường hợp sau?

**GHI CHÚ**Quản lý vòng đời chỉ mục (ILM) là một chủ đề nâng cao và bạn có thể không cần đến nó khi bắt đầu sử dụng Elasticsearch. Nếu đúng như vậy, bạn có thể bỏ qua phần này và quay lại khi bạn cần tìm hiểu cách Elasticsearch xử lý dữ liệu chuỗi thời gian và cách hành động trên các chỉ mục bằng cách chuyển chúng sang, đóng băng hoặc xóa chúng dựa trên một số điều kiện "tự động" (tức là dựa trên các chính sách đã xác định) và nhiều hơn nữa.

Chúng ta cũng cần xem xét dữ liệu chuỗi thời gian. Lấy ví dụ về nhật ký được ghi vào tệp hàng ngày. Các nhật ký này sau đó được xuất sang các chỉ mục có hậu tố là dấu chấm, như my-app-2021-10

-24.log. Khi một ngày được chuyển sang ngày tiếp theo, chỉ số tương ứng cũng phải được chuyển sang ngày tiếp theo: ví dụ,my-app-2021-10-24.logĐẾNmy-app-2021-10-25.log(ngày là

tăng dần theo ngày), như hình 6.10 cho thấy.

**Hình 6.10 Lăn sang một chỉ mục mới khingày mới đang đến**

**228 CPHẦN6*Hoạt động lập chỉ mục***

Chúng ta có thể viết một công việc theo lịch trình để thực hiện việc này cho chúng ta. Nhưng may mắn thay, Elastic gần đây đã phát hành một tính năng mới có tên là quản lý vòng đời chỉ mục (ILM). Như tên gọi của nó, ILM là về việc quản lý các chỉ mục dựa trên chính sách vòng đời. Chính sách là một định nghĩa khai báo các quy tắc được thực thi bởi công cụ khi các điều kiện của các quy tắc được đáp ứng. Ví dụ, chúng ta có thể định nghĩa các quy tắc dựa trên việc chuyển chỉ mục hiện tại sang chỉ mục mới khi

* Chỉ mục đạt đến một kích thước nhất định (ví dụ: 40 GB)
* Số lượng tài liệu trong chỉ mục đạt tới, chẳng hạn, 10.000
* Ngày trôi qua

Trước khi bắt đầu viết chính sách, chúng ta hãy xem xét vòng đời của một chỉ mục: các giai đoạn mà một chỉ mục trải qua dựa trên các tiêu chí và điều kiện.

* + 1. Vòng đời chỉ mục

Một chỉ mục có năm giai đoạn vòng đời: nóng, ấm, lạnh, đóng băng và xóa, như thể hiện trong hình 6.11. Chúng ta hãy mô tả ngắn gọn từng giai đoạn này:

* + - * *Nóng*—Chỉ mục đang ở chế độ hoạt động đầy đủ. Chỉ mục có thể đọc và ghi, do đó cho phép chỉ mục thực hiện cả lập chỉ mục và truy vấn.
      * *Ấm*—Chỉ mục ở chế độ chỉ đọc. Lập chỉ mục bị tắt, nhưng chỉ mục vẫn mở để truy vấn để có thể phục vụ các truy vấn tìm kiếm và tổng hợp.
      * *Lạnh lẽo*—Chỉ mục ở chế độ chỉ đọc. Tương tự như giai đoạn ấm, lập chỉ mục bị tắt nhưng chỉ mục vẫn mở để truy vấn, mặc dù các truy vấn được mong đợi là không thường xuyên. Khi chỉ mục ở giai đoạn này, các truy vấn tìm kiếm có thể dẫn đến thời gian phản hồi chậm.

**Chế độ hoạt động đầy đủ. Chỉ mục có sẵn cho cả hoạt động đọc và ghi.**

**Chỉ mục chỉ đọc. Việc truy vấn dự kiến ​​sẽ không thường xuyên và chậm.**

**Giai đoạn cuối cùng.**

**Chỉ mục sẽ bị xóa vĩnh viễn.**

NÓNG



XÓA BỎ

ĐÔNG LẠNH

LẠNH LẼO

ẤM

**Chỉ mục chỉ đọc, do đó không được phép lập chỉ mục. Cho phép truy vấn thường xuyên.**

**Chỉ mục chỉ đọc. Việc truy vấn được dự kiến ​​là hiếm hoặc rất ít khi xảy ra và chậm chạp.**

**Nóng là giai đoạn bắt buộc duy nhất. Chỉ số có thể sống ở chế độ nóng mãi mãi hoặc chuyển từ nóng sang bất kỳ chế độ tùy chọn nào khác.**

**Hình 6.11 Vòng đời của một chỉ mục**

***6.9 Quản lý vòng đời chỉ mục(ILM)* 229**

* + - * + *Đông lạnh*—Tương tự như giai đoạn lạnh, khi chỉ mục bị tắt để lập chỉ mục nhưng vẫn cho phép truy vấn. Tuy nhiên, các truy vấn ít thường xuyên hơn hoặc thậm chí hiếm hơn. Khi chỉ mục ở giai đoạn này, người dùng có thể nhận thấy thời gian phản hồi truy vấn lâu hơn.
        + *Xóa bỏ*—Đây là giai đoạn cuối cùng của chỉ mục, nơi nó bị xóa vĩnh viễn. Như vậy, dữ liệu bị xóa và tài nguyên được giải phóng. Thông thường, chúng tôi chụp ảnh nhanh chỉ mục trước khi xóa nó để dữ liệu từ ảnh chụp nhanh có thể được khôi phục trong tương lai.

Chuyển đổi từ pha nóng sang mọi pha khác là tùy chọn. Nghĩa là, sau khi được tạo trong pha nóng, một chỉ mục có thể duy trì ở pha đó hoặc chuyển sang bất kỳ pha nào trong bốn pha khác. Trong các phần sau, chúng tôi sẽ xem xét một số ví dụ thiết lập chính sách vòng đời lập chỉ mục để hệ thống có thể tự động quản lý các chỉ mục.

* + 1. Quản lý vòng đời chỉ mục theo cách thủ công

Cho đến nay, chúng tôi đã tạo hoặc xóa một chỉ mục theo yêu cầu khi cần (can thiệp thủ công). Nhưng chúng tôi không thể xóa, chuyển tiếp hoặc thu hẹp các chỉ mục dựa trên các điều kiện như kích thước của chỉ mục vượt quá ngưỡng nhất định, sau một số ngày nhất định, v.v. Chúng tôi có thể sử dụng ILM để giúp chúng tôi thiết lập tính năng này.

Elasticsearch cung cấp API để làm việc với chính sách vòng đời chỉ mục với

Điểm cuối \_ilm. Định dạng là \_ilm/policy/<index\_policy\_name>. Quy trình được chia thành hai bước: xác định chính sách vòng đời và liên kết chính sách đó với chỉ mục để thực thi.

**STEP1: ĐĐỊNH NGHĨA CHÍNH SÁCH VÒNG ĐỜI**

Bước đầu tiên là xác định chính sách vòng đời, trong đó chúng ta cung cấp các giai đoạn cần thiết và thiết lập các hành động liên quan trong các giai đoạn đó.

 **Liệt kê 6.48 Tạo chính sách với hot và deletecác giai đoạn**

PUT \_ilm/policy/hot\_delete\_policy

{

**ILM là gì?Giao diện lập trình ứng dụng (API)**

"chính sách": {

"giai đoạn": {

"nóng": {

**Xác định chính sách và các giai đoạn của nó**

**Xác định giai đoạn đầu tiên (nóng)**

**Đặt mức độ ưu tiên**

"min\_age": "1 ngày", "hành động": {

"set\_priority": { "priority": 250

}

**Đặt độ tuổi tối thiểu trước khi thực hiện các hành động khác**

**Xác định các hành động phải được thực hiện**



}

},

"xóa bỏ": {

"hành động": {

"xóa bỏ" : { }

**Xác định giai đoạn xóa**

}

}

}

}

}

**230 CPHẦN6*Hoạt động lập chỉ mục***

hot\_delete\_policy định nghĩa một chính sách có hai giai đoạn: hot và delete. Sau đây là nội dung định nghĩa:

* *Giai đoạn nóng*—Chỉ mục dự kiến ​​sẽ tồn tại ít nhất một ngày trước khi thực hiện các hành động. Khối được xác định tronghành độngđối tượng đặt mức độ ưu tiên (250, trong ví dụ này). Các chỉ mục có mức độ ưu tiên cao hơn sẽ được xem xét trước trong quá trình khôi phục nút.
* *Xóa pha*—Chỉ mục sẽ bị xóa sau khi giai đoạn nóng hoàn tất tất cả các hành động. Vì không cótuổi\_của tôitrong giai đoạn xóa, hành động xóa sẽ diễn ra ngay lập tức khi giai đoạn nóng kết thúc.

Làm thế nào để chúng ta gắn chính sách này vào một chỉ mục? Đó là điều chúng ta sẽ thảo luận tiếp theo.

Bây giờ chúng ta đã xác định chính sách, bước tiếp theo là liên kết một chỉ mục với chính sách đó. Để xem điều này hoạt động như thế nào, hãy tạo một chỉ mục và đính kèm chính sách từ danh sách 6.48 vào đó.

**Liệt kê 6.49 Tạo một chỉ mục có vòng đời chỉ mục liên quan**

PUT hot\_delete\_policy\_index

{

"cài đặt": {

**Sử dụng đối tượng cài đặt để thiết lập thuộc tính**

"index.lifecycle.name":"hot\_delete\_policy" # Tên của chính sách

}

}

Tập lệnh này tạo chỉ mục hot\_delete\_policy\_index với thiết lập thuộc tính trên chỉ mục: index.lifecycle.name. Chỉ mục hiện được liên kết với chính sách vòng đời, vì index.lifecycle.name trỏ đến chính sách đã tạo trước đó của chúng tôi (hot\_delete\_policy) từ danh sách 6.48. Điều này có nghĩa là chỉ mục trải qua quá trình chuyển đổi pha dựa trên định nghĩa chính sách. Khi chỉ mục được tạo, trước tiên nó sẽ vào pha hot và duy trì trong pha đó trong một ngày (min\_age=1d, như được định nghĩa trong chính sách của chúng tôi) trước khi áp dụng một vài hành động (trong trường hợp này, thiết lập mức độ ưu tiên trên chỉ mục).

Ngay khi giai đoạn nóng hoàn tất (một ngày, dựa trên định nghĩa chính sách), chỉ mục chuyển sang giai đoạn tiếp theo: giai đoạn xóa, trong trường hợp này. Đây là giai đoạn đơn giản, trong đó chỉ mục được xóa tự động.

**GHI CHÚ**Chính sách hot\_delete\_policy được định nghĩa trong danh sách 6.48 xóa chỉ mục sau một ngày dựa trên định nghĩa. Lưu ý rằng nếu bạn sử dụng chính sách này trong sản xuất, bạn có thể không tìm thấy chỉ mục nào khả dụng sau giai đoạn xóa (giai đoạn xóa sẽ xóa mọi thứ).

Tóm lại, việc gắn chính sách vòng đời chỉ mục vào chỉ mục sẽ chuyển chỉ mục thành các giai đoạn nhất định và thực hiện các hành động nhất định được xác định trong mỗi giai đoạn. Chúng ta có thể xác định một chính sách phức tạp như giai đoạn nóng trong 45 ngày, giai đoạn ấm trong một tháng, sau đó là giai đoạn lạnh trong một năm và cuối cùng xóa chỉ mục sau một năm.

Giả sử chúng ta muốn chuyển đổi chỉ mục dựa trên các điều kiện, chẳng hạn như hàng tháng hoặc ở một kích thước cụ thể. May mắn thay, Elasticsearch cung cấp các vòng đời chỉ mục tự động và có điều kiện.

***6.9 Quản lý vòng đời chỉ mục(ILM)* 231**

* + 1. Vòng đời với chuyển tiếp

Trong phần này, chúng tôi đặt điều kiện cho chỉ số chuỗi thời gian để chuyển tiếp khi các điều kiện đó được đáp ứng. Giả sử chúng tôi muốn chỉ số được chuyển tiếp dựa trên các điều kiện sau:

* + - * Vào mỗi ngày mới
      * Khi số lượng tài liệu đạt 10.000
      * Khi kích thước chỉ mục đạt tới 10 GB

Đoạn mã sau đây định nghĩa chính sách vòng đời đơn giản, khai báo giai đoạn nóng trong đó các phân đoạn dự kiến ​​sẽ chuyển tiếp dựa trên các điều kiện này, với một vài hành động.

**Liệt kê 6.50 Định nghĩa chính sách đơn giản cho giai đoạn nóng**

ĐẶT \_ilm/policy/hot\_simple\_policy

{

"chính sách": {

"giai đoạn": {

"nóng": {

**Tuyên bố một**

**nónggiai đoạn Chỉ số nhập vào đây**

"min\_age": "0ms", "hành động": {

"rollover": { "max\_age": "1 ngày", "max\_docs": 10000, "max\_size": "10gb"

**pha ngay lập tức.**

**Chỉ số sẽ được chuyển tiếp nếu bất kỳ điều kiện nào được đáp ứng.**

}

}

}

}

}

}

Trong chính sách của chúng tôi, chúng tôi khai báo một giai đoạn—giai đoạn nóng—với rollover là hành động được thực hiện khi bất kỳ điều kiện nào được khai báo trong các hành động rollover được đáp ứng: ví dụ, nếu số lượng tài liệu đạt 10.000, kích thước chỉ mục vượt quá 10 GB hoặc chỉ mục đã một ngày tuổi, thì index sẽ rollover. Vì chúng tôi khai báo tuổi tối thiểu (min\_age) là 0ms, nên ngay khi index được tạo, index sẽ ngay lập tức được chuyển vào giai đoạn nóng và sau đó rollover.

Bước tiếp theo là tạo mẫu lập chỉ mục và đính kèm chính sách vòng đời vào mẫu đó.

Đoạn mã sau đây khai báo một mẫu chỉ mục với mẫu chỉ mục mysql-\*.

**Liệt kê 6.51 Đính kèm chính sách vòng đời vàobản mẫu**

ĐẶT \_index\_template/mysql\_logs\_template

{

"index\_patterns": ["mysql-\*"], "mẫu":{

"cài đặt":{

**Mẫu chỉ mục cho tất cả các chỉ mục MySQL**

**Đính kèm**

"index.lifecycle.name":"hot\_simple\_policy",

**chính sách**

**232 CPHẦN6*Hoạt động lập chỉ mục***

"index.lifecycle.rollover\_alias":"mysql-logs-alias"

}

}

}

**Đính kèm một bí danh**

Chúng ta cần lưu ý một vài điều từ tập lệnh này. Chúng ta phải liên kết chính sách chỉ mục đã định nghĩa trước đó của mình bằng cách đặt nó là index.lifecycle.name với mẫu chỉ mục này. Ngoài ra, vì định nghĩa của chính sách có giai đoạn nóng với rollover được xác định, chúng ta phải cung cấp tên index.lifecycle.rollover\_alias khi tạo mẫu chỉ mục này.

Bước cuối cùng là tạo một chỉ mục khớp với mẫu chỉ mục được xác định trong mẫu chỉ mục, với một số làm hậu tố để các chỉ mục chuyển tiếp được tạo chính xác. Lưu ý rằng chúng ta phải xác định bí danh và khai báo rằng chỉ mục hiện tại có thể ghi được bằng cách đặt is\_write\_index thành true.

**Liệt kê 6.52 Thiết lập chỉ mục có thể ghi chobí danh**

ĐẶT mysql-index-000001

{

"bí danh": {

"mysql-logs-alias": { "is\_write\_index":đúng



}

}

}

**Tạo một chỉ mục có định dạng phù hợp**

**Cho phép bí danh**

**Chỉ mục sao lưu phải có thể ghi được.**

Khi chúng ta tạo chỉ mục, chính sách sẽ được kích hoạt. Trong ví dụ của chúng tôi, chỉ mục nhập vàopha nóng, vì min\_age được đặt thành 0 mili giây, và sau đó chuyển sang hành động chuyển pha. Chỉ mục ở lại pha này cho đến khi một trong các điều kiện (tuổi, kích thước của chỉ mục hoặc số lượng tài liệu) được đáp ứng. Ngay khi điều kiện là tích cực, pha chuyển pha được thực thi và chỉ mục mới mysql-index-000002 được tạo (lưu ý hậu tố chỉ mục). Bí danh được ánh xạ lại để tự động trỏ đến chỉ mục mới này. Sau đó, mysql-index-000002 được chuyển sang chỉ mục mysql-index-000003 (một lần nữa, nếu một trong các điều kiện được đáp ứng) và chu kỳ tiếp tục.

**Khoảng thời gian quét chính sách**

Theo mặc định, chính sáchđược quét mỗi 10 phút. Để thay đổi thời gian quét này, chúng ta cần cập nhật cài đặt cụm bằng cách sử dụng\_cụmđiểm cuối.

Khi chúng ta thử nghiệm các chính sách vòng đời trong quá trình phát triển, một vấn đề thường gặp là không có giai đoạn nào được thực hiện. Ví dụ, mặc dù chúng ta đặt thời gian của các giai đoạn (tuổi\_của tôi,tuổi\_tối\_đa) tính bằng mili giây, không có pha nào được thực hiện. Nếu chúng ta không biếttrong khoảng thời gian quét, chúng ta có thể nghĩ rằng các chính sách vòng đời không được áp dụng—nhưng thực tế là các chính sách này đang chờ được quét.

Chúng ta có thể thiết lập lại thời gian quét bằng cách gọi\_cluster/cài đặtđiểm cuối vớikhoảng thời gian thích hợp. Ví dụ, đoạn mã sau đặt lại khoảng thời gian thăm dò thành 10 mili giây:

***6.9 Quản lý vòng đời chỉ mục(ILM)* 233**

ĐẶT \_cluster/cài đặt

{

"liên tục": { "indices.lifecycle.poll\_interval": "10ms"

}

}

Bây giờ chúng ta đã hiểu về việc chuyển đổi chỉ mục bằng cách sử dụng các chính sách vòng đời, hãy cùng viết một chính sách khác với nhiều giai đoạn.

**Liệt kê 6.53 Tạo chính sách vòng đời nâng cao**

ĐẶT \_ilm/policy/hot\_warm\_delete\_policy

{

"chính sách": {

"giai đoạn": {

"nóng": { "min\_age": "1 ngày", "hành động": {

"rollover": { "max\_size": "40gb",

"max\_age": "6 ngày"

},

"set\_priority": { "priority": 50

**Chỉ số này phải đợi một ngày mới tăng trở lại.**

**Lăn qua khi một trong các điều kiện được đáp ứng**

**Đặt mức độ ưu tiên (một hành động bổ sung)**

}

}

},

"ấm áp": { "min\_age": "7 ngày", "hành động": {

"thu nhỏ": { "số\_mảnh\_shards":

**Chờ bảy ngày trước**

**thực hiện các hành động**

**Thu hẹp chỉ số**



1

}

}

}, **Xóa bỏ**

"xóa": { "min\_age": "30 ngày", "hành động": {

"xóa bỏ": {}

**chỉ mục, nhưng …**

**… đầu tiên ở giai đoạn này trong 30 ngày.**

}

}

}

}

}

Chính sách này bao gồm các giai đoạn nóng, ấm và xóa. Hãy cùng xem điều gì xảy ra và những hành động nào được thực hiện trong các giai đoạn này:

* *Giai đoạn nóng*—Chỉ số đi vào giai đoạn này sau một ngày vìtuổi\_của tôithuộc tính được thiết lập thành1 ngày. Sau một ngày, chỉ số chuyển sang giai đoạn chuyển tiếp và chờ đợi

**234 CPHẦN6*Hoạt động lập chỉ mục***

để các điều kiện được thỏa mãn: kích thước tối đa là 40 GB ("max\_size":"40gb") hoặc tuổi thọ đã hơn sáu ngày ("max\_age": "6d"). Khi một trong những điều kiện này được đáp ứng, chỉ số sẽ chuyển từ giai đoạn nóng sang giai đoạn ấm.

* *Giai đoạn ấm áp*—Khi chỉ số bước vào giai đoạn ấm áp, nó sẽ duy trì ở đó trong một tuần ("min\_age": "7 ngày") trước khi bất kỳ hành động nào của nó được thực hiện. Sau ngày thứ bảy, chỉ mục được thu nhỏ thành một nút ("số\_mảnh\_shards": 1), sau đó chỉ mục sẽ bị xóa.
* *Xóa pha*—Chỉ số duy trì ở giai đoạn này trong 30 ngày ("min\_age": "30 ngày"). Sau thời gian này, chỉ mục sẽ bị xóa vĩnh viễn. Hãy cẩn thận với giai đoạn này, vì thao tác xóa là không thể đảo ngược! Lời khuyên của tôi là hãy sao lưu dữ liệu trước khi bạn xóa dữ liệu vĩnh viễn.

Đã đến lúc kết thúc. Trong chương này, chúng ta đã học được nhiều điều về hoạt động lập chỉ mục và ILM. Trong chương tiếp theo, chúng ta sẽ đi sâu vào phân tích văn bản, vì vậy hãy chú ý theo dõi.

#### Bản tóm tắt

* Elasticsearch cung cấp API chỉ mục để tạo, đọc, xóa và cập nhật chỉ mục.
* Mỗi chỉ mục có ba bộ cấu hình: bí danh, cài đặt và ánh xạ.
* Chỉ mục có thể được tạo ngầm định hoặc rõ ràng:
  + Quá trình tạo ngầm định bắt đầu khi chỉ mục không tồn tại và tài liệu được lập chỉ mục lần đầu tiên. Cấu hình mặc định (chẳng hạn như một bản sao và một phân đoạn) được áp dụng cho chỉ mục được tạo ngầm định.
  + Việc tạo rõ ràng xảy ra khi chúng ta khởi tạo các chỉ mục với một tập hợp cấu hình tùy chỉnh bằng cách sử dụng API chỉ mục.
* Mẫu chỉ mục cho phép chúng ta tạo các chỉ mục với các thiết lập cấu hình được xác định trước, dựa trên tên trùng khớp, được áp dụng trong quá trình tạo chỉ mục.
* Có thể thay đổi kích thước chỉ mục bằng cơ chế thu nhỏ hoặc chia tách. Thu nhỏ làm giảm số lượng phân đoạn, trong khi chia tách thêm nhiều phân đoạn chính hơn.
* Một chỉ mục có thể được chuyển tiếp có điều kiện khi cần thiết.
* Quản lý vòng đời chỉ mục (ILM) giúp chuyển đổi chỉ mục giữa các giai đoạn vòng đời này: nóng, ấm, lạnh, đông lạnh và xóa. Trong giai đoạn nóng, chỉ mục hoạt động hoàn toàn và mở để tìm kiếm và lập chỉ mục; nhưng trong giai đoạn ấm và lạnh, chỉ mục chỉ đọc.

*Phân tích văn bản*

***Chương này bao gồm***

* Tổng quan về phân tích văn bản
* Giải phẫu của một máy phân tích
* Bộ phân tích tích hợp
* Phát triển các trình phân tích tùy chỉnh
* Hiểu về tokenizer
* Tìm hiểu về bộ lọc ký tự và mã thông báo

Elasticsearch thực hiện nhiều công việc nền tảng (và công việc nặng nhọc) đằng sau hậu trường đối với dữ liệu văn bản đến. Nó chuẩn bị dữ liệu để dữ liệu có thể được lưu trữ và tìm kiếm hiệu quả. Nói một cách ngắn gọn, Elasticsearch dọn dẹp các trường văn bản, chia dữ liệu văn bản thành các mã thông báo riêng lẻ và làm giàu các mã thông báo trước khi lưu trữ chúng trong các chỉ mục đảo ngược. Khi thực hiện truy vấn tìm kiếm, chuỗi truy vấn được tìm kiếm theo các mã thông báo đã lưu trữ và mọi kết quả khớp đều được truy xuất và chấm điểm. Quá trình chia văn bản thành các mã thông báo riêng lẻ và lưu trữ trong các cấu trúc bộ nhớ trong này được gọi là phân tích văn bản.

Mục đích của phân tích văn bản không chỉ là trả về kết quả tìm kiếm nhanh chóng và hiệu quả mà còn là thu thập các kết quả có liên quan. Công việc được thực hiện bằng cách sử dụng các trình phân tích:

**235**

**236 CPHẦN7*Phân tích văn bản***

các thành phần phần mềm được xây dựng sẵn để kiểm tra văn bản đầu vào theo các quy tắc khác nhau. Ví dụ, nếu người dùng tìm kiếm “K8s”, chúng ta sẽ có thể tìm được sách trên Kubernetes. Tương tự, nếu các câu tìm kiếm bao gồm biểu tượng cảm xúc như  (cà phê), công cụ tìm kiếm phải có khả năng trích xuất các kết quả phù hợp với cà phê. Những tiêu chí tìm kiếm này và nhiều tiêu chí khác được công cụ này tôn trọng nhờ cách chúng tôi cấu hình các trình phân tích.

Trong chương này, chúng ta sẽ xem xét chi tiết cơ chế trích xuất và phân tích văn bản. Chúng ta bắt đầu bằng cách xem xét một trong những trình phân tích phổ biến—trình phân tích chuẩn—là trình phân tích mặc định cho phép chúng ta dễ dàng làm việc trên văn bản tiếng Anh bằng cách phân chia các từ bằng khoảng trắng và dấu câu cũng như viết thường các mã thông báo cuối cùng. Nó cũng hỗ trợ tùy chỉnh: nghĩa là, nếu chúng ta muốn dừng lập chỉ mục một tập hợp các giá trị được xác định trước (có thể là các từ phổ biến như a, an, the, and, if, v.v. hoặc các từ chửi thề), chúng ta có thể tùy chỉnhtomize trình phân tích để thực hiện điều đó. Ngoài việc làm việc với trình phân tích chuẩn, sau này trong chương, chúng ta sẽ xem xét các trình phân tích được dựng sẵn như keyword, simple, stop, whitespace, pattern và các trình phân tích khác. Chúng ta cũng sẽ thực hành với các trình phân tích ngôn ngữ cụ thể cho tiếng Anh, tiếng Đức, tiếng Tây Ban Nha, tiếng Pháp, tiếng Hindi, v.v. Chúng ta sẽ bắt đầu bằng phần tổng quan về phân tích văn bản.

**GHI CHÚ** Mã cho chương này có sẵn trên GitHub ([http://mng.bz/](http://mng.bz/JgYQ) [JgYQ](http://mng.bz/JgYQ)) và trên trang web của cuốn sách ([https://www.manning.com/books/elastic-](https://www.manning.com/books/elasticsearch-in-action-second-edition) [tìm kiếm-trong-hành-động-phiên-bản-thứ-hai](https://www.manning.com/books/elasticsearch-in-action-second-edition)).

#### Tổng quan

Elasticsearch lưu trữ cả dữ liệu có cấu trúc và không có cấu trúc. Như chúng ta đã thấy trong các chương trước, làm việc với dữ liệu có cấu trúc rất đơn giản. Chúng tôi khớp các tài liệu cho truy vấn đã cho và trả về kết quả: ví dụ, truy xuất thông tin của khách hàng theo địa chỉ email của họ, tìm các chuyến bay bị hủy giữa các ngày, lấy số liệu bán hàng trong quý trước, lấy danh sách bệnh nhân được chỉ định cho một bác sĩ phẫu thuật vào một ngày nhất định, v.v. Kết quả là chắc chắn: kết quả được trả về nếu truy vấn khớp với các tài liệu hay không nếu truy vấn không khớp.

Mặt khác, truy vấn dữ liệu phi cấu trúc liên quan đến việc xác định xem một tài liệu có khớp với truy vấn hay không và tài liệu đó có liên quan đến truy vấn như thế nào (tài liệu khớp tốt như thế nào). Ví dụ, tìm kiếm "Konda" trên các tiêu đề sách sẽ lấy Elasticsearch in Action và các cuốn sách khác mà tôi đã viết nhưng không phải các bộ phim liên quan đến anaconda.

**Các bộ phân tích được áp dụng trong quá trình tìm kiếm**

Văn bản được phân tích khi dữ liệu được lập chỉ mục cũng như tại thời điểm truy vấn. Cũng giống như một trường được phân tích trong quá trình lập chỉ mục, truy vấn tìm kiếm cũng trải qua cùng một quy trình. Cùng một trình phân tích thường được sử dụng trong quá trình tìm kiếm, nhưng các yêu cầu của chúng tôi có thể khiến một trình phân tích khác phù hợp hơn. Phần 7.5 thảo luận về việc sử dụng các trình phân tích khác nhau để tìm kiếm và chỉ định các trình phân tích cần thiết để lập chỉ mục.

* 1. ***Tổng quan* 237**
     1. Truy vấn dữ liệu phi cấu trúc

Dữ liệu phi cấu trúc là thông tin không phù hợp với các bảng hoặc cơ sở dữ liệu thông thường. Dữ liệu này thường chứa văn bản, nhưng cũng có thể bao gồm những thứ như ngày tháng, số liệu hoặc sự kiện. Ví dụ về dữ liệu phi cấu trúc là email, tài liệu, ảnh, bài đăng trên mạng xã hội, v.v.

Giả sử công cụ tìm kiếm của chúng ta chứa câu trích dẫn này của Albert Einstein:

"trích dẫn": "Trí tưởng tượng quan trọng hơn kiến ​​thức"

Người dùng nhận được kết quả tích cực với các truy vấn bao gồm một từ riêng lẻ hoặc tổ hợp các từ (“trí tưởng tượng”, “kiến thức”, v.v.). Bảng 7.1 hiển thị một tập hợp các từ khóa tìm kiếm mà người dùng có thể tìm kiếm trong trường hợp này và các kết quả mong đợi.

**Bảng 7.1 Các truy vấn tìm kiếm có thể và kết quả mong đợi**

|  |  |  |
| --- | --- | --- |
| **Tìm kiếm từ khóa** | **Kết quả** | **Ghi chú** |
| “trí tưởng tượng”, “kiến thức” | Đúng | Từ khóa riêng lẻ khớp chính xác, do đó kết quả tích cực |
|  |  | được trả lại. |
| “kiến thức tưởng tượng”, | Đúng | Các từ khóa kết hợp cũng phù hợp với tài liệu và do đó |
| "kiến thứcquan trọng" |  | trả về kết quả. |

Tuy nhiên, tìm kiếm theo các tiêu chí khác có thể không mang lại kết quả nào. Ví dụ, như được hiển thị trong bảng 7.2, nếu người dùng tìm kiếm “passion”, “importance”, “passionate wisdom”, “curious cognizance” hoặc các thuật ngữ tương tự, công cụ không trả về kết quả khớp với các thiết lập mặc định. Các từ như passion và cognizance là từ đồng nghĩa trong trích dẫn. (Tương tự, các thiết lập mặc định không có từ viết tắt.) Chúng ta có thể điều chỉnh các trình phân tích để tôn trọng các từ đồng nghĩa, từ gốc, lỗi chính tả, v.v., như chúng ta sẽ thấy sau trong chương này.

**Bảng 7.2 Các truy vấn tìm kiếm có thể không có kết quả**

|  |  |  |
| --- | --- | --- |
| **Tìm kiếm từ khóa** | **Kết quả** | **Ghi chú** |
| “tưởng tượng”, “đam mê”, “tò mò”, “tầm quan trọng”, “nhận thức”, “trí tuệ”, “trí tuệ đam mê”, “cực kỳ quan trọng”  “quan trọng”, “kiến thức”, “tưởng tượng”  Imp, KNWL, IMGN | KHÔNG  không không | Từ đồng nghĩa và tên thay thế không mang lại kết quả tích cực.  Lỗi chính tả có thể dẫn đến kết quả không khớp hoặc không khớp.  Viết tắt không trả về kết quả tích cực. |

Chúng ta nên mong đợi người dùng truy vấn công cụ tìm kiếm của chúng ta bằng nhiều kết hợp: từ đồng nghĩa, từ viết tắt, từ viết tắt, biểu tượng cảm xúc, thuật ngữ chuyên ngành, v.v. Một công cụ tìm kiếm giúp tìm ra câu trả lời có liên quan cho nhiều tiêu chí tìm kiếm khác nhau luôn chiến thắng.

**238 CPHẦN7*Phân tích văn bản***

* + 1. Máy phân tích để giải cứu

Để xây dựng một công cụ tìm kiếm thông minh không bao giờ làm thất vọng, chúng tôi cung cấp cho công cụ hỗ trợ thêm trong quá trình lập chỉ mục dữ liệu bằng cách sử dụng phân tích văn bản do các mô-đun phần mềm gọi là trình phân tích thực hiện. Để phục vụ nhiều truy vấn, chúng tôi phải chuẩn bị công cụ tìm kiếm trên dữ liệu mà nó sử dụng trong giai đoạn lập chỉ mục.

**Chỉ có các trường văn bản được phân tích, các trường còn lại thì không!**

Elasticsearch chỉ phân tíchchữcác trường trước khi lưu trữ chúng trong các chỉ mục đảo ngược tương ứng của chúng. Không có kiểu dữ liệu nào khác trải qua phân tích văn bản. Elasticsearch cũng sử dụng cùng một nguyên tắc phân tích các trường văn bản của truy vấn khi thực hiện truy vấn tìm kiếm.

Người dùng thường không muốn nhập một chuỗi tìm kiếm chính xác—họ có thể bỏ qua một số từ thông dụng, đảo ngược thứ tự, bỏ qua cách viết đúng, v.v. Sức mạnh của một công cụ tìm kiếm như Elasticsearch đến từ thực tế là nó có thể tìm kiếm không chỉ các từ riêng lẻ mà còn cả các từ đồng nghĩa, từ viết tắt, từ gốc, v.v. Chúng ta có thể đính kèm các trình phân tích khác nhau—bao gồm trình phân tích chuẩn, ngôn ngữ và tùy chỉnh—dựa trên các yêu cầu của mình.

Phân tích dữ liệu trong quá trình lập chỉ mục và nắm bắt dữ liệu dựa trên các yêu cầu của chúng tôi cho phép Elasticsearch đáp ứng phổ các biến thể truy vấn tìm kiếm. Một trình phân tích bao gồm một số thành phần khác giúp phân tích văn bản.

#### Mô-đun phân tích

Analyzer là một mô-đun phần mềm có hai chức năng: tokenization và normalization. Elasticsearch sử dụng các quy trình này để phân tích các trường văn bản và lưu trữ trong các chỉ mục đảo ngược để khớp truy vấn. Hãy cùng xem xét các khái niệm này ở cấp độ cao trước khi đi sâu vào giải phẫu của analyzer.

* + 1. Mã hóa

Tokenization, như tên gọi của nó, là một quá trình chia câu thành các từ riêng lẻ bằng cách tuân theo các quy tắc nhất định. Ví dụ, chúng ta có thể hướng dẫn quá trình này chia câu tại một dấu phân cách, chẳng hạn như khoảng trắng, chữ cái, mẫu hoặc các tiêu chí khác.

Quá trình này được thực hiện bởi một thành phần gọi là tokenizer, có nhiệm vụ duy nhất là cắt câu thành các từ riêng lẻ gọi là token bằng cách tuân theo các quy tắc nhất định. Một tokenizer khoảng trắng thường được sử dụng trong quá trình tokenization: nó tách từng từ trong câu bằng khoảng trắng và xóa bất kỳ dấu câu và các ký tự không phải ký tự nào khác.

Từ cũng có thể được phân chia dựa trên các ký tự không phải chữ cái, dấu hai chấm hoặc các dấu phân cách tùy chỉnh khác. Ví dụ, đánh giá của một nhà phê bình phim nói rằng "Bộ phim thật tuyệt!!! Thật buồn cười :)

:)” có thể được chia thành các từ riêng lẻ: The, movie, was, sick, Hilarious, v.v. (lưu ý rằng các từ vẫn chưa được viết thường). Tương tự như vậy, pickled-peppers có thể được mã hóa thành pickled và peppers, K8s có thể được mã hóa thành K và s, v.v.

***7.2 Bộ phân tíchcác mô-đun* 239**

Mặc dù điều này giúp tìm kiếm theo từ (riêng lẻ hoặc kết hợp), nhưng nó chỉ có thể đi xa đến mức trả lời các truy vấn như truy vấn có từ đồng nghĩa, số nhiều và các yếu tố khác đã đề cập trước đó. Quá trình chuẩn hóa đưa phân tích từ đây đến giai đoạn tiếp theo.

* + 1. Chuẩn hóa

Trong quá trình chuẩn hóa, các mã thông báo (từ) được xoa bóp, chuyển đổi, sửa đổi và làm giàu bằng cách sử dụng từ gốc, từ đồng nghĩa và từ dừng. Các tính năng này được thêm vào quy trình phân tích để đảm bảo rằng dữ liệu được lưu trữ phù hợp cho mục đích tìm kiếm. Một trong những tính năng như vậy là từ gốc: một hoạt động trong đó các từ được rút gọn (từ gốc) thành từ gốc của chúng. Ví dụ, tác giả là từ gốc của tác giả, tác giả- ing và tác giả. Ngoài từ gốc, chuẩn hóa còn xử lý việc tìm các từ đồng nghĩa phù hợp trước khi thêm chúng vào chỉ mục đảo ngược. Ví dụ, tác giả có các từ đồng nghĩa như wordsmith, tiểu thuyết gia và nhà văn. Và cuối cùng, mỗi tài liệu sẽ chứa các từ như a, an, and, is, but và the được gọi là từ dừng vì chúng

không có chỗ trong việc tìm kiếm các tài liệu có liên quan.

Cả tokenization và chuẩn hóa đều được thực hiện bởi mô-đun phân tích. Một máy phân tích thực hiện điều này bằng cách sử dụng bộ lọc và một tokenizer. Chúng ta hãy phân tích mô-đun phân tích và xem nó được tạo thành từ những gì.

* + 1. Giải phẫu của một máy phân tích

Tokenization và chuẩn hóa được thực hiện bởi ba thành phần phần mềm—character filters, tokenizers và token filters—về cơ bản hoạt động cùng nhau như một mô-đun phân tích. Như hình 7.1 cho thấy, một mô-đun phân tích bao gồm một tập hợp các bộ lọc và một tokenizer. Các bộ lọc hoạt động trên văn bản thô (bộ lọc ký tự) và văn bản được token hóa (bộ lọc token). Nhiệm vụ của tokenizer là chia câu thành các từ riêng lẻ (token).

**Các trường văn bản thô được đưa qua bộ phân tích.**



**Các mã thông báo được chuẩn hóa sẽ được máy phân tích đưa ra.**

Tính cáchbộ lọc

Mã thông báo

Bộ lọc mã thông báo

**Bộ lọc ký tự (tùy chọn: không hoặc nhiều)**

**Tokenizer (bắt buộc: chỉ có một)**

**Bộ lọc mã thông báo**

**(tùy chọn: không hoặc nhiều)**

**Hình 7.1 Giải phẫu của một mô-đun phân tích**

**240 CPHẦN7*Phân tích văn bản***

Tất cả các trường văn bản đều đi qua đường ống này: văn bản thô được làm sạch bằng bộ lọc ký tự và văn bản kết quả được chuyển đến bộ phân tích mã thông báo. Bộ phân tích mã thông báo chia văn bản thành các mã thông báo (các từ riêng lẻ). Sau đó, các mã thông báo sẽ đi qua bộ lọc mã thông báo, tại đó chúng được sửa đổi, làm giàu và tăng cường. Các mã thông báo đã hoàn thiện được lưu trữ trong các chỉ mục đảo ngược thích hợp. Các truy vấn tìm kiếm được phân tích theo cùng một cách.

Hình 7.2 cho thấy một ví dụ về quá trình phân tích. Chúng ta đã thấy hình này trong chương 3 và nó được trình bày lại ở đây để hoàn thiện.

Dảiđánh dấu HTML (html\_dảibộ lọc ký tự)

<h2>Xin chào THẾ GIỚI</h2>

Phân biệt các từ (khoảng trắngbộ phân tích dữ liệu)

Xin chào THẾ GIỚI

Chữ thườngcác mã thông báo (chữ thường

|  |  |
| --- | --- |
| [Xin chào] [THẾ GIỚI] | bộ lọc mã thông báo) |
|  |
|  | |

[xin chào] [thế giới]

**Hình 7.2 Một ví dụ về phân tích văn bản trong thực tế**

Như chúng tôi đã đề cập, một máy phân tích bao gồm ba khối xây dựng cấp thấp:

* *Bộ lọc ký tự*—Áp dụng ở cấp độ ký tự. Mỗi ký tự của văn bản đều đi qua các bộ lọc này. Nhiệm vụ của bộ lọc là xóa các ký tự không mong muốn khỏi chuỗi văn bản. Ví dụ, quy trình này có thể xóa các thẻ HTML như<h1>,<href>, Và<nguồn>từ văn bản đầu vào. Nó cũng giúp thay thế văn bản bằng văn bản khác (ví dụ, chữ cái Hy Lạp bằng các từ tiếng Anh tương đương) hoặc khớp văn bản trong biểu thức chính quy (regex) và thay thế bằng biểu thức tương đương (ví dụ, khớp email dựa trên biểu thức chính quy và trích xuất tên miền của tổ chức). Bộ lọc ký tự là tùy chọn; trình phân tích có thể tồn tại mà không cần bộ lọc ký tự. Elasticsearch cung cấp ba bộ lọc ký tự ngay lập tức:html\_dải,lập bản đồ, Vàmẫu\_thay\_thế.
* *Bộ phân tích mã thông báo*-Tách rachữcác trường thành các từ bằng cách sử dụng một dấu phân cách như khoảng trắng, dấu câu hoặc ranh giới từ. Mỗi trình phân tích phải có một và chỉ một trình phân tích mã thông báo. Elasticsearch cung cấp một số trình phân tích mã thông báo để giúp phân chia văn bản đến thành các mã thông báo riêng lẻ sau đó được đưa qua các bộ lọc mã thông báo để chuẩn hóa thêm. Elasticsearch sử dụngtiêu chuẩntheo mặc định, nó sẽ phân tách các từ dựa trên ngữ pháp và dấu câu.
  1. ***Máy phân tíchcác mô-đun* 241**
  + *Bộ lọc mã thông báo*—Thực hiện xử lý thêm trên các mã thông báo do các trình phân tích tạo ra. Ví dụ, bộ lọc mã thông báo có thể thay đổi trường hợp, tạo từ đồng nghĩa, cung cấp từ gốc (gốc), tạo n-gram và shingles, v.v. Bộ lọc mã thông báo là tùy chọn. Có thể có số không, một hoặc nhiều liên kết với một mô-đun phân tích. Elasticsearch cung cấp một danh sách dài các bộ lọc mã thông báo ngay khi xuất xưởng.

Bộ lọc ký tự và mã thông báo là tùy chọn, nhưng chúng ta phải có một bộ mã thông báo. Chúng ta sẽ xem xét các thành phần này chi tiết hơn ở phần sau của chương, nhưng trước tiên hãy tìm một API để giúp kiểm tra các trình phân tích trước khi đưa chúng vào sản xuất.

* + 1. Kiểm tra máy phân tích

Bạn có thể tò mò về cách Elasticsearch chia nhỏ văn bản, sửa đổi và sau đó xử lý nó. Xét cho cùng, việc biết cách chia nhỏ và cải thiện văn bản ngay từ đầu giúp chúng ta chọn trình phân tích phù hợp và tùy chỉnh nếu cần.

Elasticsearch hiển thị một điểm cuối chỉ để thử nghiệm quá trình phân tích văn bản:

Điểm cuối \_analyze giúp chúng ta hiểu chi tiết về quy trình. API tiện dụng này cho phép chúng ta kiểm tra cách công cụ xử lý văn bản trong quá trình lập chỉ mục. Có thể dễ dàng hiểu bằng ví dụ.

**Điểm cuối \_analyze**

Các\_phân tíchđiểm cuối giúp ích rất nhiều trong việc hiểu cách văn bản được xử lý và lập chỉ mục bởi công cụ tìm kiếm, cũng như lý do tại sao truy vấn tìm kiếm có thể không tạo ra kết quả mong muốn. Chúng tôicó thể sử dụng điều này như bước đầu tiên để kiểm tra văn bản của chúng tôi với các trình phân tích dự kiến ​​trước khi đưa mã vào sản xuất.

Giả sử chúng ta muốn xác định Elasticsearch sẽ xử lý đoạn văn bản này như thế nào khi nó được lập chỉ mục: “James Bond 007”.

**Liệt kê 7.1 Kiểm tra máy phân tích bằng cách sử dụng\_phân tíchđiểm cuối**

NHẬN \_phân tích

{

"văn bản": "James Bond 007"

}

Thực thi tập lệnh này sẽ tạo ra tập hợp các mã thông báo được hiển thị trong hình 7.3. Đầu ra của truy vấn cho chúng ta thấy cách trình phân tích xử lý trường văn bản. Trong trường hợp này, trường được chia thành ba mã thông báo (james, bond và 007), tất cả đều là chữ thường. Bởi vì chúng ta không chỉ định một ana-lyzer trong mã, theo mặc định, nó được coi là trình phân tích chuẩn. Mỗi mã thông báo có một kiểu: ALPHANUM cho chuỗi, NUM cho mã thông báo số, v.v. Vị trí của mã thông báo cũng được lưu, như bạn có thể thấy trong kết quả. Điều đó đưa chúng ta đến điểm tiếp theo: chỉ định trình phân tích một cách rõ ràng trong quá trình kiểm tra \_analyze.

**242 CPHẦN7*Phân tích văn bản***

* **Cáctiêu chuẩntrình phân tích được sử dụng theo mặc định vì nó không được chỉ định trong truy vấn.**
* **Văn bản đầu vào được chia thành một tập hợp các mã thông báo. Có ba mã thông báo trong trường hợp này: “james”, “bond” và “007”.**
* **Elasticsearch suy ra loại được liên kết với mỗi mã thông báo:**

**CHỮ CÁIcho chữ và số vàNUMcho số.**

* **bắt đầu\_bù trừVàđộ lệch cuốichỉ ra độ lệch ký tự bắt đầu và kết thúc của từ.**
* **Tất cả các mã thông báo đều được viết thường.**

**Hình 7.3 Các mã thông báo được tạo ra bằng cách gọi\_phân tíchđiểm cuối**

**VÀ****KIỂM TRA PHÂN TÍCH XPLICIT**

Trong danh sách 7.1, chúng tôi không đề cập đến bộ phân tích, mặc dù công cụ đã áp dụngtrình phân tích chuẩn theo mặc định. Tuy nhiên, chúng ta cũng có thể bật trình phân tích một cách rõ ràng. Mã trong danh sách sau đây bật trình phân tích đơn giản.

**Liệt kê 7.2 Bật một trình phân tích một cách rõ ràng**

NHẬN \_phân tích

{

"text": "James Bond 007", "analyzer": "đơn giản"

}

Bộ phân tích đơn giản (chúng ta sẽ tìm hiểu về nhiều loại bộ phân tích khác nhau trong phần tiếp theo) cắt bớt văn bản khi gặp phải ký tự không phải chữ cái. Vì vậy, mã này chỉ tạo ra hai mã thông báo: “james” và “bond” (“007” bị cắt bớt), trái ngược với ba mã thông báođược tạo ra bởi tập lệnh trước đó sử dụng trình phân tích tiêu chuẩn.

Nếu bạn tò mò, hãy đổi trình phân tích sang tiếng Anh. Các mã thông báo đầu ra sau đó là “jame”, “bond” và “007”. Điểm đáng chú ý là “james” được chuyển thành “jame”khi trình phân tích tiếng Anh được áp dụng. (Chúng tôi thảo luận về trình phân tích tiếng Anh trong phần 7.3.7.)

**CMÁY PHÂN TÍCH ONFIGURING ON FLY**

Chúng ta cũng có thể sử dụngAPI \_analyze để kết hợp và khớp các bộ lọc và trình phân tích, về cơ bản là tạo ra một trình phân tích tùy chỉnh ngay lập tức (chúng tôi không thực sự xây dựng hoặc phát triển một trình phân tích mới). Một trình phân tích tùy chỉnh theo yêu cầu được hiển thị trong mã sau.

* 1. ***Tích hợp sẵnmáy phân tích* 243**

**Liệt kê 7.3 Tạo trình phân tích tùy chỉnh**

NHẬN \_phân tích

{

"tokenizer": "path\_hierarchy", "filter": ["chữ hoa"], "text": "/Volumes/FILES/Dev"

}

Mã này sử dụng một trình phân tích path\_hierarchy với bộ lọc chữ hoa và tạo ra ba mã thông báo từ văn bản đầu vào đã cho: “/VOLUMES”, “/VOLUMES/FILES” và “/VOLUMES/FILES/DEV”. Trình phân tích path\_hierarchy chia tách văn bản dựa trên một trình phân tách đường dẫn; do đó, ba mã thông báo cho chúng ta biết về ba thư mục trong hệ thống phân cấp.

Chúng ta đã nói về các mô-đun phân tích trong vài phần gần đây. Elasticsearch cung cấp một số mô-đun như vậy; hãy cùng xem xét chi tiết các mô-đun phân tích tích hợp này.

#### 7.3 Tích hợp sẵn máy phân tích

Elasticsearch cung cấp tám trình phân tích có sẵn mà chúng ta có thể sử dụng trong giai đoạn phân tích văn bản. Các trình phân tích này thường đủ cho các trường hợp cơ bản, nhưng nếu chúng ta cần tạo một trình phân tích tùy chỉnh, chúng ta có thể thực hiện bằng cách khởi tạo một mô-đun trình phân tích mới với các thành phần cần thiết. Bảng 7.3 liệt kê các trình phân tích Elasticsearch.

**Bảng 7.3 Bộ phân tích tích hợp**

|  |  |
| --- | --- |
| **Máy phân tích** | **Sự miêu tả** |
| tiêu chuẩn | Bộ phân tích mặc định mã hóanhập văn bản dựa trên ngữ pháp, dấu câu và khoảng trắng. Các mã thông báo đầu ra được viết thường. |
| đơn giản | Tách văn bản đầu vào thành bất kỳ ký tự nào không phải chữ cái, chẳng hạn như khoảng trắng, dấu gạch ngang và số. Không giống nhưtiêu chuẩnmáy phân tích,đơn giảnmáy phân tích cũng thấp hơntrường hợp các mã thông báo đầu ra. |
| dừng lại | MỘTđơn giảnbộ phân tích với các từ dừng tiếng Anh được bật theo mặc định |
| khoảng trắng | Phân tách văn bản đầu vào dựa trên các dấu phân cách khoảng trắng |
| từ khóa | Không làm thay đổi văn bản đầu vào. Giá trị của trường được lưu trữ nguyên trạng. |
| ngôn ngữ | Giúp làm việc với ngôn ngữ của con người, như tên gọi của nó. Elasticsearch cung cấp hàng chục trình phân tích cho nhiều ngôn ngữ khác nhau như tiếng Anh, tiếng Tây Ban Nha, tiếng Pháp, tiếng Nga, tiếng Hindi, v.v. |
| mẫu | Phân chia các mã thông báo dựa trên biểu thức chính quy (regex). Theo mặc định, tất cả các ký tự không phải từ đều giúp phân chia câu thành các mã thông báo. |
| dấu vân tay | Sắp xếp các mã thông báo và loại bỏ các mã trùng lặp để tạo ra một mã thông báo được nối tiếp duy nhất |

Trình phân tích chuẩn là mặc định và được sử dụng rộng rãi trong quá trình phân tích văn bản. Hãy cùng xemtại máy phân tích tiêu chuẩn với ví dụ ở phần tiếp theo, sau đó chúng ta sẽ lần lượt xem xét các máy phân tích khác.

**244 CPHẦN7*Phân tích văn bản***

**GHI CHÚ** Elasticsearch cho phép chúng ta tạo ra vô số các bộ phân tích tùy chỉnh bằng cách kết hợp và khớp các bộ lọc và bộ mã hóa. Sẽ quá dài dòng và không thực tế nếu đi sâu vào từng bộ lọc và bộ mã hóa, nhưng tôi sẽ trình bày càng nhiều ví dụ càng tốt trong chương này. Tham khảo tài liệu chính thức để biết các thành phần cụ thể và cách tích hợp chúng vào ứng dụng của bạn. Tôi cũng đã thêm nhiều ví dụ hơn vào mã nguồn của cuốn sách mà bạn có thể sử dụng để thử nghiệm với các bộ phân tích.

* + 1. Tiêu chuẩnmáy phân tích

Trình phân tích chuẩn là trình phân tích mặc định được sử dụng trong Elasticsearch. Công việc của nó là mã hóa các câu dựa trên khoảng trắng, dấu câu và ngữ pháp. Giả sử chúng ta muốn xây dựng một chỉ mục với sự kết hợp kỳ lạ giữa đồ ăn nhẹ và đồ uống. Hãy xem xét văn bản sau đây có đề cập đến cà phê với bỏng ngô:

Tách nóng và một là một tổ hợp kỳ lạ :(!! Chúng ta có thể lập chỉ mục văn bản này vào một chỉ mục weird\_combos:

ĐĂNG weird\_combos/\_doc

{

"text": "Cốc nóngcủavà Một là một sự kết hợp kỳ lạ :(!!"

}

Văn bản được phân chia thành các mã thông báo và danh sách các mã thông báo được chia nhỏ (hiển thị ở đây dưới dạng tóm tắt):

["nóng", "cốc", "của", "", "và", "a", """""", "là", "một", "kỳ lạ", "kết hợp"]

Các mã thông báo được viết thường, như bạn có thể thấy trong đầu ra. Trình phân tích mã thông báo chuẩn sẽ xóa biểu tượng mặt cười ở cuối và các dấu chấm than, nhưng các biểu tượng cảm xúc được lưu như thể chúng là thông tin văn bản. Đây là hành vi mặc định của trình phân tích chuẩn, phân tích các từ dựa trên khoảng trắng và loại bỏ các ký tự không phải chữ cái như dấu câu. Hình 7.4 cho thấy hoạt động của văn bản đầu vào ví dụ này khi được truyền qua trình phân tích.



"Cốc nóngcủavà a là một sự kết hợp kỳ lạ :(!!"

Bộ lọc ký tự

Mã thông báo

"Nóng", "cốc", "của", " ", "và", "một", " ", "là","một" "Lạ" "Kết hợp"

Bộ lọc mã thông báo

"nóng", "tách", "của", " ", "và","Một", ,"là", "một" "kỳ lạ" "combo"



**Không có bộ lọc ký tự, vì vậy văn bản sẽ đi qua như vậy**

**Cáctiêu chuẩntokenizer chia câu thành các từ dựa trên khoảng trắng và dấu chấm câu.**

**Các mã thông báo được viết thường bằng**

" "

**chữ thườngbộ lọc mã thông báo.**

**Hình 7.4 Cáctiêu chuẩn(mặc định) máy phân tích đang hoạt động**

***7.3 Tích hợp sẵnmáy phân tích* 245**

Tiếp theo, hãy xem cách chúng ta có thể sử dụng API \_analyze để kiểm tra đầu ra trước khi lập chỉ mục văn bản (một lần nữa, chúng ta không chỉ định trình phân tích vì Elasticsearch sử dụng trình phân tích tiêu chuẩn nếu chúng ta không đề cập rõ ràng đến trình phân tích):

NHẬN \_phân tích

{

"text": "Cốc nóngcủavà Một là một sự kết hợp kỳ lạ :(!!"

}

Đầu ra của lệnh GET như sau (ngoại trừ mã thông báo đầu tiên, phần còn lại được tóm tắt lại cho ngắn gọn):

{

"mã thông báo" : [

{

**"token" : "nóng",**

"start\_offset" : 0,

"end\_offset" : 3, "type" : "<ALPHANUM>",

"vị trí" : 0

},

{ "token" : "cốc", ... },

{ "token" : "của", ... },

{ "mã thông báo" : "**"**, ... },

{ "token" : "và", ... },

{ "mã thông báo" : "a", ... },

**Bộ lọc mã thông báo chữ thường sẽ chuyển các từ thành chữ thường.**

**Các từ dừng không bị xóa vì bộ lọc dừng bị tắt theo mặc định.**

**Tách cà phê được lập chỉ mục theo nguyên trạng—chiều rộng là một ký tự.**

{ "mã thông báo" : """**"""**, ... },

{ "token" : "là", ... },

{ "mã thông báo" : "a", ... },

{ "token" : "lạ", ... },

{ "mã thông báo" : "kết hợp", ... }

]

}

**Biểu tượng cảm xúc bỏng ngô được lập chỉ mục theo nguyên trạng—chiều rộng là hai ký tự.**

**Biểu tượng mặt cười và dấu chấm than sẽ bị loại bỏ bởi công cụ phân tích chuẩn.**

Các từ được chia dựa trên khoảng trắng và các ký tự không phải chữ cái (dấu câu), đây là đặc điểm của trình phân tích cú pháp chuẩn. Sau đó, các mã thông báo được chuyển qua bộ lọc mã thông báo chữ thường. Hình 7.5 hiển thị đầu ra cô đọng của lệnh này trong Dev Tools.

**Các thành phần của một máy phân tích tích hợp**

Như đã thảo luận, mỗi trình phân tích tích hợp đều đi kèm với một bộ thành phần được xác định trước như bộ lọc ký tự, bộ phân tích mã thông báo và bộ lọc mã thông báo—ví dụ:dấu vân taymáy phân tích được cấu thành từ mộttiêu chuẩnbộ mã hóavà một số bộ lọc mã thông báo (dấu vân tay,chữ thường,gấp lại, Vàdừng lại) nhưng không có bộ lọc ký tự. Nó không phải làdễ dàng để biết cấu tạo của một máy phân tíchtrừ khi bạn ghi nhớ chúng theo thời gian! Vì vậy, lời khuyên của tôi là hãy kiểm tra định nghĩa trên trang tài liệu nếu bạn cần biết các chi tiết cụ thể của một trình phân tích.

**246 CPHẦN7*Phân tích văn bản***

**Tất cả văn bản đều được phân cách bằng khoảng trắng.**



**Tất cả văn bản đều được viết thường.**

**Biểu tượng cảm xúc cà phê và bỏng ngô được lưu nguyên trạng.**

**Các dấu chấm than (!!) và**

**biểu tượng mặt cười (:)) đã bị xóa.**

**MỘTtiêu chuẩnbộ phân tích mã hóa các từ trên khoảng trắng phân cách, loại bỏ**

**không phải chữ cái và viết thường các ký hiệu.**

**Hình 7.5 Các mã thông báo đầu ra từ mộttiêu chuẩnmáy phân tích**

Như bạn có thể thấy, các token cho “coffee” và “popcorn” được lưu trữ nguyên trạng và các ký tự không phải chữ cái như “:(“ và “!!” bị xóa. Token “coffee” có chiều rộng là một ký tự (kiểm tra các offset) và token “popcorn” được lưu trữ rộng là hai ký tự.

**T****ĐÁNH GIÁ MÁY PHÂN TÍCH CHUẨN**

Chúng ta có thể thêm một trình phân tích cụ thể trong giai đoạn thử nghiệm phân tích văn bản của mình bằng cách thêm một

máy phân tíchthuộc tính trong mã. Danh sách sau đây minh họa điều này.

**Liệt kê 7.4 Kiểm tratiêu chuẩnmáy phân tích với lệnh gọi rõ ràng**



NHẬN \_phân tích

{

"máy phân tích": "chuẩn",

**Chỉ định trình phân tích (không thực sự cần thiết ở đây vì tiêu chuẩn là trình phân tích mặc định)**

"text": "Cốc nóngcủavà ai là một sự kết hợp kỳ lạ :(!!"

}

Mã này tạo ra kết quả được hiển thị trong hình 7.5. Chúng ta có thể thay thế giá trị củabộ phân tích với một bộ phân tích đã chọn nếu chúng ta đang kiểm tra trường văn bản bằng một bộ phân tích khác: ví dụ: "analyzer": "whitespace".

Đầu ra cho thấy văn bản đã được phân tách thành ký tự và viết thường.

Hình 7.6 minh họa bộ phân tích chuẩn với các thành phần bên trong và giải phẫu của nó. Nó bao gồm một bộ phân tích chuẩn và hai bộ lọc mã thông báo: chữ thường và dừng. Không có bộ lọc ký tự nào được xác định trên bộ phân tích chuẩn. Hãy nhớ rằng, bộ phân tích bao gồm không hoặc nhiều bộ lọc ký tự, ít nhất một bộ phân tích và không hoặc nhiều bộ lọc mã thông báo.

***7.3 Tích hợp sẵnmáy phân tích* 247**

**Không có bộ lọc ký tự**

**dừng lạibộ lọc bị vô hiệu hóa theo mặc định**

**Hình 7.6 Giải phẫu của mộttiêu chuẩnmáy phân tích**

Bộ lọc ký tự

Mã thông báo

Bộ lọc mã thông báo

tiêu chuẩn

máy phân tích

dừng lại

lọc

chữ thường

lọc

tiêu chuẩn

t máy đánh dấu

Mặc dù trình phân tích chuẩn được liên kết với bộ lọc mã thông báo từ dừng (như thể hiện trong hình 7.6), bộ lọc bị vô hiệu hóa theo mặc định. Chúng ta có thể bật nó bằng cách cấu hình các thuộc tính của nó, như đã thảo luận trong phần tiếp theo.

**CHÌNH DẠNG MÁY PHÂN TÍCH CHUẨN**

Elasticsearch cho phép chúng ta cấu hình một số tham số như bộ lọc stop words, đường dẫn stop words và độ dài mã thông báo tối đa trên trình phân tích chuẩn, do đó tùy chỉnh trình phân tích. Cách cấu hình các thuộc tính là thông qua cài đặt chỉ mục. Khi chúng ta tạo một chỉ mục, chúng ta có thể cấu hình trình phân tích thông qua thành phần cài đặt:

ĐẶT <my\_index>

{

"cài đặt": {

"Phân tích": {

"máy phân tích": {

...

**Đối tượng phân tích**

**thiết lập bộ phân tích.**

**Bộ phân tích chỉ số này được liên kết với**

}

}

}

}

Trong vài phần tiếp theo, chúng ta sẽ tìm hiểu cơ chế tùy chỉnh trình phân tích này.

**CTỪ DỪNG ĐẶC BIỆT**

Hãy xem xét ví dụ về việc bật từ dừng tiếng Anh trên trình phân tích chuẩn. Chúng ta có thể thực hiện việc này bằng cách thêm bộ lọc trong quá trình tạo chỉ mục, như danh sách sau đây cho thấy.

**Liệt kê 7.5tiêu chuẩnbộ phân tích có bộ lọc dừng từ được bật**

ĐẶT my\_index\_with\_stopwords

{

"cài đặt": {

"Phân tích": {

"máy phân tích": {

**Thiết lập bộ phân tích**

**chochỉ số Têncái**

"chuẩn\_với\_từ\_dừng":{

**máy phân tích**

**248 CPHẦN7*Phân tích văn bản***

**Loại máy phân tích tiêu chuẩn**

}

"loại":"chuẩn", "từ dừng":"\_tiếng Anh\_"

}

}

}

**Cho phép tiếng Anhdừng từ lọc**

}

Như chúng ta đã thấy trước đó, bộ lọc từ dừng trên trình phân tích chuẩn bị vô hiệu hóa theo mặc định. Bây giờ chúng ta đã tạo chỉ mục với trình phân tích chuẩn được cấu hình với từ dừng, bất kỳ văn bản nào được lập chỉ mục đều đi qua trình phân tích đã sửa đổi này. Để kiểm tra điều này, chúng ta có thể gọi điểm cuối \_analyze trên chỉ mục.

**Liệt kê 7.6tiêu chuẩnmáy phân tích đang hoạt động với các từ dừng**

ĐĂNG my\_index\_with\_stopwords/\_analyze

{

"text": ["Cốc nóngcủa và trở lại một sự kết hợp kỳ lạ :(!!"],

**Gọi API \_analyze trên chỉ mục**

"máy phân tích": "standard\_with\_stopwords"

}

**Bộ phân tích chúng tôi đã tạo trong danh sách 7.5**

Đầu ra của cuộc gọi này cho thấy các từ dừng phổ biến (tiếng Anh) như of, a và is

đã được gỡ bỏ:

["nóng", "cốc", "" "","kỳ lạ", "kết hợp"]

Chúng ta có thể thay đổi các từ dừng cho ngôn ngữ mà chúng ta lựa chọn. Ví dụ, mã trong danh sách tiếp theo hiển thị chỉ mục với các từ dừng tiếng Hindi và trình phân tích chuẩn.

**Liệt kê 7.7 Trình phân tích chuẩn cho phép dừng từ tiếng Hindi**

ĐẶT my\_index\_with\_stopwords\_hindi

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "standard\_with\_stopwords\_hindi":{

"loại":"chuẩn", "từ dừng":"\_hindi\_"

}

}

}

}

}

Chúng ta có thể kiểm tra văn bản bằng cách sử dụngchuẩn\_với\_từ\_dừng\_tiếng\_hindimáy phân tích:

ĐĂNG my\_index\_with\_stopwords\_hindi/\_analyze

{

"chữ": ["?"],

"máy phân tích": "standard\_with\_stopwords\_hindi"

}

***7.3 Tích hợp sẵnmáy phân tích* 249**

Nếu bạn tò mò, câu tiếng Hindi trong ví dụ này có thể dịch là "bạn đang làm gì?"

Đầu ra của tập lệnh được hiển thị ở đây:

"mã thông báo" : [{

"mã thông báo" : "","start\_offset" : 3,

"end\_offset" : 7, "type" : "<ALPHANUM>",

"vị trí" : 1

}]

Mã thông báo duy nhất được đưa ra là (từ thứ hai) vì các từ còn lại là từ dừng. (Chúng phổ biến trong tiếng Hindi.)

**FVỚI****-TỪ DỪNG DỰA TRÊN**

Trong các ví dụ trước, chúng tôi đã cung cấp cho trình phân tích một manh mối về việc nên sử dụng stop word nào (tiếng Anh hoặc tiếng Hindi) bằng cách chỉ định một bộ lọc có sẵn. Nếu các bộ lọc stop word tích hợp không đáp ứng được yêu cầu của chúng tôi, chúng tôi có thể cung cấp stop word thông qua một tệp rõ ràng.

Giả sử chúng ta không muốn người dùng nhập những từ tục tĩu vào ứng dụng của chúng ta. Chúng ta có thể tạo một tệp với tất cả các từ tục tĩu bị cấm và thêm đường dẫn của tệp làm tham số.eter đến trình phân tích chuẩn. Đường dẫn tệp phải được cung cấp theo thư mục cấu hình của cài đặt Elasticsearch. Danh sách tiếp theo tạo chỉ mục với trình phân tích chấp nhận tệp stop-word.

**Liệt kê 7.8 Tạo chỉ mục với các từ dừng tùy chỉnh trongtài liệu**

PUT index\_with\_swear\_stopwords

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích lời thề":{

**Đặt tên cho trình phân tích để có thể tham chiếu khi lập chỉ mục/kiểm tra**

**Sử dụng tiêu chuẩn**

**máy phân tích** }

}

}

}

"loại":"chuẩn", "stopwords\_path":"swearwords.txt"

**Vị trí tệp phải liên quan đến thư mục cấu hình.**

}

Thuộc tính stopwords\_path tìm kiếm một tệp (swearwords.txt, trong trường hợp này) trong một thư mục bên trong thư mục cấu hình của Elasticsearch. Danh sách sau đây tạo tệp trong thư mục cấu hình. Hãy đảm bảo bạn thay đổi vào thư mục $ELASTICSEARCH\_HOME/config và tạo tệp swearwords.txt tại đó. Lưu ý rằng mỗi từ bị liệt kê đen sẽ nằm trên một dòng mới.

**250 CPHẦN7*Phân tích văn bản***

**Liệt kê 7.9 Các từ bị cấm trong tệp văn bản preachwords.txt**

đồ khốn nạn chết tiệt

cái quái gì tệ thế

Sau khi tệp được tạo và chỉ mục được phát triển, chúng ta đã sẵn sàng sử dụng trình phân tích với các từ chửi thề tùy chỉnh của mình.

**Liệt kê 7.10 Đưa những lời chửi thề tùy chỉnh vàocông việc**

POST index\_with\_swear\_stopwords/\_analyze

{

"text": ["Chết tiệt, tệ quá!"], "analyzer": "swearwords\_analyzer"

}

Mã này sẽ ngăn không cho các từ đầu tiên và cuối cùng đi qua quá trình lập chỉ mục vì chúng nằm trong danh sách đen các từ chửi thề của chúng tôi. Thuộc tính tiếp theo chúng ta có thể cấu hình là độ dài của các mã thông báo: độ dài của chúng khi xuất ra.

**CĐỊNH DẠNG CHIỀU DÀI TOKEN**

Chúng ta có thể cấu hình độ dài mã thông báo tối đa để chia mã thông báo dựa trên độ dài mà chúng ta yêu cầu. Ví dụ, danh sách 7.11 tạo một chỉ mục với một trình phân tích chuẩn. Trình phân tích được cấu hình để có độ dài mã thông báo tối đa là bảy ký tự. Nếu chúng ta cung cấp một từ dài 13 ký tự, nó sẽ được chia thành hai mã thông báo gồm bảy và sáu ký tự (ví dụ: “Elasticsearch” sẽ trở thành “Elastic”, “search”).

**Liệt kê 7.11 Tạo một trình phân tích với độ dài mã thông báo tùy chỉnh**

PUT my\_index\_with\_max\_token\_length

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "standard\_max\_token\_length":{

"kiểu":"chuẩn", "chiều dài mã thông báo tối đa":7

}

}

}

}

}

Cho đến nay, chúng ta đã làm việc với trình phân tích chuẩn. Tiếp theo trong danh sách các trình phân tích tích hợp là trình phân tích đơn giản, có mục đích duy nhất là tách văn bản thành các ký tự không phải chữ cái. Hãy cùng thảo luận về chi tiết sử dụng trình phân tích đơn giản.

***7.3 Tích hợp sẵnmáy phân tích* 251**

* + 1. Bộ phân tích đơn giản

Trong khi trình phân tích chuẩn chia văn bản thành các mã thông báo khi gặp khoảng trắng hoặc dấu câu, trình phân tích đơn giản sẽ mã hóa các câu khi xuất hiện một ký tự không phải chữ cái như số, khoảng trắng, dấu nháy đơn hoặc dấu gạch nối. Nó thực hiện điều này bằng cách sử dụng trình mã hóa chữ thường, không liên kết với bất kỳ bộ lọc ký tự hoặc mã thông báo nào (xem hình 7.7).

**Không có ký tựbộ lọcKhông bộ lọc mã thông báo**

Bộ lọc ký tự

Mã thông báo

Bộ lọc mã thông báo

đơn giản

máy phân tích

chữ thường

bộ mã hóa

**Hình 7.7 Giải phẫu của mộtđơn giảnmáy phân tích**

Hãy cùng xem xét một ví dụ về việc lập chỉ mục cho văn bản “Lukša's K8s in Action”.

**Liệt kê 7.12 Phân tích văn bản bằng cách sử dụngđơn giảnmáy phân tích**

POST \_phân tích

{

"text": ["K8s của Lukša đang hoạt động"], "analyzer": "simple"

}

Kết quả thu được như sau:

["lukša","s","k","s","in","action"]

Các mã thông báo được tách ra khi gặp dấu nháy đơn (“Lukša's” trở thành “Lukša” và “s”) hoặc số (“K8s” trở thành “k” và “s”) và các mã thông báo kết quả được viết thường.

Một trình phân tích đơn giản không thể thực hiện nhiều cấu hình, nhưng nếu chúng ta muốn thêm bộ lọc (ký tự hoặc mã thông báo), cách dễ nhất để thực hiện là tạo một trình phân tích tùy chỉnh vớibộ lọc bắt buộc và bộ phân tích chữ thường (bộ phân tích đơn giản có một bộ phân tích chữ thường duy nhất). Chúng ta thấy điều này trong phần 7.4.

* + 1. Bộ phân tích khoảng trắng

Như tên gọi của nó, trình phân tích khoảng trắng sẽ chia văn bản thành các mã thông báo khi gặp khoảng trắng. Không có bộ lọc ký tự hoặc mã thông báo nào trên trình phân tích này ngoại trừ một trình phân tích khoảng trắng (xem hình 7.8). Danh sách sau đây hiển thị một tập lệnh sử dụng trình phân tích khoảng trắng.

**252 CPHẦN7*Phân tích văn bản***

**Không có ký tựbộ lọcKhông bộ lọc mã thông báo**

Bộ lọc ký tự

Mã thông báo

Bộ lọc mã thông báo

khoảng trắng

máy phân tích

khoảng trắng

bộ mã hóa

**Hình 7.8 Giải phẫu củakhoảng trắngmáy phân tích**

**Liệt kê 7.13khoảng trắngtokenizer đang hoạt động**

POST \_phân tích

{

"text":"Peter\_Piper hái một giỏ ớt ngâm!!", "analyzer": "khoảng trắng"

}

Nếu chúng ta kiểm tra tập lệnh này, chúng ta sẽ nhận được tập hợp các mã thông báo sau:

["Peter\_Piper", "đã hái", "một", "mổ", "của", "ớt ngâm!!"]

Lưu ý hai điểm sau từ kết quả: đầu tiên, văn bản chỉ được mã hóa bằng khoảng trắng, không phải dấu gạch ngang, dấu gạch dưới hoặc dấu chấm câu. Thứ hai, chữ hoa được giữ nguyên—chữ viết hoa của các ký tự và từ vẫn được giữ nguyên.

Như đã đề cập trước đó, tương tự như trình phân tích đơn giản, trình phân tích khoảng trắng không được tiếp xúc với các tham số có thể cấu hình. Nếu chúng ta cần sửa đổi hành vi của trình phân tích, chúng ta có thể cần tạo một phiên bản tùy chỉnh đã sửa đổi. Chúng ta sẽ xem xét các trình phân tích tùy chỉnh ngay sau đây.

* + 1. Từ khóamáy phân tích

Như tên gọi của nó, trình phân tích từ khóa lưu trữ văn bản mà không có bất kỳ sửa đổi hoặc mã hóa nào. Nghĩa là, trình phân tích không mã hóa văn bản, văn bản cũng không trải qua bất kỳ phân tích nào nữa thông qua bộ lọc hoặc trình mã hóa. Thay vào đó, nó được lưu trữ dưới dạng chuỗi biểu diễn một loại từ khóa. Như hình 7.9 cho thấy, trình phân tích từ khóa bao gồm một trình mã hóa noop (không hoạt động) và không có bộ lọc ký tự hoặc mã thông báo.

**Không có ký tựbộ lọcKhông bộ lọc mã thông báo**

Bộ lọc ký tự

Mã thông báo

Bộ lọc mã thông báo

từ khóa

máy phân tích

Không

bộ mã hóa

**Hình 7.9 Giải phẫu củatừ khóamáy phân tích**

***7.3 Tích hợp sẵnmáy phân tích* 253**

Văn bản được truyền qua máy phân tích được chuyển đổi và lưu trữ dưới dạng từ khóa. Ví dụ,nếu chúng ta truyền “Elasticsearch in Action” qua trình phân tích từ khóa, toàn bộ chuỗi văn bản sẽ được lưu trữ nguyên trạng, không giống như cách các trình phân tích trước đó chia văn bản thành các mã thông báo.

**Liệt kê 7.14 Sử dụngtừ khóamáy phân tích**

POST \_phân tích

{

"text":"Elasticsearch trong hành động", "analyzer": "từ khóa"

}

Đầu ra của tập lệnh này được hiển thị ở đây:

"mã thông báo" : [{

"token" : "Elasticsearch đang hoạt động", "start\_offset" : 0,

"end\_offset" : 23, "type" : "word", "position" : 0

}]

Như bạn có thể thấy, chỉ có một mã thông báo được tạo ra do kết quả của việc xử lý văn bản thông quatrình phân tích từ khóa. Cũng không có chữ thường. Tuy nhiên, có một thay đổi trong cách chúng ta tìm kiếm nếu chúng ta sử dụng trình phân tích từ khóa để xử lý văn bản. Tìm kiếm một từ duy nhất sẽ không khớp với chuỗi văn bản—chúng ta phải cung cấp một kết quả khớp chính xác. Trong ví dụ của chúng ta, chúng ta phải cung cấp nhóm từ chính xác như trong câu gốc: “Elasticsearch in Action”.

* + 1. Máy phân tích dấu vân tay

Cácdấu vân taymáy phân tích loại bỏ các từ trùng lặp và các ký tự mở rộng và sắp xếp các từ theo thứ tự bảng chữ cái để tạo ra một mã thông báo duy nhất. Nó bao gồm mộttiêu chuẩnbộ mã thông báo và bốn bộ lọc mã thông báo:dấu vân tay,chữ thường,dừng lạivà gấp ASCII (gấp lại)dấu hiệubộ lọc (xem hình 7.10).

**Không có bộ lọc ký tự**

Bộ lọc ký tự

Mã thông báo

Bộ lọc mã thông báo

dấu vân tay

máy phân tích

tiêu chuẩn

t máy đánh dấu

|  |
| --- |
| chữ thường |
| gấp lại |
| dừng lại |
| dấu vân tay |

**Hình 7.10 Giải phẫu củadấu vân taymáy phân tích**

**254 CPHẦN7*Phân tích văn bản***

Ví dụ, hãy phân tích văn bản hiển thị trong danh sách tiếp theo, định nghĩa dosa—một loại bánh kếp mặn của Nam Ấn Độ. Mã phân tích mô tả về món ăn này bằng cách sử dụng máy phân tích dấu vân tay.

**Liệt kê 7.15 Phân tích văn bản bằng cách sử dụngdấu vân taymáy phân tích**

POST \_phân tích

{

"text": "Dosa là một loại bánh kếp mỏng hoặc bánh crepe có nguồn gốc từ Nam Ấn Độ.

➥Nó được làm từ bột lên men gồm đậu lăng và gạo.", "analyzer": "fingerprint"

}

Khi chúng ta thực thi đoạn mã này, kết quả sẽ như sau:

"mã thông báo" : [{

"token": "một loại bột nhão gồm bánh crepe dosa lên men từ Ấn Độ là nó

➥đậu lăng làm từ hoặc có nguồn gốc từ bánh kếp gạo mỏng miền Nam", "start\_offset" : 0,

"end\_offset" : 130, "type" : "vân tay", "vị trí" : 0

}]

Khi bạn xem xét kỹ phản hồi, bạn thấy rằng đầu ra chỉ bao gồm một mã thông báo. Các từ được viết thường và được sắp xếp, các từ trùng lặp (a, of, from) đượcđã xóa và tập hợp các từ đã được chuyển thành một mã thông báo duy nhất. Bất cứ khi nào chúng ta có văn bản cần được loại bỏ trùng lặp, sắp xếp và nối lại, dấu vân tay là lựa chọn hợp lý.

* + 1. Bộ phân tích mẫu

Đôi khi chúng ta muốn phân tích và mã hóa văn bản dựa trên một mẫu nhất định—ví dụ, xóa n số đầu tiên của số điện thoại hoặc xóa dấu gạch ngang giữa bốn chữ số của số thẻ tín dụng. Elasticsearch cung cấp trình phân tích mẫu cho mục đích đó.

Trình phân tích mẫu mặc định chia câu thành các mã thông báo dựa trên các ký tự không phải từ. Mẫu này được biểu diễn dưới dạng \W+ bên trong. Như hình 7.11 cho thấy, trình phân tích mẫu và bộ lọc chữ thường và dừng tạo thành trình phân tích mẫu.

**Không có bộ lọc ký tự**

Bộ lọc ký tự

Mã thông báo

Bộ lọc mã thông báo

mẫu

máy phân tích

dừng lại

chữ thường

mẫu

t máy đánh dấu

**Hình 7.11 Giải phẫu củamẫumáy phân tích**

***7.3 Tích hợp sẵnmáy phân tích* 255**

Vì trình phân tích mặc định (chuẩn) chỉ hoạt động trên các ký tự phân cách không phải chữ cái, chúng ta cần cấu hình trình phân tích để cung cấp bất kỳ mẫu nào khác được yêu cầu. Các mẫu là biểu thức chính quy Java được cung cấp dưới dạng chuỗi khi cấu hình trình phân tích.

**GHI CHÚ**Để tìm hiểu thêm về biểu thức chính quy Java, hãy truy cập[http://mng.bz/](http://mng.bz/QPDe) [QPDe](http://mng.bz/QPDe).

Giả sử chúng ta có một ứng dụng cho phép thanh toán thương mại điện tử và chúng ta nhận được yêu cầu cho phép thanh toán từ nhiều bên khác nhau. Một số thẻ gồm 16 chữ số được cung cấp theo định dạng 1234-5678-9000-0000. Chúng ta muốn mã hóa dữ liệu thẻ này trên một dấu gạch ngang (-) và trích xuất bốn mã thông báo riêng lẻ. Chúng ta có thể tạo một mẫu chia trường thành các mã thông báo dựa trên dấu phân cách dấu gạch ngang.

Để cấu hình trình phân tích mẫu, chúng ta phải tạo một chỉ mục bằng cách thiết lậppattern\_analyzer là trình phân tích trong đối tượng cài đặt. Danh sách sau đây hiển thị cấu hình.

**Danh sách 7.16 Mẫu phân định các mã thông báo dựa trêndấu gạch ngang**

PUT index\_with\_dash\_pattern\_analyzer

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích mẫu": {

"kiểu": "mẫu",



**Tạo một chỉ mục với các thiết lập phân tích**

**Trong cài đặt, xác định trình phân tích trong đối tượng phân tích**

**Chỉ định loại**

**Biểu thức chính quy biểu diễn**

**dấu gạch ngang**

}

"mẫu": "[-]",

"chữ thường": đúng

}

}

}

**bộ phân tích như mẫu Đính kèm một bộ lọc mã thông báo chữ thường**

}

Trong mã, chúng ta tạo một chỉ mục với các thiết lập phân tích mẫu. Thuộc tính mẫu chỉ ra regex, tuân theo cú pháp biểu thức chính quy của Java. Trong trường hợp này, chúng ta đặt dấu gạch ngang làm dấu phân cách, do đó văn bản được mã hóa khi gặp ký tự đó. Bây giờ chúng ta đã có chỉ mục, hãy đưa trình phân tích này vào hoạt động.

**Liệt kê 7.17 Kiểm tramẫumáy phân tích**

POST index\_with\_dash\_pattern\_analyzer/\_analyze

{

"văn bản": "1234-5678-9000-0000",

"máy phân tích": "máy phân tích mẫu"

}

Đầu ra của lệnh này chứa bốn mã thông báo:["1234","5678","9000","0000"].

Văn bản có thể được mã hóa dựa trên vô số các mẫu. Tôi đề xuất thử nghiệmvới các mẫu biểu thức chính quy để tận dụng tối đa lợi ích từ trình phân tích mẫu.

**256 CPHẦN7*Phân tích văn bản***

* + 1. Máy phân tích ngôn ngữ

Elasticsearch cung cấp các trình phân tích ngôn ngữ để làm việc với các ngôn ngữ sau: tiếng Ả Rập, tiếng Armenia, tiếng Basque, tiếng Bengal, tiếng Bulgaria, tiếng Catalan, tiếng Séc, tiếng Hà Lan, tiếng Anh, tiếng Phần Lan, tiếng Pháp, tiếng Galicia, tiếng Đức, tiếng Hindi, tiếng Hungary, tiếng Indonesia, tiếng Ireland, tiếng Ý, tiếng Latvia, tiếng Litva, tiếng Na Uy, tiếng Bồ Đào Nha, tiếng Rumani, tiếng Nga, tiếng Sorani, tiếng Tây Ban Nha, tiếng Thụy Điển và tiếng Thổ Nhĩ Kỳ. Chúng ta có thể định cấu hình các trình phân tích ngôn ngữ có sẵn này để thêm bộ lọc dừng từ để chúng ta không lập chỉ mục các từ không cần thiết (hoặc phổ biến) của ngôn ngữ. Danh sách sau đây minh họa ba trình phân tích ngôn ngữ (tiếng Anh, tiếng Đức và tiếng Hindi) đang hoạt động.

**Danh sách 7.18 Trình phân tích ngôn ngữ tiếng Anh, tiếng Đức và tiếng Hindi**

POST \_phân tích

{

"text": "Cô ấy bán vỏ sò ở bờ biển", "analyzer": "tiếng Anh"

**Sử dụng trình phân tích ngôn ngữ tiếng Anh. Đầu ra là ["she", "sell", "sea",**

**"vỏ sò", "biển", "bờ biển"].**

}

POST \_phân tích

{

"văn bản": "Guten Morgen", "máy phân tích": "tiếng Đức"

}

**Sử dụng công cụ phân tích ngôn ngữ tiếng Đức.**

**Đầu ra là ["gut","morg"].**

POST \_phân tích

{

"chữ": "",

"máy phân tích": "tiếng Hin-ddi"

}

**Sử dụng công cụ phân tích ngôn ngữ Hindi.**

**Đầu ra là ["","",""].**

Chúng ta có thể cấu hình một trình phân tích ngôn ngữ với một vài tham số bổ sung để cung cấp danh sách các từ dừng của riêng mình hoặc yêu cầu trình phân tích loại trừ hoạt động bắt nguồn. Đối vớiVí dụ, bộ lọc mã thông báo dừng được trình phân tích tiếng Anh sử dụng phân loại một số ít từ thành từ dừng. Chúng ta có thể ghi đè danh sách này dựa trên các yêu cầu của mình. Ví dụ, nếu chúng ta chỉ muốn ghi đè a, an, is, and, và for, chúng ta có thể định cấu hình các từ dừng của mình như danh sách sau đây hiển thị.

**Liệt kê 7.19 Mục lục với các từ dừng tùy chỉnh trênTiếng Anhmáy phân tích**

PUT index\_with\_custom\_english\_analyzer

{

"cài đặt": {

"Phân tích": {

"trình phân tích": { "index\_with\_custom\_english\_analyzer":{

**Tạo một chỉ mục**

**với cài đặt phân tích**

**Cung cấp tên tùy chỉnh**

**Loạicủa máy phân tích ở đây là tiếng Anh.**

"loại":"tiếng Anh", "từ dừng":["a","an","is","and","for"]

}

**Cung cấp bộ của chúng tôicủa các từ dừng**

* 1. ***Tích hợp sẵnmáy phân tích* 257**

}

}

}

}

Chúng tôi tạo một chỉ mục với một trình phân tích tiếng Anh tùy chỉnh và một tập hợp các từ dừng do người dùng xác định. Khi chúng tôi kiểm tra một đoạn văn bản với trình phân tích này, các từ dừng sẽ được tôn trọng.

**Liệt kê 7.20 Kiểm tra các từ dừng tùy chỉnh choTiếng Anhmáy phân tích**

POST index\_with\_custom\_english\_analyzer/\_analyze

{

"text":"Một con chó là cả cuộc đời", "analyzer":"index\_with\_custom\_english\_analyzer"

}

Mã này chỉ xuất ra hai mã thông báo: “dog” và “life”. Các từ a, is và for đã bị xóa vì chúng khớp với các từ dừng mà chúng tôi chỉ định.

Các trình phân tích ngôn ngữ có một tính năng khác mà chúng luôn mong muốn triển khai: stemming. Stemming là một cơ chế để rút gọn các từ thành dạng gốc của chúng. Ví dụ, bất kỳ dạng nào của từ author (authors, authoring, authored, v.v.) đều được rút gọn thành một từ author. Danh sách sau đây cho thấy hành vi này.

**Liệt kê 7.21 Giảm tất cả các hình thứctác giảđếntác giảtừ khóa**

POST index\_with\_custom\_english\_analyzer/\_analyze

{

"text":"tác giả tác giả tác giả đã viết", "analyzer":"tiếng anh"

}

Mã này tạo ra bốn mã thông báo (dựa trên trình phân tích khoảng trắng), tất cả đều là “tác giả”—vì từ gốc của bất kỳ dạng tác giả nào đều là tác giả!

Nhưng đôi khi, việc trích dẫn đi quá xa. Nếu bạn thêm “authorization” hoặc “authority” vào danh sách các từ trong danh sách trước đó, nó cũng được trích dẫn và lập chỉ mục là “author”! Rõ ràng là chúng ta không thể tìm thấy câu trả lời thích hợp khi tìm kiếm “authority” hoặc “authorization” vì những từ đó không được đưa vào chỉ mục đảo ngược do trích dẫn.

Không phải tất cả đều mất. Chúng ta có thể cấu hình trình phân tích tiếng Anh của mình, yêu cầu nó bỏ qua một số từ nhất định (chẳng hạn như ủy quyền và thẩm quyền, trong trường hợp này) không cần phải vượt quatrình phân tích. Chúng ta có thể sử dụng thuộc tính stem\_exclusion để cấu hình các từ loại trừ khỏi quá trình trích xuất. Danh sách tiếp theo thực hiện điều này bằng cách tạo chỉ mục với các thiết lập tùy chỉnh và truyền các đối số cho tham số stem\_exclusion.

**Liệt kê 7.22 Tạo chỉ mục với các từ loại trừ gốc tùy chỉnh**

PUT index\_with\_stem\_exclusion\_english\_analyzer

{

"cài đặt": {

**258 CPHẦN7*Phân tích văn bản***

"Phân tích": {

"máy phân tích": { "máy phân tích tiếng Anh loại trừ gốc":{

"type":"tiếng Anh", "stem\_exclusion":["authority","authorization"]

}

}

}

}

}

Sau khi chúng ta đã tạo chỉ mục với các thiết lập này, bước tiếp theo là kiểm tra việc lập chỉ mụcyêu cầu. Danh sách sau đây sử dụng trình phân tích tiếng Anh để kiểm tra một đoạn văn bản.

**Liệt kê 7.23 Loại trừ gốc tronghoạt động**

POST index\_with\_stem\_exclusion\_english\_analyzer/\_analyze

{

"text": "Không ai có thể thách thức thẩm quyền của tôi nếu không có sự cho phép của tôi", "analyzer": "stem\_exclusion\_english\_analyzer"

}

Các token được tạo ra bởi mã này là các token thông thường của chúng tôi cộng với hai từ không bị thay đổi: “authority” và “authorization”. Hai từ này sẽ được xuất ra cùng với các từ “gốc” khác.

Chúng ta cũng có thể tùy chỉnh thêm các trình phân tích ngôn ngữ. Chúng ta sẽ thảo luận về việc tùy chỉnh các trình phân tích trong phần tiếp theo.

Trong khi hầu hết các trình phân tích đều thực hiện những gì chúng ta muốn trong hầu hết các trường hợp, đôi khi chúng ta cần triển khai phân tích văn bản cho các yêu cầu bổ sung. Ví dụ, chúng ta có thể muốn xóa các ký tự đặc biệt như thẻ HTML khỏi văn bản đến hoặc tránh các từ dừng cụ thể. Công việc xóa các thẻ HTML được bộ lọc ký tự html\_strip thực hiện, nhưng thật không may, không phải tất cả các trình phân tích đều có bộ lọc này. Trong những trường hợp như vậy, chúng ta có thể tùy chỉnh trình phân tích bằng cách định cấu hình chức năng cần thiết: chúng ta có thể thêm bộ lọc ký tự mới nhưhtml\_strip và có thể bật bộ lọc mã thông báo dừng. Hãy xem cách cấu hình trình phân tích chuẩn để phù hợp với các yêu cầu nâng cao.

#### Bộ phân tích tùy chỉnh

Elasticsearch cung cấp rất nhiều tính linh hoạt khi nói đến các trình phân tích: nếu các trình phân tích có sẵn không đáp ứng được nhu cầu của chúng ta, chúng ta có thể tạo ra các trình phân tích tùy chỉnh của riêng mình. Chúng có thể kết hợp và khớp các thành phần hiện có từ một kho lưu trữ lớn trong thư viện thành phần của Elasticsearch.

Thực hành thông thường là xác định một trình phân tích tùy chỉnh trong cài đặt khi tạo chỉ mục với các bộ lọc cần thiết và một trình phân tích mã thông báo. Chúng tôi có thể cung cấp bất kỳ số lượng bộ lọc ký tự và mã thông báo nào nhưng chỉ có một trình phân tích mã thông báo (xem hình 7.12).

Như hình minh họa, chúng tôi định nghĩa một trình phân tích tùy chỉnh trên một chỉ mục bằng cách đặt loại thành tùy chỉnh. Trình phân tích tùy chỉnh của chúng tôi được phát triển với một mảng các bộ lọc ký tự đại diện chođược gửi bởi đối tượng char\_filter và một mảng các bộ lọc mã thông báo được biểu diễn bởi thuộc tính filter.

* 1. ***Phong tụcmáy phân tích* 259**

**Tạo định nghĩa chỉ mục với các thiết lập cho chức năng phân tích văn bản**

PUT index\_with\_custom\_analyzer

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích\_tùy\_chỉnh":{

"loại":"tùy chỉnh",

**Xác định**

**kiểuBẰNG Khai báo một mảng**

**phong tục bộ lọc ký tự**

"char\_filter":["charfilter1","charfilter2"], "tokenizer":"chuẩn", "bộ lọc":["tokenfilter1","tokenfilter2"]

}

} **Đính kèm một bắt buộc**

}

} **Khai báo một mảng các bộ lọc mã thông báo**

}

**bộ mã hóa**

**Hình 7.12 Giải phẫu của một máy phân tích tùy chỉnh**

**GHI CHÚ**Theo tôi, các nhà phát triển Elasticsearch nên đặt tênlọcsự vậtbộ lọc token, thay vào đó, để phù hợpbộ lọc ký tự. Và sử dụng tên số nhiều (bộ lọc ký tựVàbộ lọc token) cũng có ý nghĩa hơn vì chúng mong đợi một mảng các bộ lọc được chuỗi hóa!

Chúng tôi cung cấp một trình phân tích từ danh sách các trình phân tích có sẵn với cấu hình tùy chỉnh của chúng tôi. Hãy xem một ví dụ. Liệt kê 7.24 hiển thị tập lệnh để phát triển trình phân tích tùy chỉnh. Nó có các mục sau:

* Một bộ lọc ký tự (html\_dải) xóa các ký tự HTML khỏi trường nhập liệu
* MỘTtiêu chuẩnbộ mã hóa dùng để mã hóa trường dựa trên khoảng trắng và dấu câu
* Bộ lọc mã thông báo cho các từ viết hoa

**Danh sách 7.24 Trình phân tích tùy chỉnh với bộ lọc và trình phân tích mã thông báo**

PUT index\_with\_custom\_analyzer

{

**Mảng ký tự**

**bộ lọc**

"cài đặt": {

"Phân tích": {

**Chỉ định**

**phân tích tùy chỉnh**

"máy phân tích": { "máy phân tích\_tùy\_chỉnh":{

**"kiểu":"tùy chỉnh", "char\_filter":["html\_strip"], "tokenizer":"chuẩn", "filter":["chữ hoa"]**



**Loại phải là tùy chỉnh để Elasticsearch biết về trình phân tích tùy chỉnh của chúng tôi.**

**Khai báo một trình phân tích cú pháp duy nhất (chuẩn trong trường hợp này)**

}

} **Bộ lọc mã thông báo thành chữ hoa**

} **đang đếncác mã thông báo**

}

}

**260 CPHẦN7*Phân tích văn bản***

Chúng ta có thể kiểm tra trình phân tích bằng đoạn mã sau:

POST index\_with\_custom\_analyzer/\_analyze

{

"text": "<H1>XIN CHÀO, WoRLD</H1>",

"máy phân tích": "máy phân tích tùy chỉnh"

}

Chương trình này tạo ra hai mã thông báo: ["HELLO", "WORLD"]. Bộ lọc html\_strip của chúng tôi đã xóa các thẻ HTML H1 trước khi cho phép trình phân tích mã thông báo chuẩn chia trường thành hai mã thông báo dựa trên dấu phân cách khoảng trắng. Cuối cùng, các mã thông báo được viết hoa nhưhọ đã vượt qua bộ lọc ký tự viết hoa.

Việc tùy chỉnh giúp đáp ứng nhiều yêu cầu, nhưng chúng ta có thể đạt được kết quả tiên tiến hơn nữa. Hãy cùng xem xét.

* + 1. Trình độ cao tùy chỉnh

Trong khi cấu hình mặc định của các thành phần phân tích hoạt động hầu hết thời gian, đôi khi chúng ta cần tạo các bộ phân tích với cấu hình không mặc định cho thành phần của chúng.thành phần. Ví dụ, chúng ta có thể muốn sử dụng bộ lọc ký tự ánh xạ để ánh xạ ký tự & thành từ và và các ký tự < và > thành nhỏ hơn và lớn hơn tương ứng.

Giả sử yêu cầu của chúng ta là phát triển một trình phân tích tùy chỉnh phân tích văn bản cho các chữ cái Hy Lạp và tạo ra một danh sách các chữ cái Hy Lạp. Mã sau đây tạo một chỉ mục với các thiết lập phân tích này.

**Liệt kê 7.25 Trích xuất các chữ cái Hy Lạp bằng trình phân tích tùy chỉnh**

PUT index\_with\_parse\_greek\_letters\_custom\_analyzer

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích chữ cái Hy Lạp":{

"loại":"tùy chỉnh",

**Tạo ramột trình phân tích cú pháp chữ cái Hy Lạp tùy chỉnh**

**Bộ phân tích tùy chỉnh được thực hiện**

**Một tiêu chuẩntokenizer phân tách văn bản thành dạng mã hóa.**

"char\_filter":["greek\_symbol\_mapper"], "tokenizer":"standard", "filter":["lowercase", "greek\_keep\_words"]

}

}, **Định nghĩa các chữ cái Hy Lạp và**

**ánh xạ chúng thành các từ tiếng Anh**

**của char\_filter tùy chỉnh.**

**Cung cấp hai bộ lọc mã thông báo; greek\_keep\_words được định nghĩa tiếp theo.**

"bộ lọc ký tự": {

"greek\_symbol\_mapper":{ "type":"bản đồ",

"ánh xạ":[

"α => alpha", "β => Beta", "γ => Gamma"

**Các ánh xạ thực tế: danh sách các ký hiệu và giá trị tương ứng**

]

}

}, **Chúng tôi không muốn lập chỉ mục tất cả**

"bộ lọc": { "greek\_keep\_words":{

**các giá trị trường, chỉ có từ ngữ**

**phù hợp với các từ giữ.**

* 1. ***Chỉ định các bộ phân tích***

"loại":"giữ",

"keep\_words":["alpha", "beta", "gamma"]

}

}

}

}

}

**Giữ lại các từ; tất cả các từ khác sẽ bị loại bỏ.**

**261**

Mã này hơi khó hiểu một chút, nhưng dễ hiểu. Trong phần đầu tiên, nơi chúng ta định nghĩa một trình phân tích tùy chỉnh, chúng tôi cung cấp danh sách các bộ lọc (cả bộ lọc ký tự và mã thông báo, nếu cần) và một trình phân tích. Bạn có thể hình dung đây là điểm vào cho định nghĩa trình phân tích.

Phần thứ hai của mã định nghĩa các bộ lọc được khai báo trước đó. Ví dụ:người lập bản đồ biểu tượng Hy Lạp, được tuyên bố lại trong mớibộ lọc ký tựphần, sử dụnglập bản đồnhư loại bộ lọc với một tập hợp các ánh xạ. Tương tự như vậy đối vớilọckhối, xác địnhgiữ\_từbộ lọc. Bộ lọc này loại bỏ bất kỳ từ nào không có tronggiữ\_lờidanh sách.

Bây giờ chúng ta có thể thực hiện một mẫu thử nghiệm để phân tích. Câu trong danh sách sau đây phải được chuyển qua giai đoạn phân tích thử nghiệm.

**Liệt kê 7.26 Phân tích các chữ cái Hy Lạp từ văn bản bình thường**

POST index\_with\_parse\_greek\_letters\_custom\_analyzer/\_analyze

{

"text": "α và β là nghiệm của phương trình bậc hai. γ thì không", "analyzer": "greek\_letter\_custom\_analyzer"

}

Các chữ cái Hy Lạp (ví dụ: ,  và ) được xử lý bởi trình phân tích tùy chỉnh (phân tích chữ cái tiếng Hy Lạp), kết quả trả về như sau:

"alpha", "beta", "gamma"

Các từ còn lại, như căn bậc hai và phương trình bậc hai, sẽ bị loại bỏ.

Cho đến nay, chúng ta đã xem xét chi tiết các trình phân tích, bao gồm các trình phân tích tích hợp và tùy chỉnh. Chúng ta có thể cấu hình các trình phân tích không chỉ ở cấp độ trường mà còn ở cấp độ chỉ mục hoặc truy vấn. Và chúng ta cũng có thể chỉ định mộtbộ phân tích cho các truy vấn tìm kiếm nếu yêu cầu của chúng tôi yêu cầu. Chúng tôi thảo luận về những điểm này trong phần tiếp theo.

#### Chỉ định các bộ phân tích

Có thể chỉ định trình phân tích ở cấp độ chỉ mục, trường và truy vấn. Việc khai báo trình phân tích ở cấp độ chỉ mục cung cấp trình phân tích bắt tất cả mặc định trên toàn chỉ mục cho tất cả các trường văn bản. Tuy nhiên, nếu cần tùy chỉnh thêm ở cấp độ trường, chúng ta cũng có thể bật trình phân tích khác ở cấp độ đó. Ngoài ra, chúng ta có thể cung cấp trình phân tích khác thay vì trình phân tích thời gian chỉ mục trong khi tìm kiếm. Hãy cùng xem xét từng tùy chọn này.

* + 1. Máy phân tích để lập chỉ mục

Đôi khi chúng ta có thể có yêu cầu thiết lập các trường khác nhau với các trình phân tích khác nhau—ví dụ: liên kết trường tên với trình phân tích đơn giản và thẻ tín dụng

**262 CPHẦN7 *Phân tích văn bản***

trường số với một trình phân tích mẫu. May mắn thay, Elasticsearch cho phép chúng ta thiết lập các trình phân tích khác nhau trên từng trường khi cần thiết. Tương tự như vậy, chúng ta có thể thiết lập một trình phân tích mặc định cho mỗi chỉ mục để bất kỳ trường nào không được liên kết rõ ràng với một trình phân tích cụ thể trong quá trình ánh xạ đều kế thừa trình phân tích cấp chỉ mục.

**FTIẾNG ANH****-MÁY PHÂN TÍCH MỨC**

Chúng ta có thể chỉ định các bộ phân tích cần thiết ở cấp độ trường trong khi tạo định nghĩa ánh xạ của chỉ mục. Mã sau đây cho thấy cách chúng ta có thể thực hiện việc này trong quá trình tạo chỉ mục.

**Liệt kê 7.27 Thiết lập trình phân tích cấp trường trong quá trình tạo chỉ mục**

PUT tác giả\_với\_trường\_cấp\_phân\_tích

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

},

"Về":{

"type": "text", "analyzer": "tiếng anh"

},

"mô tả":{ "kiểu": "văn bản", "trường": {

"Của tôi":{

"loại": "văn bản",

**Sử dụng tiêu chuẩnmáy phân tích**

**Thiết lập rõ ràng trình phân tích tiếng Anh**

"máy phân tích": "dấu vân tay"

}

}

}

}

}

}

**Máy phân tích dấu vân tay trên nhiều trường**

CácVềVàSự miêu tảcác trường được chỉ định với các bộ phân tích khác nhau ngoại trừ

têncánh đồng,mà ngầm thừa hưởngtiêu chuẩnmáy phân tích.

**TÔINDEX****-MÁY PHÂN TÍCH MỨC**

Chúng ta cũng có thể thiết lập một trình phân tích mặc định theo lựa chọn của mình ở cấp độ chỉ mục. Danh sách sau đây minh họa.

**Liệt kê 7.28 Tạo chỉ mục với trình phân tích mặc định**

PUT tác giả\_với\_default\_analyzer

{

"cài đặt": {

"Phân tích": {

"máy phân tích": {

"mặc định":{

"loại":"từ khóa"

**Thiết lập thuộc tính này sẽ thiết lập trình phân tích mặc định của chỉ mục.**

* 1. ***Chỉ địnhmáy phân tích* 263**

}

}

}

}

}

Trên thực tế, chúng tôi đang thay thế trình phân tích chuẩn, là trình phân tích mặc định cho một từ khóa

bộ phân tích. Chúng ta có thể kiểm tra bộ phân tích bằng cách gọi điểm cuối \_analyze trên chỉ mục.

**Liệt kê 7.29 Kiểm tra trình phân tích mặc định**

PUT tác giả\_với\_default\_analyzer/\_analyze

{

"văn bản":"John Doe"

}

Mã này đưa ra một mã thông báo duy nhất "John Doe" không có chữ thường hoặc mã thông báo, chỉ địnhcho biết trình phân tích từ khóa của chúng tôi đã phân tích nó. Bạn có thể thử cùng một mã bằng trình phân tích chuẩn và bạn sẽ nhận thấy sự khác biệt.

Việc thiết lập các trình phân tích ở cấp độ chỉ mục hoặc cấp độ trường hoạt động trong quá trình lập chỉ mục. Tuy nhiên, chúng ta cũng có thể sử dụng một trình phân tích khác trong quá trình truy vấn—hãy xem lý do và cách thực hiện.

* + 1. Các bộ phân tích để tìm kiếm

Elasticsearch cho phép chúng ta chỉ định một trình phân tích khác trong thời gian truy vấn so với khi lập chỉ mục. Chúng ta hãy xem hai phương pháp này trong phần này cũng như một số quy tắc mà Elasticsearch tuân theo khi chọn một trình phân tích được xác định ở nhiều cấp độ khác nhau.

**MỘTNALYZER TRONG MỘT QUERY**

Chúng tôi chưa đi sâu vào tìm kiếm, vì vậy đừng lo lắng nếu danh sách sau đây khiến bạn hơi bối rối (chúng tôi sẽ thảo luận về tìm kiếm trong vài chương tiếp theo).

**Liệt kê 7.30 Thiết lập trình phân tích bên cạnh truy vấn tìm kiếm**

NHẬN tác giả\_index\_for\_search\_analyzer/\_search

{

"truy vấn": {

"cuộc thi đấu": {

**Truy vấn để tìm kiếm tất cả các tác giả**

**với các tiêu chí đã cho**

"author\_name": { "query": "M Konda", "analyzer": "simple"

}

}

}

**}**

**Bộ phân tích được chỉ định rõ ràng và thường khác với bộ phân tích mà trường được lập chỉ mục.**

Chúng tôichỉ định rõ ràng trình phân tích (trường author\_name có thể đã được lập chỉ mục bằng một loại trình phân tích khác!) khi tìm kiếm tác giả.

**264 CPHẦN7*Phân tích văn bản***

**SĐẶT MÁY PHÂN TÍCH Ở CẤP ĐỘ TRƯỜNG**

Cơ chế thứ hai thiết lập trình phân tích tìm kiếm cụ thể ở cấp độ trường. Cũng giống như chúng ta thiết lập trình phân tích trên một trường cho mục đích lập chỉ mục, chúng ta có thể thêm một thuộc tính được gọi làsearch\_analzyer trên một trường để chỉ định trình phân tích tìm kiếm.

**Liệt kê 7.31 Thiết lập trình phân tích tìm kiếm ở cấp độ trường**

PUT tác giả\_index\_với\_cả\_cả\_trường\_phân\_tích\_cấp

{

"ánh xạ": { "thuộc tính": {

"tên\_tác\_giả":{ "loại": "văn\_bản",

**"analyzer": "dừng", "search\_analyzer": "đơn giản"**

}

}

}

}

Cáctên tác giảtrường được thiết lập với mộtdừng lạimáy phân tích để lập chỉ mục và mộtđơn giảnbộ phân tích thời gian tìm kiếm.

**DMÁY PHÂN TÍCH LỖI MẶC ĐỊNH Ở CẤP ĐỘ CHỈ MỤC**

Chúng ta cũng có thể thiết lập trình phân tích mặc định cho các truy vấn tìm kiếm, giống như chúng ta đã làm đối với thời gian lập chỉ mục, bằng cách thiết lập trình phân tích cần thiết trên chỉ mục tại thời điểm tạo chỉ mục.

**Liệt kê 7.32 Thiết lập trình phân tích mặc định để tìm kiếm và lập chỉ mục**

PUT tác giả\_index\_with\_default\_analyzer

{

"cài đặt": {

"Phân tích": {

"trình phân tích": { "tìm kiếm mặc định":{

"loại":"đơn giản"

},

"mặc định":{ "kiểu":"chuẩn"

}

}

}

}

}

**Đặt trình phân tích tìm kiếm mặc định cho chỉ mục bằng cách sử dụng thuộc tính default\_search**

**Bộ phân tích mặc định cho chỉ mục**

Chúng tôi thiết lập trình phân tích mặc định để lập chỉ mục và tìm kiếm cùng lúc. Bạn có thể tự hỏi liệu chúng tôi có thể thiết lập trình phân tích tìm kiếm ở cấp trường trong quá trình lập chỉ mục thay vì tại thời điểm chạy trong quá trình truy vấn hay không. Danh sách sau đây minh họa chính xác điều đó: nó thiết lập các trình phân tích khác nhau để lập chỉ mục và tìm kiếm ở cấp trường trong quá trình tạo chỉ mục.

* 1. ***Tính cáchbộ lọc* 265**

**Liệt kê 7.33 Chỉ định các trình phân tích chỉ mục/tìm kiếm trong quá trình tạo chỉ mục**

PUT tác giả\_index\_với\_cả\_cả\_trường\_phân\_tích\_cấp

{

"ánh xạ": { "thuộc tính": {

"tên\_tác\_giả":{ "loại": "văn\_bản",

"analyzer": "chuẩn", "search\_analyzer": "đơn giản"

}

}

}

}

Cáctên tác giảlĩnh vực sử dụng mộttiêu chuẩnmáy phân tích để lập chỉ mục và mộtđơn giảnmáy phân tích trong quá trình tìm kiếm.

**THETHỨ TỰ CỦA TIỀN LỆ**

Công cụ bắt đầu sử dụng các trình phân tích dựa trên thứ tự ưu tiên sau khi tìm thấy các trình phân tích ở nhiều cấp độ khác nhau (mức ưu tiên cao nhất sẽ nằm ở phía trước):

**1** Một trình phân tích được xác định ở cấp độ truy vấn

**2**Một trình phân tích được xác định bằng cách đặt thuộc tính search\_analyzer trên một trường khi xác định ánh xạ chỉ mục

**3** Một bộ phân tích được xác định ở cấp độ chỉ mục

**4** Bộ phân tích lập chỉ mục được đặt trên một trường hoặc một chỉ mục

Bây giờ chúng ta đã hiểu về các bộ phân tích tích hợp và cách tạo các bộ phân tích tùy chỉnh, đã đến lúc xem xét chi tiết các thành phần riêng lẻ của bộ phân tích. Trong ba phần tiếp theo, chúng ta sẽ xem xét các thành phần tạo nên một bộ phân tích: bộ phân tích mã thông báo và bộ lọc ký tự và mã thông báo. Hãy bắt đầu với bộ lọc ký tự.

#### 7.6 Bộ lọc ký tự

Khi người dùng tìm kiếm câu trả lời, kỳ vọng là họ sẽ không tìm kiếm bằng dấu câu hoặc ký tự đặc biệt. Ví dụ, có khả năng cao là người dùng sẽ tìm kiếm “không tìm thấy chìa khóa của tôi” (không có dấu câu) thay vì “không tìm thấy chìa khóa của tôi !!!”. Tương tự như vậy, người dùng không nên tìm kiếm chuỗi “<h1>Chiếc phô mai của tôi ở đâu?</h1>” (có thẻ HTML). Chúng tôi cũng không mong đợi người dùng tìm kiếm bằng thẻ XML như “<operation>callMe</operation>”. Tiêu chí tìm kiếm không cần phải bị ô nhiễm bởi các ký tự không cần thiết. Chúng tôi cũng không muốn người dùng tìm kiếm bằng các ký hiệu: ? thay vì “alpha” hoặc ? thay cho “beta”, chẳng hạn.

Dựa trên những giả định này, chúng ta có thể phân tích và làm sạch văn bản đến bằng bộ lọc ký tự. Bộ lọc ký tự giúp loại bỏ các ký tự không mong muốn khỏi luồng đầu vào. Mặc dù chúng là tùy chọn, nhưng nếu được sử dụng, chúng sẽ tạo thành thành phần đầu tiên trong mô-đun phân tích.

**266 CPHẦN7*Phân tích văn bản***

Một bộ phân tích có thể có không hoặc nhiều bộ lọc ký tự. Một bộ lọc ký tự thực hiện các chức năng cụ thể sau:

* *Xóa các ký tự không mong muốn khỏi luồng đầu vào*—Ví ​​dụ, nếu văn bản đến có đánh dấu HTML, như trong “<h1>Phô mai của tôi ở đâu?</h1>”, yêu cầu là phải bỏ<h1>Và</h1>thẻ.
* *Thêm vào hoặc thay thế các ký tự bổ sung trong luồng hiện có—*Nếu trường đầu vào chứa 0 và 1, chúng ta có thể muốn thay thế chúng bằng “false” và “true”, tương ứng. Hoặc nếu luồng đầu vào bao gồm ký tự ?, chúng ta có thể ánh xạ nó thành từ “beta” và lập chỉ mục cho trường.

Chúng ta có thể sử dụng ba bộ lọc ký tự để xây dựng một bộ phân tích: bộ lọc html\_strip, mapping và pattern. Chúng ta đã thấy những bộ lọc này hoạt động trước đó và trong phần này, chúng ta sẽ xem xét ngắn gọn về ngữ nghĩa.

7.6.1 Dải HTML(hmtl\_strip) bộ lọc

Như tên gọi của nó, bộ lọc này loại bỏ các thẻ HTML không mong muốn khỏi các trường nhập. Đối vớiVí dụ, khi một trường nhập có giá trị là<h1>Phô mai của tôi đâu?</h1>được xử lý bởihtml\_dảibộ lọc ký tự,<h1>Và</h1>các thẻ được xóa, để lại "Phô mai của tôi đâu?". Chúng ta có thể kiểm trahtml\_dảibộ lọc ký tự sử dụng\_phân tíchGiao diện lập trình ứng dụng (API).

**Liệt kê 7.34html\_dảibộ lọc ký tự đang hoạt động**

POST \_phân tích

{

"text":"<h1>Phô mai của tôi đâu?</h1>", "tokenizer": "standard", "char\_filter": ["html\_strip"]

}

Bộ lọc ký tự chỉ đơn giản là loại bỏ các thẻ HTML khỏi trường nhập và tạo ra các mã thông báo “Where”, “is”, “my” và “Cheese”. Tuy nhiên, chúng ta có thể cần tránh phân tích cú pháp trường nhập cho một số thẻ HTML nhất định. Ví dụ, yêu cầu kinh doanh có thể làxóa thẻ <h1> khỏi phần đầu vào sau nhưng vẫn giữ nguyên các thẻ được định dạng sẵn (<pre>):

<h1>Phô mai của tôi đâu?</h1>

<pre>Chúng ta là con người và luôn tìm kiếm pho mát!</pre>

Chúng ta có thể cấu hình bộ lọc html\_strip để thêm một mảng escaped\_tags với danh sách các thẻ để không phân tích cú pháp. Bước đầu tiên là tạo một chỉ mục với trình phân tích tùy chỉnh cần thiết.

**Danh sách 7.35 Bộ phân tích tùy chỉnh có cấu hình bộ lọc được thêm vào**

PUT index\_with\_html\_strip\_filter

{

"cài đặt": {

"Phân tích": {

"máy phân tích": {

***7.6 Bộ lọc ký tự***

"custom\_html\_strip\_filter\_analyzer":{ "tokenizer":"từ khóa", "char\_filter":["my\_html\_strip\_filter"]

}

},

"char\_filter": { "my\_html\_strip\_filter":{

"kiểu":"html\_strip",

**Tuyên bố một phong tụcbộ lọc ký tự**

**267**

"escaped\_tags":["h1"]

}

}

}

}

}

**Thuộc tính escaped\_tags bỏ qua việc phân tích thẻ <h1> trong trường nhập liệu.**

Chỉ mục này có một trình phân tích tùy chỉnh bao gồm bộ lọc ký tự html\_strip.html\_strip được mở rộng trong ví dụ này để sử dụng tùy chọn escaped\_tags, tùy chọn này chỉ định rằng thẻ <h1> không được đụng đến. Để kiểm tra điều này, chúng ta có thể chạy đoạn mã sau.

**Liệt kê 7.36 Kiểm tra trình phân tích tùy chỉnh**

POST index\_with\_html\_strip\_filter/\_analyze

{

"text": "<h1>Xin chào,</h1> <h2>Thế giới!</h2>", "analyzer": "custom\_html\_strip\_filter\_analyzer"

}

Mã này để lại từ với<h1>gắn thẻ như vậy và xóa bỏ<h2>thẻ. Kết quả đầu ra như sau:"Xin chào,</h1> Thế giới!".

* + 1. Bộ lọc ký tự ánh xạ

Cáccông việc duy nhất của bộ lọc ký tự ánh xạ là khớp một khóa và thay thế nó bằng một giá trị. Trong ví dụ trước đó của chúng tôi về việc chuyển đổi các chữ cái Hy Lạp thành các từ tiếng Anh, bộ lọc ánh xạ đã phân tích các ký hiệu và thay thế chúng bằng các từ: như alpha,  như beta, v.v.

Hãy thử bộ lọc ký tự ánh xạ. Trong danh sách sau, “UK” được thay thế bằng “United Kingdom” khi phân tích cú pháp bằng bộ lọc ánh xạ.

**Liệt kê 7.37lập bản đồbộ lọc ký tự đang hoạt động**

POST \_phân tích

{

"text": "Tôi đến từ Vương quốc Anh", "char\_filter": [

{

"loại": "ánh xạ", "ánh xạ": [

"Anh => Vương quốc Anh"

]

}

]

}

**268 CPHẦN7*Phân tích văn bản***

Nếu chúng ta muốn tạo một trình phân tích tùy chỉnh với bộ lọc ký tự ánh xạ được cấu hình, chúng ta làm theo cùng một quy trình như trước đó để tạo một chỉ mục với các thiết lập trình phân tích và các bộ lọc bắt buộc. Ví dụ này cho thấy quy trình tùy chỉnh trình phân tích từ khóa để đính kèm bộ lọc ký tự ánh xạ.

**Liệt kê 7.38 Atừ khóamáy phân tích với mộtlập bản đồbộ lọc ký tự**

PUT index\_with\_mapping\_char\_filter

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích xã hội của tôi": {

"tokenizer": "từ khóa",

**Bộ phân tích tùy chỉnh với bộ lọc ký tự ánh xạ**

}

},

"char\_filter": [ "my\_social\_abbreviations"

]

**Tuyên bố một**

**bộ lọc ký tự**

**Chỉ địnhtên của bộ lọc: ánh xạ, trong trường hợp này**

"char\_filter": { "my\_social\_abbreviations": {

"loại": "ánh xạ", "ánh xạ": [

"LOL => cười phá lên", "BRB => quay lại ngay", "OMG => trời ơi"

**Mở rộng ký tự được xác định**

**lọc với ánh xạ**

**Cung cấp một tập hợp các ánh xạ trong đối tượng ánh xạ dưới dạng cặp tên-giá trị**

]

}

}

}

}

}

Chúng tôi đã tạo một chỉ mục với các thiết lập phân tích tùy chỉnh và cung cấp các ánh xạ trongbộ lọc ký tự. Bây giờ chúng ta có thể kiểm tra nó bằng API \_analyze.

**Liệt kê 7.39 Kiểm tra trình phân tích tùy chỉnh**

POST index\_with\_mapping\_char\_filter/\_analyze

{

"văn bản": "LOL",

"máy phân tích": "my\_social\_abbreviations\_analyzer"

}

Văn bản dẫn đến"token" : "cười lớn", điều này cho thấy rằng “LOL” đã được thay thế bằng dạng đầy đủ là “cười lớn”.

* + 1. Ánh xạ thông qua một tập tin

Chúng tôi cũng có thể cung cấp một tệp chứa các ánh xạ thay vì chỉ định chúng trong định nghĩa. Liệt kê 7.40 sử dụng bộ lọc ký tự với các ánh xạ được tải từ một bên ngoài

* 1. ***Tính cáchbộ lọc* 269**

tệp, secret\_organizations.txt (hãy đảm bảo tạo tệp này nếu bạn không có tệp này trong thư mục config). Tệp này phải có trong thư mục config của Elasticsearch (<INSTALL\_DIR>/ elasticsearch/config) hoặc nhập đường dẫn tuyệt đối đến vị trí tệp này.

**Liệt kê 7.40 Tải các ánh xạ bên ngoài thông qua một tệp**

POST \_phân tích

{

"text": "FBI và CIA là các tổ chức an ninh của Hoa Kỳ", "char\_filter": [

{

"loại": "bản đồ",

"mappings\_path": "secret\_organizations.txt"

}

]

}

Tệp mẫu secret\_organizations.txt phải bao gồm các dữ liệu sau:

FBI=>Cục Điều tra Liên bang CIA=>Cơ quan Tình báo Trung ương Hoa Kỳ=>Hoa Kỳ

* + 1. Bộ lọc ký tự pattern\_replace

Bộ lọc ký tự pattern\_replace, như tên gọi của nó, thay thế một ký tự bằng một ký tự mới khi trường khớp với một biểu thức chính quy (regex). Theo cùng một mẫu mã như trong danh sách 7.38, chúng ta hãy tạo một chỉ mục có trình phân tích được liên kết với bộ lọc ký tự pattern\_replace.

**Liệt kê 7.41 Sử dụng mộtmẫu\_thay\_thếbộ lọc ký tự**

PUT index\_with\_pattern\_replace\_filter

{

"cài đặt": {

"Phân tích": {

"trình phân tích": { "trình phân tích thay thế mẫu của tôi":{

"tokenizer":"từ khóa",

**Bộ phân tích tùy chỉnh với bộ lọc ký tự pattern\_replace**

"char\_filter":["pattern\_replace\_filter"]

}

},

**Khai báo bộ lọc pattern\_replace**

**Chỉ ra mẫu để tìm kiếm**

**và thay thế**

"bộ lọc\_ký\_tự": { "bộ lọc\_thay\_thế\_mẫu":{

"kiểu":"thay thế mẫu","mẫu":"\_",

"thay thế":"-"

}

**Mở rộng định nghĩa**

**bộ lọc pattern\_replace với các tùy chọn**

**Chỉ ra loại bộ lọc (pattern\_replace)**

**Xác định giá trị thay thế**

}

}

}

}

**270 CPHẦN7*Phân tích văn bản***

Mã này cố gắng khớp và thay thế trường nhập của chúng ta, thay thế ký tự gạch dưới (\_) bằng dấu gạch ngang (-). Chúng ta có thể kiểm tra trình phân tích bằng mã sau.

**Liệt kê 7.42 Kiểm tra bộ lọc ký tự thay thế mẫu tùy chỉnh**

POST index\_with\_pattern\_replace\_filter/\_analyze

{

"văn bản": "Apple\_Boy\_Cat",

"máy phân tích": "my\_pattern\_replace\_analyzer"

}

Đầu ra là “Apple-Boy-Cat” với tất cả các dấu gạch dưới được thay thế bằng dấu gạch ngang.

Văn bản đầu vào đã được xóa các ký tự không mong muốn, nhưng vẫn còn công việc chia văn bản thành các mã thông báo riêng lẻ dựa trên các dấu phân cách, mẫu và các tiêu chí khác. Công việc đó được thực hiện bởi một thành phần phân tích mã thông báo, được thảo luận trong phần tiếp theo.

#### Bộ phân tích mã thông báo

Công việc của một tokenizer là tạo ra các token dựa trên các tiêu chí nhất định. Tokenizer chia các trường nhập liệu đến thành các token thường là các từ riêng lẻ của một câu. Elasticsearch cung cấp hơn một chục tokenizer, mỗi tokenizer sẽ token hóa các trường dựa trên định nghĩa của tokenizer.

**GHI CHÚ**Như bạn có thể tưởng tượng, thảo luận về tất cả các tokenizertrong cuốn sách này sẽ không thực tế (và cũng nhàm chán khi đọc). Tôi đã chọn một số tokenizer quan trọng và phổ biến ở đây để bạn có thể hiểu khái niệm và cơ chế. Mã ví dụ có sẵn trong các tệp của cuốn sách.

* + 1. Bộ phân tích chuẩn

Một trình phân tích cú pháp chuẩn sẽ chia văn bản thành các mã thông báo dựa trên ranh giới từ (dấu phân cách khoảng trắng) cũng như dấu câu (dấu phẩy, dấu gạch nối, dấu hai chấm, dấu chấm phẩy, v.v.). Mã sau sử dụng API \_analyze để thực thi trình phân tích cú pháp trên một trường:

POST \_phân tích

{

"text": "Xin chào, thế giới tàn khốc!", "tokenizer": "chuẩn"

}

Điều này tạo ra ba mã thông báo: “Hello”, “cruel” và “world”. Dấu phẩy và khoảng trắng đóng vai trò là dấu phân cách để mã hóa trường thành các mã thông báo riêng lẻ.

Trình phân tích chuẩn chỉ có một thuộc tính có thể tùy chỉnh: max\_token\_length. Thuộc tính này tạo ra các mã thông báo có kích thước mà nó xác định (kích thước mặc định là 255). Chúng ta có thể thiết lập thuộc tính này bằng cách tạo một trình phân tích tùy chỉnh với một trình phân tích mã thông báo tùy chỉnh.

***7.7 Bộ phân tích mã thông báo* 271**

**Liệt kê 7.43 Chỉ mục với trình phân tích tùy chỉnh**

PUT index\_with\_custom\_standard\_tokenizer

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích độ dài mã thông báo tùy chỉnh": {

**Tạo một trình phân tích tùy chỉnh với một con trỏ đến trình phân tích mã thông báo tùy chỉnh**

"tokenizer": "custom\_token\_length\_tokenizer"

}

},

"bộ mã hóa": { "bộ mã hóa\_chiều dài\_mã\_hóa\_tùy\_chỉnh": {

"loại": "chuẩn",

"max\_token\_length": 2

}

}

}

}

}

**Bộ mã hóa tùy chỉnh với max\_token\_length được đặt thành hai ký tự**

Tương tự như cách chúng ta tạo chỉ mục với thành phần tùy chỉnh cho các loại bộ lọc ký tự trong phần 7.7.1, chúng ta có thể tạo chỉ mục với trình phân tích tùy chỉnh bao gồm trình phân tích mã thông báo chuẩn. Sau đó, trình phân tích mã thông báo được mở rộng bằng cách cung cấp kích thước max\_ token\_length (trong danh sách 7.43, độ dài được đặt thành 2). Sau khi chỉ mục được tạo, chúng tacó thể sử dụng API \_analyze để kiểm tra trường.

**Liệt kê 7.44 Kiểm tra kích thước mã thông báo cho trình phân tích mã thông báo**

POST index\_with\_custom\_standard\_tokenizer/\_analyze

{

"text": "Trái phiếu",

"máy phân tích": "custom\_token\_length\_analyzer"

}

Mã này đưa ra hai mã thông báo: “Bo” và “nd”, đáp ứng yêu cầu của chúng tôi về kích thước mã thông báo là hai ký tự.

* + 1. Các trình phân tích ngram và edge\_ngram

Trước khi tìm hiểu về các tokenizer n-gram, chúng ta hãy cùng tóm tắt lại n-gram, n-gram cạnh và shingles. N-gram là các chuỗi từ có kích thước nhất định được chuẩn bị từ một từ nhất định. Lấy ví dụ từ coffee. Các n-gram hai chữ cái, thường được gọi là bi-gram, là “co”, “of”, “ff”, “fe” và “ee”. Tương tự, các tri-gram ba chữ cái là “cof”, “off”, “ffe” và “fee”. Như bạn có thể thấy, n-gram được tạo ra bằng cách trượt cửa sổ chữ cái.

Mặt khác, n-gram cạnh là các từ có các chữ cái được neo ở đầu từ đã cho. Một lần nữa sử dụng coffee làm ví dụ, các n-gram cạnh là “c”, “co”, “cof”, “coff”, “coffe” và “coffee”. Hình 7.13 cho thấy các n-gram và n-gram cạnh này.

**272 CPHẦN7*Phân tích văn bản***

BI-GRAMSTRI-GRAMSEDGE N\_GRAMS



F

F

VÀ

VÀ

VÀ

F

F

THE

THE

C



CỦA

TỐT

VÀ

TỐT

F

F

CỦA

C

CÀ PHÊ

CÀ PHÊ

KÝ ỨC

COF

Đồng bằng

C

THE

THE

F

F

F

CỦA

THE

F

F

VÀ

**Hình 7.13 Biểu diễn hình ảnh của n-gram và n-gram cạnh**

F

VÀ

VÀ

TỐT

Các tokenizer ngram và edge\_ngram phát ra n-gram, như tên gọi của chúng. Hãy cùng xem chúng hoạt động như thế nào.

**THE NGRAM TOKENIZER**

Để sửa lỗi chính tả và các từ bị lỗi, chúng ta thường sử dụng n-gram. Theo mặc định, trình phân tích ngram sẽ phát ra n-gram với kích thước tối thiểu là 1 và kích thước tối đa là 2. Ví dụ, đoạn mã này tạo ra n-gram của từ “Bond”:

POST \_phân tích

{

"text": "Trái phiếu", "tokenizer": "ngram"

}

Đầu ra là[B, Bo, o, on, n, nd, d]. Mỗi n-gram được tạo thành từ một hoặc hai chữ cái:đây là hành vi mặc định. Chúng ta có thể tùy chỉnhmy\_gramVàmax\_gramkích thước bằng cách chỉ định cấu hình.

**Liệt kê 7.45 Mộtngrambộ mã hóa**

PUT index\_with\_ngram\_tokenizer

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích ngram":{

"tokenizer":"ngram\_tokenizer"

}

},

"bộ mã hóa": { "bộ mã hóa ngram":{

* 1. ***Bộ phân tích mã thông báo* 273**

"loại":"ngram", "min\_gram":2,

"max\_gram":3, "token\_chars":[

"thư"

]

}

}

}

}

}

Sử dụng các thuộc tính min\_gram và max\_gram của trình phân tích ngram này (lần lượt được đặt thành 2 và 3), chúng ta có thể cấu hình chỉ mục để tạo ra n-gram. Hãy thử nghiệm tính năng này.

**Liệt kê 7.46 Kiểm trangrambộ mã hóa**

POST index\_with\_ngram\_tokenizer/\_analyze

{

"văn bản": "trái phiếu",

"máy phân tích": "ngram\_analyzer"

}

Điều này tạo ra các n-gram sau: “bo”, “bon”, “on”, “ond” và “nd”. Các n-gram có hai hoặc ba ký tự.

**TCẠNH CỦA ANH\_****TRÌNH TẠO MÃ NGRAM**

Theo cùng một đường dẫn, chúng ta có thể sử dụng trình phân tích edge\_ngram để đưa ra edgen-gram. Đây là mã tạo ra trình phân tích với trình phân tích edge\_ngram:

..

"bộ mã hóa": { "bộ mã hóa\_mặt\_của\_tôi":{

**"kiểu":"edge\_ngram",**

"min\_gram":2,

"max\_gram":6, "token\_chars":["chữ cái","chữ số"]

}

}

Sau khi trình phân tích edge\_ngram được đính kèm vào trình phân tích tùy chỉnh, chúng ta có thể kiểm tra trường bằng API \_analyze:

POST index\_with\_edge\_ngram/\_analyze

{

"văn bản": "trái phiếu",

"máy phân tích": "edge\_ngram\_analyzer"

}

Lời gọi này đưa ra các n-gram cạnh sau: “b”, “bo”, “bon” và “bond”. Lưu ý rằng tất cả các từ đều được neo vào chữ cái đầu tiên.

**274 CPHẦN7*Phân tích văn bản***

* + 1. Các tokenizer khác

Như đã đề cập trước đó, còn có các trình phân tích cú pháp khác và chúng được mô tả tóm tắt trong bảng 7.4. Bạn có thể tìm thấy các ví dụ mã trong các tệp của cuốn sách.

**Bảng 7.4 Sẵn sàng sử dụngbộ phân tích**

|  |  |
| --- | --- |
| **Mã thông báo** | **Sự miêu tả** |
| mẫu uax\_url\_email  khoảng trắng từ khóa chữ thường  đường dẫn\_phân cấp | Chia một trường thành các mã thông báo khi khớp với regex. Mẫu mặc định là chia các từ khi gặp phải ký tự không phải từ.  Phân tích các trường và lưu giữ URL và email. URL và email được trả về nguyên trạng, không có bất kỳ mã thông báo nào.  Chia văn bản thành các mã thông báo khi gặp khoảng trắng Không chạm vào các mã thông báo. Bộ phân tích mã thông báo này trả về văn bản theo nguyên trạng.  Chia văn bản thành các mã thông báo khi gặp phải ký tự không phải chữ cái và viết thường các mã thông báo  Chia văn bản phân cấp như thư mục hệ thống tập tin thành các mã thông báo dựa trên các dấu phân cách đường dẫn |

Thành phần cuối cùng của một bộ phân tích là bộ lọc mã thông báo. Nhiệm vụ của nó là xử lý các mã thông báo do các bộ phân tích mã thông báo đưa ra. Chúng tôi sẽ thảo luận ngắn gọn về bộ lọc mã thông báo trong phần tiếp theo.

#### Bộ lọc mã thông báo

Các mã thông báo do trình phân loại mã thông báo tạo ra có thể cần được làm giàu hoặc cải tiến thêm như viết thường (hoặc viết hoa) các mã thông báo, cung cấp các từ đồng nghĩa, phát triển các từ gốc, loại bỏ dấu nháy đơn hoặc các dấu câu khác, v.v. Bộ lọc mã thông báo hoạt động trên các mã thông báo để thực hiện các chuyển đổi như vậy.

Elasticsearch cung cấp gần 50 bộ lọc mã thông báo và chúng tôi không thể thảo luận tất cả chúng ở đây. Chúng tôi sẽ xem xét một số bộ lọc—bạn có thể tham khảo tài liệu chính thức để tìm hiểu về phần còn lại của các bộ lọc mã thông báo. Chúng tôi có thể kiểm tra bộ lọc mã thông báo bằng cách chỉ cần đính kèm vào một bộ mã thông báo và sử dụng nó trong lệnh gọi API \_analyze.

**Liệt kê 7.47 Thêm bộ lọc mã thông báo với trình phân tích mã thông báo**

NHẬN \_phân tích

{

"tokenizer" : "chuẩn",

**"bộ lọc": ["chữ hoa","đảo ngược"],**

"văn bản" : "trái phiếu"

}

Bộ lọc chấp nhận một mảng các bộ lọc mã thông báo; ví dụ, ở đây chúng tôi cung cấp các bộ lọc chữ hoa và đảo ngược. Đầu ra là “DNOB” (“bond” được viết hoa và đảo ngược).

Chúng ta cũng có thể đính kèm bộ lọc vào một trình phân tích tùy chỉnh, như danh sách sau đây cho thấy. Sau đó, vì chúng ta biết cách đính kèm bộ lọc mã thông báo, chúng ta sẽ xem một vài ví dụ.

***7.8 Mã thông báobộ lọc* 275**

**Danh sách 7.48 Bộ phân tích tùy chỉnh với các bộ lọc bổ sung**

PUT index\_with\_token\_filters

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích\_bộ\_lọc\_token": {

"tokenizer": "chuẩn",

**Xác định một trình phân tích tùy chỉnh**

"bộ lọc": [ "chữ hoa","đảo ngược"]

}

}

}

}

}

**Cung cấp các bộ lọc mã thông báo dưới dạng một mảng các bộ lọc**

* + 1. Bộ lọc phiếu bầu

Như đã giải thích trước đó trong chương, stemming rút gọn các từ thành từ gốc của chúng (ví dụ, bark là từ gốc của barking). Elasticsearch cung cấp một stemmer có sẵn rút gọn các từ thành dạng gốc của chúng. Danh sách tiếp theo hiển thị một ví dụ.

**Liệt kê 7.49 Sử dụng một stemmer làm mã thông báolọc**

POST \_phân tích

{

"tokenizer": "standard", "filter": ["stemmer"], "text": "sủa là cuộc sống của tôi"

}

Khi được thực thi, mã này tạo ra một danh sách các mã thông báo: “bark”, “is”, “my” và “life”. Từ gốc “barking” được chuyển thành “bark”.

* + 1. Bộ lọc ván lợp

*Bệnh zona*là các n-gram từ được tạo ra ở cấp độ mã thông báo (không giống như n-gram và n-gram cạnh phát ra n-gram ở cấp độ chữ cái). Ví dụ, văn bản, “james bond” phát ra các shingles “james” và “james bond”. Sau đây là một ví dụ về bộ lọc shingle:

POST \_phân tích

{

"tokenizer": "chuẩn", "filter": ["shingle"], "text": "java python go"

}

Kết quả là [java, java python, python, python go, go]. Hành vi mặc định của bộ lọc là phát ra unigram và n-gram hai từ. Chúng ta có thể thay đổi hành vi mặc định này bằng cách tạo một trình phân tích tùy chỉnh với bộ lọc shingle tùy chỉnh.

**276 CPHẦN7*Phân tích văn bản***

**Liệt kê 7.50 Tạo một trình phân tích tùy chỉnh chobệnh zona**

PUT index\_with\_shingle

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích bệnh zona":{

"tokenizer":"chuẩn", "bộ lọc":["shingles\_filter"]

**Tạo một trình phân tích tùy chỉnh với bộ lọc shingles**

}

},

"bộ lọc": { "bộ lọc\_bệnh zona":{

"loại":"ván lợp", "kích thước ván lợp tối thiểu":2, "kích thước ván lợp tối đa":3, "đầu ra unigrams":sai

}

**Thuộc tính của bộ lọc ván lợp (ví dụ, kích thước ván lợp tối thiểu và tối đa)**

**Tắt đầu ra của các từ đơn**

}

}

}

}

Gọi mã này trên một số văn bản sẽ tạo ra nhóm gồm hai hoặc ba từ.

**Liệt kê 7.51 Chạyván lợpmáy phân tích**

POST index\_with\_shingle/\_phân tích

{

"text": "java python go", "analyzer": "shingles\_analyzer"

}

Bộ phân tích trả về [java python, java python go, python go] vì chúng tôi đã thiết kế bộ lọc để chỉ tạo ra các cụm từ hai và ba từ. Các cụm từ một từ (cụm từ một từ) như “java”, “python”, v.v. bị xóa khỏi đầu ra vì chúng tôi đã vô hiệu hóa đầu ra của chúng trong bộ lọc của mình.

* + 1. Bộ lọc đồng nghĩa

Chúng tôi đã làm việc với các từ đồng nghĩa trước đó mà không đi sâu vào chi tiết. Từ đồng nghĩa là những từ khác nhau có cùng nghĩa. Ví dụ, nếu người dùng tìm kiếm “football” hoặc “soc- cer” (từ sau là tên gọi của football ở Mỹ), thì cả hai tìm kiếm đều phải trỏ đếnmột trận bóng đá. Bộ lọc từ đồng nghĩa tạo ra một tập hợp các từ để mang lại trải nghiệm người dùng phong phú hơn khi tìm kiếm.

Elasticsearch mong đợi chúng ta cung cấp một tập hợp các từ và các từ đồng nghĩa của chúng bằng cách cấu hìnhphân tích bộ lọc mã thông báo đồng nghĩa. Chúng tôi tạo bộ lọc đồng nghĩa trên cài đặt của chỉ mục như danh sách tiếp theo hiển thị.

***7.8 Mã thông báobộ lọc* 277**

**Liệt kê 7.52 Tạo chỉ mục với bộ lọc đồng nghĩa**

PUT index\_with\_synonyms

{

"cài đặt": {

"Phân tích": {

"bộ lọc": { "bộ lọc từ đồng nghĩa":{

"loại":"từ đồng nghĩa",

"từ đồng nghĩa":[ "bóng đá => bóng bầu dục"]

}

}

}

}

}

Ở đây, chúng tôi tạo một danh sách từ đồng nghĩa (bóng đá được coi là tên thay thế cho bóng bầu dục) liên kết với loại từ đồng nghĩa. Sau khi chỉ mục được cấu hình với bộ lọc này, chúng tôi có thể kiểm tra trường văn bản:

POST index\_with\_synonyms/\_analyze

{

"text": "Bóng đá là gì?", "tokenizer": "standard", "filter": ["synonyms\_filter"]

}

Điều này tạo ra hai mã thông báo: “What's” và “football”. Từ soccer được thay thế bằng từ đồng nghĩa.

Chúng ta có thể cung cấp các từ đồng nghĩa thông qua một tệp trên hệ thống tệp thay vì mã hóa cứng chúng như chúng ta đã làm trong danh sách 7.52. Để làm như vậy, chúng ta cung cấp đường dẫn tệp trong biến synonyms\_path.

**Liệt kê 7.53 Từ đồng nghĩa được tải từ một tệp**

PUT index\_with\_synonyms\_from\_file\_analyzer

{

"cài đặt": {

"Phân tích": {

"máy phân tích": { "máy phân tích từ đồng nghĩa":{

"loại":"chuẩn", "bộ lọc":["từ đồng nghĩa\_từ\_bộ\_lọc\_tệp"]

}

}

,"bộ lọc": { "từ đồng nghĩa từ bộ lọc tệp":{

"loại":"từ đồng nghĩa", "đường dẫn từ đồng nghĩa":"từ đồng nghĩa.txt"

}

**Đường dẫn tương đối của tệp đồng nghĩa**

}

}

}

}

**278 CPHẦN7*Phân tích văn bản***

Hãy chắc chắn tạo một tệp có tên là synonyms.txt trong $ELASTICSEARCH\_HOME/config với nội dung sau.

**Liệt kê 7.54 tệp synonyms.txt với một tập hợp các từ đồng nghĩa**

quan trọng => bắt buộc đẹp => lộng lẫy

Chúng ta có thể gọi tệp bằng đường dẫn tương đối hoặc tuyệt đối; đường dẫn tương đối trỏ đến thư mục con-fig của thư mục cài đặt Elasticsearch. Chúng ta có thể kiểm tra trình phân tích này bằng cách gọi API \_analyze với đầu vào sau.

**Liệt kê 7.55 Kiểm tratừ đồng nghĩa**

POST index\_with\_synonyms\_from\_file\_analyzer/\_analyze

{

"text": "quan trọng", "tokenizer": "chuẩn",

"bộ lọc": ["từ đồng nghĩa\_từ\_bộ\_lọc\_tệp"]

}

Chúng tôi nhận được mã thông báo "bắt buộc" làm phản hồi, chứng minh rằng các từ đồng nghĩa được lấy từ tệp synonyms.txt mà chúng tôi đã thả vào thư mục cấu hình. Bạn có thể thêm nhiều giá trị hơn vào tệp này trong khi Elasticsearch đang chạy và cũng có thể thử chúng.

Vậy là xong! Chương này là cơ bản cho cuộc thảo luận về tìm kiếm mà chúng ta sẽ bắt đầu ở chương tiếp theo. Chúng ta đã hoàn thành tất cả các thông tin cơ bản cần thiết để làm việc với chức năng tìm kiếm một cách chuyên sâu.

#### Bản tóm tắt

* Elasticsearch phân tíchchữcác trường thông qua quy trình phân tích văn bản. Phân tích văn bản được thực hiện bằng cách sử dụng trình phân tích tích hợp hoặc tùy chỉnh. Các trường không phải văn bản không được phân tích.
* Phân tích văn bản bao gồm hai giai đoạn: token hóa và chuẩn hóa. Token hóa chia trường đầu vào thành các từ hoặc token riêng lẻ và chuẩn hóa tăng cường từ (ví dụ, thay đổi từ thành từ đồng nghĩa, loại bỏ từ hoặc xóa từ).
* Elasticsearch sử dụng một mô-đun phần mềm gọi là bộ phân tích để thực hiện phân tích văn bản. Bộ phân tích là một gói được tạo thành từ các bộ lọc ký tự, bộ lọc mã thông báo và bộ mã thông báo.
* Elasticsearch sử dụng mộttiêu chuẩnbộ phân tích theo mặc định nếu không có bộ phân tích nào được nêu rõ trong quá trình lập chỉ mục và tìm kiếm. Atiêu chuẩnmáy phân tích không sử dụng bộ lọc ký tự, mộttiêu chuẩnbộ phân tích mã thông báo và hai bộ lọc mã thông báo (chữ thườngVàdừng lại), mặc dùdừng lạibộ lọc bị tắt theo mặc định.
* Mỗi máy phân tích phải có một (và chỉ một) bộ phân tích mã thông báo, nhưng có thể không có hoặc có nhiều bộ lọc ký tự hoặc mã thông báo.

***Bản tóm tắt* 279**

* + Bộ lọc ký tự giúp loại bỏ các ký tự không mong muốn khỏi các trường nhập. Bộ phân tích mã thông báo hoạt động trên các trường được bộ lọc ký tự xử lý (hoặc trên các trường thô, vì bộ lọc ký tự là tùy chọn). Bộ lọc mã thông báo hoạt động trên các mã thông báo do bộ phân tích mã thông báo phát ra.
  + Elasticsearch cung cấp một số trình phân tích có sẵn. Chúng ta có thể kết hợp và ghép các trình phân tích hiện có với các bộ lọc ký tự hoặc mã thông báo để tạo ra các trình phân tích tùy chỉnh phù hợp với yêu cầu của chúng ta.

*Giới thiệu tìm kiếm*

***Chương này bao gồm***

* Cơ bản của tìm kiếm
* Các loại phương pháp tìm kiếm
* Giới thiệu về Query DSL
* Các tính năng tìm kiếm phổ biến

Đã đến lúc bước vào thế giới tìm kiếm. Cho đến nay, chúng ta đã xem xét việc chuẩn bị Elasticsearch bằng dữ liệu và chương cuối cùng đã thảo luận về cơ chế phân tích các trường văn bản. Chúng ta đã nếm trải việc tìm kiếm dữ liệu bằng một tập hợp các truy vấn, nhưng chúng ta không xem xét tìm kiếm chuyên sâu hoặc các biến thể tìm kiếm chi tiết. Các chương từ 8 đến 12 dành riêng cho tìm kiếm.

Tìm kiếm là chức năng cốt lõi của Elasticsearch và trả lời các truy vấn của người dùng một cách hiệu quả và hiệu suất cao. Khi dữ liệu được lập chỉ mục và có sẵn để tìm kiếm, người dùng có thể đặt nhiều câu hỏi khác nhau. Ví dụ, nếu tìm kiếm trên trang web hiệu sách trực tuyến giả tưởng của chúng tôi được hỗ trợ bởi Elasticsearch, chúng tôi có thể mong đợi nhiều truy vấn từ khách hàng. Chúng có thể bao gồm từ các truy vấn đơn giản như tìm sách dựa trên từ tìm kiếm tiêu đề đến các truy vấn phức tạp tìm kiếm sách khớp với nhiều tiêu chí: một phiên bản cụ thể, được xuất bản trong một khoảng thời gian, bìa cứng, xếp hạng đánh giá cao hơn 4,5 trên

**280**

***8.1 Tổng quan* 281**

trong số 5, giá thấp hơn một số tiền nhất định, v.v. Giao diện người dùng có thể hỗ trợ nhiều tiện ích khác nhau như danh sách thả xuống, thanh trượt, hộp kiểm, v.v. để cho phép lọc tìm kiếm sâu hơn nữa.

Chương này giới thiệu về tìm kiếm và các tính năng cơ bản mà chúng ta có thể sử dụng khi tìm kiếm. Trước tiên, chúng ta thảo luận về cơ chế tìm kiếm: cách xử lý yêu cầu tìm kiếm và cách tạo phản hồi rồi gửi đến máy khách. Sau đó, chúng ta xem xét các nguyên tắc cơ bản của tìm kiếm: API tìm kiếm và bối cảnh thực hiện truy vấn tìm kiếm. Chúng ta phân tích yêu cầu và phản hồi để tìm hiểu sâu hơn về các thành phần của chúng.

Chúng tôi cũng xem xét các loại tìm kiếm, yêu cầu URI và Query DSL, cũng như lý do tại sao chúng tôi thích Query DSL hơn phương pháp yêu cầu URI. Cuối cùng, chúng tôi xem xét các tính năng tìm kiếm cắt ngang như tô sáng, phân trang, giải thích và thao tác các trường phản hồi.

**GHI CHÚ** Mã cho chương này có sẵn trên GitHub ([http://mng.bz/](http://mng.bz/e1yZ) [e1yZ](http://mng.bz/e1yZ)) và trên trang web của cuốn sách ([https://www.manning.com/books/elastic](https://www.manning.com/books/elasticsearch-in-action-second-edition) [tìm kiếm-trong-hành-động-phiên-bản-thứ-hai](https://www.manning.com/books/elasticsearch-in-action-second-edition)).

#### Tổng quan

Elasticsearch hỗ trợ chức năng tìm kiếm đơn giản và tìm kiếm nâng cao xem xét nhiều tiêu chí, bao gồm truy vấn không gian địa lý. Nói chung, Elasticsearchxử lý hai biến thể của tìm kiếm: tìm kiếm có cấu trúc và tìm kiếm không có cấu trúc. Chúng tôi đã xem xét các biến thể này trong các chương trước, vì vậy đây khá là tóm tắt; bạn có thể bỏ qua phần này.

Tìm kiếm có cấu trúc trả về kết quả không có điểm liên quan nào. Elasticsearch sẽ tìm nạp các tài liệu nếu chúng khớp chính xác và không quan tâm đến việc chúng có khớp chặt hay khớp tốt như thế nào. Tìm kiếm các chuyến bay giữa một tập hợp các ngày, sách bán chạy nhất trong một chương trình khuyến mại cụ thể, v.v. thuộc danh mục này. Khi thực hiện tìm kiếm, Elasticsearch sẽ kiểm tra xem kết quả khớp có thành công hay không: ví dụ, có hay không có chuyến bay nằm trong phạm vi ngày đã cho và có hay không có sách bán chạy nhất. Không có gì thuộc danh mục có thể. Loại tìm kiếm có cấu trúc này được cung cấp bởi các truy vấn cấp thuật ngữ trong Elasticsearch.

Mặt khác, trong tìm kiếm không có cấu trúc, Elasticsearch sẽ truy xuất các kết quả có liên quan chặt chẽ đến truy vấn. Kết quả được chấm điểm dựa trên mức độ liên quan của chúng với tiêu chí: kết quả có liên quan cao sẽ được chấm điểm cao hơn và do đó được xếp ở đầu danh sách. Tìm kiếm trên các trường văn bản sẽ cho kết quả có liên quan và Elasticsearch cung cấp tìm kiếm toàn văn để tìm kiếm dữ liệu không có cấu trúc.

Chúng tôi giao tiếp với công cụ Elasticsearch bằng cách sử dụng RESTful API để thực hiện các truy vấn. Truy vấn tìm kiếm được viết bằng cú pháp truy vấn đặc biệt gọi là Query DSL (ngôn ngữ dành riêng cho miền) hoặc một tiêu chuẩn URL gọi là query requests. Khi một truy vấn được đưa ra, bất kỳ nút nào có sẵn trong cụm sẽ chọn yêu cầu và xử lý nó. Phản hồi được trả về dưới dạng đối tượng JSON, với một mảng các tài liệu riêng lẻ làm kết quả trong đối tượng.

**282 CPHẦN8*Giới thiệu tìm kiếm***

Nếu truy vấn được thực hiện trên các trường văn bản, mỗi kết quả riêng lẻ được trả về với điểm liên quan. Điểm càng cao, mức độ liên quan càng lớn (có nghĩa là tài liệu kết quả khớp rất chặt chẽ). Kết quả được sắp xếp theo thứ tự giảm dần, với kết quả có điểm cao nhất ở trên cùng.

Không phải mọi kết quả trong phản hồi đều chính xác. Cũng giống như đôi khi chúng ta có thể mong đợi kết quả không chính xác hoặc không liên quan khi tìm kiếm trên Google, Elasticsearch có thể không trả về 100% kết quả có liên quan. Điều này là do Elasticsearch sử dụng hai chiến lược nội bộ, độ chính xác và độ thu hồi, ảnh hưởng đến tính liên quan của kết quả.

*Độ chính xác*là tỷ lệ phần trăm các tài liệu có liên quan được tìm thấy trên một tập hợp các tài liệu có sẵn trong chỉ mục và độ thu hồi là tỷ lệ phần trăm các tài liệu có liên quan được tìm thấy trên một tập hợp các tài liệu có thể áp dụng (tất cả các tài liệu có liên quan). Chúng tôi thảo luận chi tiết về độ chính xác và độ thu hồi trong chương 10.

Nhiều tính năng chung để tìm kiếm bất kể loại truy vấn nào chúng ta chọn (toàn văn, cấp thuật ngữ, không gian địa lý, v.v.). Chúng ta thảo luận về các tính năng tìm kiếm này trong suốt chương và xem các ứng dụng của chúng trong một vài chương tiếp theo khi chúng ta làm việc với tìm kiếm và tổng hợp. Phần sau đây giải thích cơ chế tìm kiếm: cách Elasticsearch hoạt động trên truy vấn tìm kiếm để lấy kết quả phù hợp.

#### Tìm kiếm hoạt động như thế nào?

Có rất nhiều điều xảy ra khi người dùng gọi truy vấn tìm kiếm trên Elasticsearch. Mặc dù chúng ta đã đề cập đến cơ chế trước đó, hãy cùng tóm tắt lại. Hình 8.1 cho thấy cách công cụ thực hiện tìm kiếm ở chế độ nền.

**Nút A**Trách nhiệm của:

* + - Tìm các nút có liên quan.



* + - Yêu cầu họ thực hiện truy vấn.
    - Điều phối các yêu cầu.
    - Nhận kết quả từ các nút khác.
    - Tổng hợp các kết quả.

**Nhóm sao chép**

q

Nút B Mảnh 2

Nút điều phối

Nút điều phối

**Nút này sẽ không được tham khảo vì không có phân đoạn liên quan nào tồn tại trên đó.**

Nút E

q

Nút điều phối

kết quả

**kết quả = r1 + r2 + r3**

**Mỗi nút là một nút điều phối, nhưng chỉ có một nút trở nên “hoạt động” cho một yêu cầu.**

r1 q

Nút A

Mảnh vỡ 1

r2

q

r3

Nút C Shard 3

Nút điều phối

Nút D Shard 4

Nút điều phối

**Nút A cũng là một nút dữ liệu với**

**Shard 1 trên đó. Nó lấy kết quả từ kho lưu trữ của riêng nó để hợp nhất dữ liệu với dữ liệu của những kho lưu trữ khác.**

**Hình 8.1 Một yêu cầu tìm kiếm thông thường và cơ chế hoạt động của tìm kiếm**

* 1. ***Mẫu phimdữ liệu* 283**

Khi nhận được yêu cầu tìm kiếm từ người dùng hoặc máy khách, công cụ sẽ chuyển tiếp yêu cầu đó đến một trong các nút khả dụng trong cụm. Theo mặc định, mọi nút trong cụm được chỉ định cho một vai trò điều phối, do đó, mọi nút đều đủ điều kiện để nhận yêu cầu của máy khách theo cơ sở luân phiên. Khi yêu cầu đến nút điều phối, nó sẽ xác định các nút mà các mảnh của tài liệu áp dụng tồn tại.

Trong hình 8.1, Node A là nút điều phối nhận được yêu cầu của máy khách. Nó được chọn làm nút điều phối không vì lý do nào khác ngoài mục đích trình diễn. Sau khi được chọn cho vai trò chủ động (điều phối), nó sẽ chọn một nhóm sao chép với một tập hợp các mảnh và bản sao trên các nút riêng lẻ trong một cụm bao gồm dữ liệu. Hãy nhớ rằng, một chỉ mục được tạo thành từ các mảnh và mỗi mảnh có thể tồn tại độc lập trên các nút khác. Trong ví dụ của chúng tôi, chỉ mục được tạo thành từ bốn mảnh (1, 2, 3 và 4) tồn tại trên các Nút A đến D, tương ứng.

Node A xây dựng yêu cầu truy vấn để gửi đến các node khác, yêu cầu chúng thực hiện tìm kiếm. Khi nhận được yêu cầu, node tương ứng thực hiện yêu cầu tìm kiếm trên phân đoạn của nó. Sau đó, nó trích xuất tập hợp kết quả hàng đầu và phản hồi cho bộ điều phối đang hoạt động với các kết quả đó. Sau đó, bộ điều phối đang hoạt động sẽ hợp nhất và sắp xếp dữ liệu trước khi gửi đến máy khách dưới dạng kết quả cuối cùng.

Nếu điều phối viên có vai trò là một nút dữ liệu, nó cũng sẽ đào sâu vào kho lưu trữ của mình để lấy kết quả. Không phải mọi nút nhận được yêu cầu đều nhất thiết là một nút dữ liệu. Tương tự, không phải mọi nút đều được mong đợi là một phần của nhóm sao chép cho truy vấn tìm kiếm này. Bây giờ, hãy dành một chút thời gian để tải một số dữ liệu phim vào công cụ Elasticsearch của chúng ta.

#### Dữ liệu mẫu phim

Hãy tạo một số dữ liệu thử nghiệm phim và ánh xạ phim cho chương này. Vì chúng ta không muốn Elasticsearch suy ra các kiểu trường, chúng ta cung cấp các kiểu dữ liệu có liên quan cho từng trường dưới dạng ánh xạ khi chúng ta tạo chỉ mục (đặc biệt là các trường release\_date và duration, không thể là trường văn bản). Danh sách sau đây hiển thị ánh xạ chỉ mục phim (nếu chỉ mục này đã tồn tại, hãy xóa nó đối với các ví dụ của chương này bằng cách phát hành DELETE movies).

**Liệt kê 8.1 Sơ đồ ánh xạ chophimlãnh địa**

ĐẶT phim

{

"ánh xạ": {

**Phimchỉ số**

**Sơ đồ ánh xạ**

"thuộc tính": { "tiêu đề": {

"kiểu": "văn bản", "trường": {

"nguyên bản": {

**Các trường và loại của chúng**

**Xây dựng đa trường**

"loại": "từ khóa"

}

}

},

"Tóm tắt": {

**284 CPHẦN8*Giới thiệu tìm kiếm***

"loại": "văn bản"

},

"diễn viên": {

"loại": "văn bản"

},

"giám đốc": {

"loại": "văn bản"

},

"xếp hạng": {

"kiểu": "nửa\_float"

},

"ngày phát hành": { "loại": "ngày",

"định dạng": "dd-MM-yyyy"

},

"chứng chỉ": { "loại": "từ khóa"

},

"thể loại": {

"loại": "văn bản"

}

}

}

}

Bảng 8.1 cho thấy một số yếu tố đáng chú ý từ bản đồ này. Các thuộc tính còn lại đều tự giải thích.

**Bảng 8.1 Một sốphimcác trường và kiểu dữ liệu tương ứng của chúng**

|  |  |
| --- | --- |
| **Cánh đồng** | **Các kiểu dữ liệu được khai báo** |
| tiêu đề  ngày phát hành chứng chỉ | văn bản vàtừ khóa  ngày tháng theo định dạng dd-MM-yyyytừ khóa |

Với việc lập bản đồ cho miền phim, nhiệm vụ tiếp theo là lập chỉ mục dữ liệu mẫu bằng API \_bulk. Danh sách tiếp theo cho thấy API này đang hoạt động với một mẫu dữ liệu.

**Liệt kê 8.2 Liệt kê 8.2 Lập chỉ mục dữ liệu phim mẫu bằng cách sử dụng\_số lượng lớnGiao diện lập trình ứng dụng (API)**

ĐẶT \_số lượng lớn

**\_Số lượng lớn**

**Giao diện lập trình ứng dụng (API) Tài liệu với một**

{"index":{"\_index":"phim","\_id":"1"}}

**ID của1 cholập chỉ mục Tài liệu**

{"title":"Nhà tù Shawshank","genre":"Phim chính kịch",..}

{"index":{"\_index":"phim","\_id":"2"}}

{"title":"Bố già","genre":"Tội phạm, Chính kịch","...}

{"index":{"\_index":"phim","\_id":"3"}}

**chính nó**

**Tài liệu có ID là 2 để lập chỉ mục**

* 1. ***Tìm kiếmcơ bản* 285**

Để ngắn gọn, chúng tôi chỉ trình bày một phần của kịch bản. Toàn bộ kịch bản có sẵn trong các tệp của cuốn sách.

**GHI CHÚ**Để tránh việc có quá nhiều mô hình miền và làm hỏng dữ liệu mẫu, chúng tôi sử dụng cùng một dữ liệu phim trong chương 8 và 9.

Sau khi hiểu được cơ chế tìm kiếm và dữ liệu mẫu, chúng ta hãy chuyển sang những nguyên tắc cơ bản của tìm kiếm.

#### Tìm kiếm cơ bản

Bây giờ chúng ta đã biết cách thức hoạt động bên trong của tìm kiếm, hãy cùng xem API tìm kiếm và cáchđể gọi công cụ thực hiện truy vấn tìm kiếm. Elasticsearch hiển thị điểm cuối \_search để giao tiếp với nó nhằm thực hiện truy vấn tìm kiếm. Hãy cùng xem xét điểm cuối chi tiết.

* + 1. Điểm cuối \_search

Elasticsearch cung cấp API RESTful để truy vấn dữ liệu: cụ thể là điểm cuối \_search.Chúng tôi sử dụng GET/POST để gọi điểm cuối này, truyền tham số truy vấn cùng với yêu cầu hoặc nội dung yêu cầu. Các truy vấn chúng tôi xây dựng phụ thuộc vào loại dữ liệu chúng tôi đang tìm kiếm. Có hai cách để truy cập điểm cuối tìm kiếm:

* + - * *Yêu cầu URI*—Chúng tôi truyền truy vấn tìm kiếm cùng với điểm cuối dưới dạng tham sốcho truy vấn. Ví dụ,GET movies/\_search?q=title:Bố giàtìm tất cả các bộ phim có từ Bố già trong tiêu đề (ví dụ: bộ ba phim Bố già).
      * *Truy vấn DSL*—Elasticsearch triển khai DSL để tìm kiếm. Tiêu chí tìm kiếm được gói trong phần thân JSON được gửi đến máy chủ cùng với URL yêu cầu. Kết quả cũng được gói trong đối tượng JSON. Chúng ta có thể cung cấp một truy vấn duy nhất hoặc kết hợp nhiều truy vấn, tùy thuộc vào yêu cầu của chúng ta. (Query DSL cũng là cơ chế gửi các truy vấn tổng hợp đến công cụ. Chúng ta sẽ xem xét thêm về tổng hợp sau trong chương này.) Sau đây là ví dụ về cùng một yêu cầu đểlấy tất cả các bộ phim có từ Bố già trongtiêu đềcánh đồng:

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"title": "Bố già"

}

}

}

Mặc dù cả hai cách tiếp cận đều hữu ích theo cách riêng của chúng, Query DSL mạnh mẽ và giàu tính năng. Đây là cơ chế truy vấn hạng nhất và dễ viết các tiêu chí phức tạp hơn khi sử dụng Query DSL so với cơ chế yêu cầu URI. Chúng ta sẽ xem xét các lệnh gọi khác nhau trong chương này và các chương từ 9 đến 12 và làm việc rộng rãi hơn với Query DSL so với phương pháp yêu cầu URI.

**286 CPHẦN8*Giới thiệu tìm kiếm***

**GHI CHÚ**Query DSL giống như một con dao quân đội Thụy Sĩ khi nói đến việc nói chuyện với Elasticsearch và đây là lựa chọn được ưa chuộng. Nhóm Elasticsearch đã phát triển DSL này đặc biệt để làm việc với công cụ. Bất kỳ điều gì và mọi thứ chúng ta muốn hỏi Elasticsearch đều có thể được truy xuất bằng Query DSL.

Đừng lo lắng nếu bạn không hiểu các truy vấn tìm kiếm và cách chúng được mã hóa. Chúng tôi sẽ xem xét một số ví dụ trong chương này và thảo luận chi tiết về chúng trong các chương tiếp theo.

* + 1. Truy vấn so với bối cảnh lọc

Chúng ta cần xem xét một khái niệm cơ bản khác: ngữ cảnh thực thi. Về mặt nội bộ, Elasticsearch sử dụng ngữ cảnh thực thi khi chạy tìm kiếm. Ngữ cảnh thực thi này có thể là ngữ cảnh bộ lọc hoặc ngữ cảnh truy vấn. Tất cả các truy vấn được đưa ra cho Elasticsearch đều được thực hiện trong một trong những ngữ cảnh này. Chúng ta không thể yêu cầu Elasticsearch áp dụng một loại ngữ cảnh nhất định—truy vấn của chúng ta cho phép Elasticsearch quyết định và áp dụng ngữ cảnh phù hợp. Ví dụ, nếu chúng ta đang chạy truy vấn khớp, truy vấn đó phải được thực thi trong ngữ cảnh truy vấn, trong khi truy vấn bộ lọc được thực thi trong ngữ cảnh bộ lọc. Hãy thực thi một vài truy vấn để hiểu ngữ cảnh mà truy vấn được thực thi.

**HỏiUERY BỐI CẢNH**

Chúng tôi sử dụng truy vấn khớp để tìm kiếm tài liệu bằng cách khớp từ khóa với giá trị của trường. Danh sách sau đây là truy vấn khớp đơn giản tìm kiếm “Godfather” trong trường tiêu đề.

**Liệt kê 8.3 Truy vấn khớp được thực hiện trong ngữ cảnh truy vấn**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"title": "Bố già"

}

}

}

Mã này trả về hai bộ phim Bố già của chúng ta, như mong đợi. Tuy nhiên, nếu chúng ta xem xét kỹ kết quả, mỗi bộ phim có điểm liên quan tương ứng:

"lượt truy cập" : [{

...

**"\_điểm" : 2.879596**

"\_nguồn" : {

"title": "Bố già"

...

}

},

{

...

**"\_điểm" : 2.261362**

"\_nguồn" : {

* 1. ***Tìm kiếmcơ bản* 287**

"title": "Bố già: Phần II"

...

}

}]

Đầu ra cho biết truy vấn đã được thực hiện trong bối cảnh truy vấn vì truy vấn không chỉ xác định xem nó có khớp với tài liệu hay không mà còn xác định tài liệu đó khớp đến mức nào.

Tại sao điểm số cho kết quả thứ hai (2.261362) lại thấp hơn một chút so với điểm số cho kết quả đầu tiên (2.879596)? Bởi vì thuật toán liên quan của công cụ tìm kiếm đã tìm thấy “Godfa- ther” trong tiêu đề có hai từ (“the”, “godfather”), và kết quả đó được xếp hạng cao hơn kết quả trong tiêu đề có bốn từ (“the”, “godfather”, “part”, “II”).

**GHI CHÚ**Các truy vấn trên trường tìm kiếm toàn văn chạy trong ngữ cảnh truy vấn vì chúng dự kiến ​​sẽ có điểm số liên quan đến từng tài liệu khớp.

Mặc dù việc lấy kết quả với điểm liên quan là tốt cho hầu hết các trường hợp sử dụng, đôi khi chúng ta không cần tìm hiểu xem tài liệu khớp tốt như thế nào. Thay vào đó, chúng ta chỉ muốn biết liệu có khớp hay không. Đây là lúc ngữ cảnh bộ lọc phát huy tác dụng.

**FBỐI CẢNH ILTER**

Hãy viết lại truy vấn từ danh sách 8.3, nhưng điều nàythời gian bọc truy vấn khớp của chúng tôi trong một bool

truy vấn với mộtlọcđiều khoản.

**Liệt kê 8.4 Abooltruy vấn không có điểm**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"bool": {

"lọc": [{

"cuộc thi đấu": {

"title": "Bố già"

}

}

]

}

}

}

Trong danh sách này, kết quả không có điểm (điểm được đặt thành 0.0) vì truy vấn của chúng ta cung cấp cho Elasticsearch một manh mối rằng nó phải được chạy trong ngữ cảnh bộ lọc. Nếu chúng ta không quan tâm đến việc chấm điểm tài liệu, chúng ta có thể yêu cầu Elasticsearch chạy truy vấn trong ngữ cảnh bộ lọc bằng cách gói truy vấn trong một mệnh đề bộ lọc, như chúng ta đã làm ở đây.

Lợi ích chính của việc chạy truy vấn trong ngữ cảnh này là vì Elasticsearch không cần tính điểm cho kết quả tìm kiếm được trả về nên có thể tiết kiệm một số chu kỳ tính toán. Các truy vấn lọc này có tính bất biến hơn, do đó Elasticsearch lưu trữ chúng trong bộ nhớ đệm để có hiệu suất tốt hơn.

**288 CPHẦN8*Giới thiệu tìm kiếm***

**Truy vấn hợp chất**

MỘTbooltruy vấn là một truy vấn hợp chất với một số mệnh đề (phải,phải\_không,nên, Vàlọc) để gói các truy vấn lá.phải\_khôngđiều khoản mâu thuẫn vớiphảiý định của truy vấn. Cả hailọcVàphải\_khôngcác mệnh đề được thực hiện trong mộtlọcbối cảnh. Chúng ta có thểcũng sử dụnglọcngữ cảnh trong truy vấn bool,chúng ta cũng có thể sử dụng nó trong mộtđiểm số hằng số(tên là một câu hỏi gợi ý).

Cácđiểm số hằng sốtruy vấn là một truy vấn hợp chất khác trong đó chúng ta có thể đính kèm một truy vấn tronglọcmệnh đề. Truy vấn sau đây cho thấy điều này trong hành động:

NHẬN phim/\_tìm kiếm

{

"truy vấn": { "điểm\_hằng\_số": {

"lọc": {

"cuộc thi đấu": {

"title": "Bố già"

}

}

}

}

}

Chúng ta sẽ xem xét các truy vấn hợp chất trong chương 11.

Biết được ngữ cảnh thực thi giúp chúng ta tiến gần hơn một bước đến việc hiểu được hoạt động bên trong của công cụ Elasticsearch. Ngữ cảnh thực thi bộ lọc giúp tạo ra các truy vấn hiệu suất vì không cần thêm nỗ lực chạy thuật toán liên quan. Chúng ta hãy xem các ví dụ minh họa các ngữ cảnh này trong các chương sau.

Bây giờ chúng ta đã có cái nhìn tổng quan về tìm kiếm nói chung (và dữ liệu tìm kiếm trong tay), hãy cùng xem yêu cầu tìm kiếm là gì và phân tích kết quả (phản hồi tìm kiếm).

#### Giải phẫu của một yêu cầu và một phản hồi

Chúng ta đã xem nhanh yêu cầu tìm kiếm và phản hồi trong vài chương trước mà không cần lo lắng về các chi tiết thuộc tính và giải thích. Tuy nhiên, điều quan trọng là phải hiểu các đối tượng yêu cầu và phản hồi, xây dựng đối tượng truy vấn mà không có lỗi và hiểu ý nghĩa của các thuộc tính có trong phản hồi. Trong phần này, chúng ta sẽ đi sâu vào các đối tượng yêu cầu và phản hồi.

* + 1. Yêu cầu tìm kiếm

Truy vấn tìm kiếm có thể được thực hiện bằng phương pháp yêu cầu URI hoặc Query DSL. Như đã thảo luận trước đó, chúng tôi tập trung vào Query DSL trong cuốn sách này vì nó mạnh mẽ và biểu cảm hơn. Hình 8.2 cho thấy cấu trúc của một yêu cầu tìm kiếm.

* 1. ***Giải phẫu của một yêu cầu vàmột phản hồi* 289**

**Giao thức HTTPLẤYhoạt động**



**Phạm vi tìm kiếm**



**Các\_tìm kiếmđiểm cuối được sử dụng cho cả tìm kiếm và tổng hợp**

**Cáctruy vấnthành phần nơi**

**Đối tượng yêu cầu tìm kiếm là JSON**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

**người dùng được yêu cầu chỉ định loại tìm kiếm và tiêu chí**

**vật thể được gắn vào**

**đến URI yêu cầu.**

"title": "Bố già"

}

}

**Loại truy vấn tìm kiếm**

} **Tiêu chí truy vấn tìm kiếm: các giá trị để tìm kiếm dựa trên một trường (hoặc nhiều trường)**

**Hình 8.2 Các thành phần của một yêu cầu tìm kiếm**

Phương thức GET là hành động HTTP chỉ định ý định của chúng ta: lấy dữ liệu từ máy chủ với các chi tiết yêu cầu được chuyển trong phần thân. Có một trường phái cho rằng phương thức GET trong kiến ​​trúc RESTful không nên gửi tham số bằng phần thâncủa yêu cầu và thay vào đó, chúng ta nên sử dụng phương thức POST nếu chúng ta đang truy vấn máy chủ. Nhưng Elasticsearch triển khai yêu cầu phương thức GET chấp nhận một phần thân, giúp xây dựng các tham số truy vấn. Chúng ta có thể thay thế GET bằng POST, vì cả GET hoặc POST đều hoạt động trên tài nguyên theo cùng một cách.

**GHI CHÚ**Internet tràn ngập những cuộc tranh luận sôi nổi về việc sử dụng GET có và không có payload. Elasticsearch sử dụng phương pháp sử dụng GET có payload. (Nếu bạn muốn hiểu các trường phái tư tưởng, hãy tìm kiếm “HTTP get with body” trên internet). Trong cuốn sách này, chúng tôi sử dụng GET có payload cho các truy vấn tìm kiếm và tổng hợp, mặc dù bạn có thể thay thế GET bằng POST nếu bạn không thoải mái khi sử dụng phương thức GET có body.

Phạm vi tìm kiếm trong hành động GET (hoặc POST) xác định chỉ mục hoặc bí danh mà công cụ sử dụng để thực hiện tìm kiếm. Chúng ta cũng có thể bao gồm nhiều chỉ mục, nhiều bí danh hoặc không có chỉ mục trong URL này. Để chỉ định nhiều chỉ mục (hoặc bí danh), hãy nhập tên chỉ mục (hoặc bí danh) được phân tách bằng dấu phẩy. Không cung cấp chỉ mục hoặc bí danh nào trên yêu cầu tìm kiếm sẽ yêu cầu tìm kiếm chạy trên tất cả các chỉ mục trong cụm. Ví dụ: nếu chúng ta có một cụm có 10 chỉ mục, chạy truy vấn tìm kiếm như GET \_search {...} mà không chỉ định tên chỉ mục hoặc bí danh sẽ tìm kiếm tất cả các tài liệu khớp trên tất cả 10 chỉ mục.

Đối tượng yêu cầu tìm kiếm hoặc tải trọng là một đối tượng JSON bao gồm các chi tiết yêu cầu. Các chi tiết yêu cầu bao gồm thành phần truy vấn và có thể bao gồm các thành phần kháccác thành phần như kích thước liên quan đến phân trang và các thuộc tính hình thức, danh sách chỉ ra trường nguồn nào sẽ trả về trong phản hồi, tiêu chí sắp xếp, tô sáng, v.v. Chúng tôi sẽ thảo luận về các tính năng riêng lẻ sau trong chương này.

**290 CPHẦN8*Giới thiệu tìm kiếm***

Thành phần chính của yêu cầu là truy vấn. Nhiệm vụ của truy vấn là soạn thảocâu hỏi cần được trả lời. Nó thực hiện điều này bằng cách tạo một đối tượng truy vấn xác định loại truy vấn và đầu vào bắt buộc của nó. Chúng ta có thể chọn từ nhiều loại truy vấn phục vụ nhiều tiêu chí tìm kiếm khác nhau. Chúng ta sẽ tìm hiểu chi tiết về các loại truy vấn này trong một vài chương tiếp theo.

Chúng ta có thể tạo các truy vấn cụ thể cho các trường hợp sử dụng cụ thể, từ các truy vấn khớp và cấp thuật ngữ đến các truy vấn đặc biệt như hình dạng địa lý. Các kiểu truy vấn có thể xây dựng một truy vấn lá đơn giản nhắm mục tiêu đến một yêu cầu tìm kiếm duy nhất hoặc xây dựng một yêu cầu phức tạp bằng cách sử dụng các truy vấn hợp chất để xử lý tìm kiếm nâng cao thông qua các mệnh đề logic. Bây giờ chúng ta đã quen thuộc với cấu trúc của một yêu cầu tìm kiếm, đã đến lúc phân tích phản hồi.

* + 1. Tìm kiếm phản hồi

Chúng ta đã xem xét các phản hồi tìm kiếm trước đó trong cuốn sách nhưng chưa bao giờ đề cập đến chi tiết. Hình 8.3 cho thấy một phản hồi điển hình; chúng ta hãy thảo luận ngắn gọn về các thuộc tính này để hiểu chúng đại diện cho điều gì.

**Thời gian thực hiện tìm kiếm truy vấn trên các nút**

**Truy vấn đã hết thời gian chưa?**

**Bao nhiêushards có trả về kết quả thành công không?**



**Bên ngoàilượt truy cậpđối tượng vớitổng cộng,điểm tối đavà bên tronglượt truy cậpmảng**

**Bên tronglượt truy cậpđối tượng bao gồm siêu dữ liệu cho mỗi tài liệu và bản thân tài liệu đó**

**Tổng số kết quả**

**Điểm số cao nhất trêntất cả các tài liệu đã trả lại**

**Siêu dữ liệu tài liệu**

**Tài liệu nguồn thực tế**

**Hình 8.3 Thành phầncủa một phản hồi tìm kiếm**

* 1. ***Yêu cầu URItìm kiếm* 291**

Thuộc tính took, được đo bằng mili giây, biểu thị thời gian cần thiết để hoàn tất yêu cầu tìm kiếm. Thời gian này được đo từ khi một nút điều phối nhận được yêu cầu đến thời điểm nó tổng hợp phản hồi trước khi gửi lại cho máy khách. Nó không bao gồm thời gian sắp xếp/bỏ sắp xếp từ máy khách đến máy chủ.

Thuộc tính timed\_out là cờ Boolean cho biết phản hồi có kết quả một phần hay không, nghĩa là có bất kỳ phân đoạn nào không phản hồi kịp thời. Ví dụ, nếu chúng ta có ba phân đoạn và một phân đoạn không trả về kết quả, phản hồi bao gồm kết quả từ hai phân đoạn nhưng cho biết trạng thái của phân đoạn không thành công trong đối tượng tiếp theo theo thuộc tính \_shards.

Thuộc tính shards cung cấpsố lượng các mảnh đã thực hiện thành công truy vấn và trả về kết quả và số lượng không thành công. Tổng số trường là sốcủa các mảnh dự kiến ​​sẽ được tìm kiếm và trường thành công biểu thị các mảnh đã trả về dữ liệu. Mặt khác, cờ lỗi biểu thị các mảnh không thành công trong quá trình thực hiện truy vấn tìm kiếm; cờ được biểu thị bằng thuộc tính lỗi.

Thuộc tính hits (gọi là outer hits) bao gồm thông tin về kết quả. Thuộc tính này chứa một trường hits khác (gọi là inner hits). Đối tượng outer hits bao gồm các kết quả trả về, điểm tối đa và tổng kết quả. Điểm tối đa, được biểu thị bằng thuộc tính max\_score, là tài liệu trả về có điểm cao nhất. Đối tượng inner hits bao gồm các kết quả (các tài liệu thực tế). Đây là một mảng của tất cả các tài liệu riêng lẻ được sắp xếp theo mức độ liên quan theo thứ tự giảm dần. Mỗi tài liệu nhận được thuộc tính \_score này nếu truy vấn được thực thi trong ngữ cảnh truy vấn.

Chúng tôi đã đề cập đến hai loại yêu cầu mà chúng tôi có thể tạo cho truy vấn đặt câu hỏi: Yêu cầu URI và DSL truy vấn. Hai phần tiếp theo sẽ thảo luận chi tiết về những yêu cầu này.

**GHI CHÚ**Như đã nêu trước đó, vì tính linh hoạt và các tính năng của Query DSL, chúng tôi sử dụng nó làm cơ chế tìm kiếm trong cuốn sách này. Tuy nhiên, để đầy đủ, chúng tôi sẽ thảo luận ngắn gọn về tìm kiếm yêu cầu URI trong phần sau. Theo cách này, bạn có thể thử nghiệm bằng cách tạo các phương thức yêu cầu URI tương đương nếu bạn muốn tạo các truy vấn tìm kiếm liên chức năng.

#### 8.6 Tìm kiếm yêu cầu URI

Phương pháp yêu cầu URI là một cách dễ dàng để tìm kiếm các truy vấn đơn giản. Chúng tôi gọi

\_search endpoint bằng cách truyền các tham số bắt buộc. Cú pháp sau đây cho biết cách triển khai tìm kiếm này:

GET|POST <tên\_index\_của\_bạn>/\_tìm\_kiếm?q=<tên:giá\_trị> VÀ|HOẶC <tên:giá\_trị>

Điểm cuối \_search được gọi trên chỉ mục của chúng tôi (hoặc nhiều chỉ mục) với một truy vấn trongdạng q=<tên:giá trị>. Lưu ý rằng truy vấn được thêm vào sau điểm cuối \_search bằng dấu phân cách dấu chấm hỏi (?). Với phương pháp này, chúng ta truyền vào các tham số truy vấn được đính kèm vào URL dưới dạng cặp tên:giá trị. Hãy đưa ra một số truy vấn tìm kiếm bằng phương pháp này.

**292 CPHẦN8*Giới thiệu tìm kiếm***

* + 1. Tìm kiếm phim theo tiêu đề

Giả sử chúng ta muốn tìm phim bằng cách tìm kiếm một từ trong trường tiêu đề (“Godfather”, ví dụ). Chúng ta sử dụng điểm cuối \_search trên chỉ mục phim với truy vấntham số Bố già làm thuộc tính tiêu đề.

**Danh sách 8.5 Truy vấn tìm kiếm để lấy phim khớp với “Bố già”**

GET movies/\_search?q=title:Bố già

URL bao gồm điểm cuối \_search theo sau là truy vấn được biểu thị bằng chữ q. Truy vấn này trả về tất cả các phim có tiêu đề khớp với từ “Godfather” (chúng ta sẽ nhận được hai phim trong phản hồi: The Godfather Phần I và Phần II).

Để tìm kiếm phim khớp với nhiều từ, chúng ta có thể thêm tiêu đề làm từ khóa tìm kiếm với khoảng cách giữa chúng. Truy vấn sau đây tìm kiếm tất cả các phim khớp với các từ “Godfather”, “Knight” và “Shawshank”.

**Liệt kê 8.6 Tìm kiếm phim theo nhiều từ**

GET movies/\_search?q=title:Bố già hiệp sĩ Shawshank

Truy vấn này trả về bốn tiêu đề phim: The Shawshank Redemption, The Dark Knight, The Godfather Part I và The Godfather Part II. Lưu ý rằng theo mặc định, Elasticsearch sử dụngHOẶCtoán tử giữa các đầu vào truy vấn, do đó chúng ta không cần phải chỉ định nó (nó được coi là có mặt ngầm định).

Nếu chúng ta kiểm tra điểm liên quan, The Shawshank Redemption và The Dark Knight nhận được cùng một điểm (3.085904) và The Godfather Part I đạt điểm cao hơn một chút so với The Godfather Part II. Chúng ta có thể yêu cầu Elasticsearch giải thích cách nó có được điểm này bằng cách vượt quagiải thíchlá cờ:

GET movies/\_search?q=title:Bố già hiệp sĩ Shawshank&explain=true

Chúng tôi thảo luận về cờ giải thích trong phần 8.8.3.



**định dạng cURL**

cURL tương đương với mã trong danh sách 8.6 là

curl -XGET "http://localhost:9200/movies/\_search?q=title:Bố già hiệp sĩ Shawshank"

Bạn có thể tự động tạo lệnh cURL bằng cách nhấp vàobiểu tượng cờ lêbên cạnh nút Phát trong Giao diện người dùng Công cụ phát triển, như thể hiện trong hình.

* 1. ***Yêu cầu URItìm kiếm* 293**
     1. Tìm kiếm một bộ phim cụ thể

Để lấy một bộ phim cụ thể, chúng ta có thể kết hợp các tiêu chí với toán tử AND. Sau đây làTruy vấn này trả về một bộ phim duy nhất, The Dark Knight, khớp với trường tiêu đề và diễn viên.

**Liệt kê 8.7 Tìm kiếm phim theo tiêu đề và diễn viên**

GET movies/\_search?q=title:Knight VÀ diễn viên:Bale

Toán tử AND giúp thu hẹp kết quả. Nếu chúng ta bỏ qua toán tử AND và chạy truy vấn như trong danh sách tiếp theo, chúng ta sẽ nhận được thêm hai bộ phim (cả phim Bố già và The Dark Knight).

**Liệt kê 8.8 Lấy tiêu đề hoặcdiễn viên**

GET movies/\_search?q=title:Bố già diễn viên:Bale

Ở đây, chúng ta đang tìm kiếm những bộ phim khớp với "Godfather" hoặc những bộ phim khớp với diễn viên Christian Bale. (Hãy nhớ rằng Elasticsearch sử dụng toán tử OR theo mặc định.) Ngoài ra, để chỉ định toán tử AND trong tìm kiếm nhiều trường, chúng ta có thể chỉ cần thêm tham số default\_operator vào truy vấn.

**Liệt kê 8.9 Cài đặttoán tử mặc địnhBẰNGVÀ**

GET movies/\_search?q=title:Diễn viên Bố già:Bale&default\_operator=VÀ

Truy vấn này không trả về kết quả nào vì không có phim Bố già nào có Chris-tian Bale (ít nhất là cho đến nay). Hãy đảm bảo tham số default\_operator được thêm tiền tố là dấu thăng (&), như được hiển thị trong danh sách 8.9. Nhưng không nên có khoảng trắng sau & vì Kibana không thích khoảng trắng giữa & và thuộc tính default\_operator.

* + 1. Các thông số bổ sung

Chúng ta có thể truyền thêm tham số vào URL và thêm nhiều tiêu chí vào truy vấn.Ví dụ, chúng ta có thể đặt hai thuộc tính, from và size, để lấy kết quả phân trang (chúng ta sẽ xem xét phân trang sau trong chương này). Chúng ta cũng có thể sắp xếp kết quả bằng cách sử dụng, ví dụ, rating để sắp xếp theo xếp hạng phim. Một lần nữa, chúng ta có thể yêu cầu Elasticsearch giải thích cách tính điểm bằng cách sử dụng tham số Explain. Hãy viết một truy vấn với tất cả các yêu cầu này và một số yêu cầu khác.

**Liệt kê 8.10 Mở rộng truy vấn của chúng tôi với các tham số bổ sung**

GET movies/\_search?q=title:Diễn viên phim Bố già:

(Brando HOẶC Pacino) xếp hạng:(>=9.0 VÀ <=9.5)&from=0&size=10 &explain=true&sort=rating&default\_operator=AND

**294 CPHẦN8*Giới thiệu tìm kiếm***

Truy vấn này là sự kết hợp của các tham số, nhưng đây là cốt lõi. Chúng tôi đang tìm kiếm phim Godfa- ther (tiêu đề) với sự tham gia của Marlon BrandoHOẶCAl Pacino (diễn viên) với mộtxếp hạnggiữa 9VÀ9.5. Chúng tôi cũng thêm phân trang (từ,kích cỡ), sắp xếp theo xếp hạng và yêu cầu Elastic-searchgiải thích. Chúng tôi lấy lại hai bộ phim (cả hai đều là Bố già), mặc dù Brando khôngtrong Bố già Phần II (chúng tôi chỉ địnhBrando HOẶC Pacinotrongdiễn viêncánh đồng).

Như bạn có thể đoán, phương pháp yêu cầu URL để viết truy vấn là thô sơ và dễ xảy ra lỗi. Lý tưởng nhất là chúng ta nên viết truy vấn bằng Query DSL. May mắn thay, chúng ta có thể gói yêu cầu URL trong Query DSL để tận dụng tối đa cả hai thế giới.

* + 1. Hỗ trợYêu cầu URI với Query DSL

Query DSL có phương thức query\_string cho phép chúng ta gói lệnh gọi yêu cầu URI (chúng ta sẽ xem Query DSL ở phần tiếp theo). Chúng ta có thể gửi các tham số truy vấn URI để tìm kiếm phim có nhiều từ khóa tiêu đề trong nội dung yêu cầu dưới dạng query\_string.

**Liệt kê 8.11 Sử dụng phương thức URI được gói trong QueryDSL**

NHẬN phim/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"default\_field": "tiêu đề",

"query": "Knight Redemption Lord Pulp", "default\_operator": "HOẶC"

}

}

}

Chuỗi truy vấntương đương với tham số q mà chúng ta đã sử dụng trước đó trong phương pháp tìm kiếm URI. Mặc dù tốt hơn nhiều so với phương pháp yêu cầu URI, phương pháp query\_string có cú pháp nghiêm ngặt và có một số đặc điểm không dễ tha thứ. Trừ khi có lý do chính đáng để không sử dụng, hãy sử dụng truy vấn Query DSL thay vì query\_string. Chúng ta có thể sử dụng query\_string để kiểm tra nhanh, nhưng việc dựa vào nó cho các truy vấn phức tạp và chuyên sâu có thể gây ra rắc rối.

Đã đến lúc khám phá vương quốc hùng mạnh của Query DSL. Vì Query DSL là một con dao đa năng của quân đội Thụy Sĩ để tìm kiếm nên nó xứng đáng có một phần riêng.

#### Truy vấn DSL

Nhóm Elasticsearch đã phát triển ngôn ngữ và cú pháp đa năng, dành riêng cho tìm kiếm có tên là Query DSL. Đây là ngôn ngữ tinh vi, mạnh mẽ và biểu cảm có thể tạo ra vô số truy vấn từ cơ bản đến phức tạp, ngoài các truy vấn lồng nhau và phức tạp hơn. Nó cũng có thể được mở rộng cho các truy vấn phân tích. Đây là ngôn ngữ truy vấn dựa trên JSON có thể được xây dựng bằng các truy vấn để tìm kiếm và phân tích. Cú pháp và định dạng như sau:

NHẬN sách/\_tìm kiếm

{

**Gọi điểm cuối \_search trên chỉ mục sách**

* 1. ***Truy vấnDSL* 295**

**Kiểu truy vấn**

"truy vấn": {

"cuộc thi đấu": {

...

}

}

}

**Tất cả các truy vấn đều được gói trong đối tượng này.**

**Tiêu chí truy vấn được đính kèm ở đây.**

Chúng tôi gọi điểm cuối \_search bằng một đối tượng truy vấn, được truyền vào dưới dạng phần thân của yêu cầu. Đối tượng truy vấn bao gồm logic để tạo tiêu chí bắt buộc.

* + 1. Mẫu truy vấn

Chúng tôi đã làm việc với một số truy vấn được viết bằng định dạng Query DSL. Để hoàn thànhVì vậy, hãy viết truy vấn multi\_match để tìm kiếm từ khóa “Lord” trên hai trường: tóm tắt và tiêu đề.

**Liệt kê 8.12 Truy vấn mẫu DSL**

NHẬN phim/\_tìm kiếm

{

"truy vấn": { "multi\_match": {

"query": "Chúa ơi",

"các trường": ["tóm tắt","tiêu đề"]

}

}

}

GET movies/\_search là yêu cầu tìm kiếm viết tắt từ máy khách đến máy chủ Elastic-search. Yêu cầu đầy đủ giống như sau:

NHẬN http://localhost:9200/movies/\_search

Tất nhiên, máy chủ Elasticsearch của tôi đang chạy cục bộ, vì vậy localhost là địa chỉ của máy chủ. Yêu cầu này mong đợi một nội dung được định dạng JSON bao gồm truy vấn.

* + 1. Truy vấn DSL cho cURL

Truy vấn tương tự có thể được gọi thông qua cURL. Danh sách sau đây cho thấy lệnh gọi này.

**Liệt kê 8.13 Truy vấn DSL quacURL**

curl -XGET "http://localhost:9200/movies/\_search" -H 'Loại nội dung: ứng dụng/json' -d'

{

"truy vấn": { "multi\_match": {

"query": "Chúa ơi",

"các trường": ["tóm tắt","tiêu đề"]

}

}

}'

**296 CPHẦN8*Giới thiệu tìm kiếm***

Truy vấn được cung cấp như một đối số cho tham số -d. Lưu ý rằng toàn bộ truy vấn (bắt đầu bằng Content-Type) được đặt trong dấu ngoặc đơn khi gửi yêu cầu qua cURL.

* + 1. Truy vấn DSL để tổng hợp

Mặc dù chúng tôi chưa giới thiệu phần phân tích của Elasticsearch, nhưng đây là một bản tóm tắt nhanh. Với Query DSL, chúng tôi sử dụng một định dạng tương tự cho tổng hợp (phân tích), vớiaggs (viết tắt của aggregations) thay vì đối tượng truy vấn. Danh sách sau đây hiển thị định dạng này.

**Liệt kê 8.14 Truy vấn tổng hợp được viết theo định dạng DSL truy vấn**

NHẬN phim/\_tìm kiếm

{

"kích thước": 0,

"aggs": { "average\_movie\_rating": {

"trung bình": {

"field": "xếp hạng"

}

}

}

}

Truy vấn này lấy xếp hạng trung bình của tất cả các bộ phim bằng cách sử dụng tổng hợp số liệu được gọi làavg (viết tắt của average). Bây giờ chúng ta đã hiểu được dạng tổng thể của Query DSL, hãy cùng xem xét kỹ hơn về các truy vấn lá và truy vấn hợp chất mà chúng ta đã đề cập trước đó.

* + 1. Truy vấn lá và hợp chất

Query DSL hỗ trợ cả truy vấn lá và truy vấn hợp chất. Nội dung của truy vấn tìm kiếm có thể chấp nhận tiêu chí truy vấn đơn giản hoặc phức tạp.

Truy vấn lá là truy vấn đơn giản, không có mệnh đề. Các truy vấn như vậy sẽ lấy kết quả dựa trên các tiêu chí nhất định (ví dụ: lấy các bộ phim được đánh giá cao nhất, các bộ phim được phát hành trong một năm cụ thể, tổng doanh thu của một bộ phim, v.v.).

Với truy vấn lá, chúng ta có thể tìm thấy kết quả cho các tiêu chí đối với các trường nhất định. Danh sách sau đây cho thấy một ví dụ. (Đừng lo lắng về nội dung của truy vấn; chúng tôi sẽ thảo luận sâu hơn về các truy vấn như vậy trong một vài chương tiếp theo.)

**Liệt kê 8.15 Truy vấn lá phù hợp với cụm từ**

NHẬN phim/\_tìm kiếm

{

"truy vấn": { "cụm từ khớp": {

"tóm tắt": "Một người Hobbit hiền lành từ vùng Shire và tám người bạn đồng hành"

}

}

}

* 1. ***Tìm kiếmđặc trưng* 297**

Truy vấn lá không thể lấy nhiều mệnh đề truy vấn. Ví dụ, chúng không được thiết kế để tìm kiếm các bộ phim khớp với tiêu đề nhưng không được khớp với một diễn viên cụ thể và đã được phát hành trong một năm cụ thể và có xếp hạng không thấp hơn một số nhất định. Yêu cầu nâng cao về việc kết hợp hợp lý các mệnh đề nhất định để phục vụ cho một truy vấn phức tạp là không thể với truy vấn lá, dẫn đến việc giới thiệu các truy vấn hợp chất.

*Truy vấn hợp chất*cho phép chúng ta tạo các truy vấn phức tạp bằng cách kết hợp các truy vấn lá và thậm chí các truy vấn hợp chất khác bằng các toán tử logic. Ví dụ, truy vấn Boolean (bool) là một truy vấn hợp chất phổ biến hỗ trợ các mệnh đề như must, must\_not, should và filter. Chúng ta có thể viết các truy vấn phức tạp đáng kể bằng các truy vấn hợp chất, như ví dụ này.

**Liệt kê 8.16 Truy vấn hợp chất**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{"match": {"title": "Bố già"}}],

"must\_not": [{"range": {"rating": {"lt": 9.0}}}],

"nên": [{"match": {"actors": "Pacino"}}],

"bộ lọc": [{"trùng khớp": {"diễn viên": "Brando"}}]

}

}

}

Truy vấn hợp chất này kết hợp một số truy vấn lá được nối với nhau bằng các toán tử logic. Nó sẽ lấy tất cả các bộ phim phải khớp với tiêu đề Bố già, và không được có xếp hạng dưới 9. Truy vấn cũng nên xem xét các bộ phim có diễn viên Pacino. Cuối cùng, nó lọc ra mọi thứ ngoại trừ các bộ phim có diễn viên Brando. Quá nhiều! Nhưng đừng lo; chúng ta sẽ làm việc với truy vấn nâng cao bằng cách sử dụng truy vấn hợp chất trong các chương sắp tới.

Truy vấn lá (cũng như truy vấn nâng cao) được gói trong đối tượng truy vấn của yêu cầu tìm kiếm. Ngoài việc triển khai logic của truy vấn nâng cao (đôi khi có thể quá phức tạp), chúng ta sẽ không thấy sự khác biệt đáng kể nào khi viết truy vấn hợp chất.

Có một số tính năng cắt ngang, chẳng hạn như sắp xếp, phân trang và làm nổi bật-ing, mà bất kỳ loại truy vấn tìm kiếm nào cũng có thể sử dụng. Chúng không dành riêng cho các truy vấn cấp độ thuật ngữ hoặc khớp, truy vấn hợp chất hoặc truy vấn lá. Chúng tôi thảo luận chi tiết về các tính năng này trong phần sau. Chúng tôi cũng sử dụng các tính năng này thỉnh thoảng trong quá trình tìm kiếm và tổng hợp.

#### 8.8 Tính năng tìm kiếm

Elasticsearch cung cấp khả năng thêm các tính năng vào truy vấn và kết quả. Chúng ta có thể thao tác các tài liệu nguồn trong phản hồi bằng cách yêu cầu Elasticsearch trả về toàn bộ tài liệu hoặc chỉ các trường cụ thể. Chúng ta có thể sắp xếp các tài liệu dựa trên một hoặc nhiều trường ngoài việc sắp xếp theo điểm liên quan của tài liệu. Elasticsearch cho phép chúng ta phân trang

**298 CPHẦN8*Giới thiệu tìm kiếm***

results: ví dụ, chúng ta có thể chỉ định rằng mỗi trang bao gồm 100 tài liệu thay vì 10 tài liệu mặc định mà Elasticsearch trả về. Ngoài ra còn có một chức năng để làm nổi bật kết quả với các từ khớp tìm kiếm trong kết quả. Chúng ta thậm chí có thể yêu cầu công cụ chỉ lấy kết quả từ một tập hợp các phân đoạn cụ thể bằng cách sử dụng chức năng định tuyến phân đoạn.

Ngoài ra, hầu hết các truy vấn tìm kiếm đều hỗ trợ một số hàm cắt ngang, bất kể loại truy vấn nào (cấp thuật ngữ, toàn văn, truy vấn không gian địa lý, v.v.). Một số không liên quan đến một loại truy vấn cụ thể; ví dụ, không nên sắp xếp trên các trường văn bản, do đó chỉ giới hạn ở các truy vấn cấp thuật ngữ. Trong một vài phần tiếp theo, chúng ta sẽ khám phá các tính năng này để hiểu chi tiết về ứng dụng của chúng.

* + 1. Phân trang

Thông thường, các truy vấn sẽ cho ra nhiều kết quả, có thể là hàng trăm hoặc thậm chí hàng nghìn. Việc gửi tất cả các kết quả cho một truy vấn cùng lúc sẽ gây rắc rối vì cả phía máy chủ và phía máy khách đều cần đủ bộ nhớ và khả năng xử lý để xử lý lượng dữ liệu tải.

Theo mặc định, Elasticsearch gửi 10 kết quả hàng đầu, nhưng chúng ta có thể thay đổi số này bằng cách đặt tham số size trên truy vấn. Giá trị tối đa là 10.000, nhưng chúng ta cũng có thể thay đổi giới hạn này, như đã thảo luận sau đây. Truy vấn trong danh sách tiếp theo đặt size thành 20, trả về 20 kết quả hàng đầu cùng một lúc.

**Liệt kê 8.17 Truy vấn để lấy một số lượng kết quả cụ thể**

NHẬN phim/\_tìm kiếm

{

**"kích thước": 20,**

"truy vấn": {

"match\_all": {}

}

}

Thiết lập kích thước thành20 trả về 20 kết quả hàng đầu. Nếu chúng ta có một chỉ mục với 1 triệu tài liệu, việc đặt kích thước thành 10.000 sẽ truy xuất được nhiều tài liệu như vậy (bỏ qua các cân nhắc về hiệu suất trong giây lát!).

**Đặt lại giới hạn kích thước 10.000 tài liệu**

Số lượng kết quả tối đa chúng ta có thể lấy được bằng cách thiết lậpkích cỡthuộc tính là 10.000. Giả sử chúng ta đặt kích thước thành, chẳng hạn,10001và thực hiện truy vấn sau:

NHẬN phim/\_tìm kiếm

{

"kích thước": 10001,

"truy vấn": {

"match\_all": {}

}

}

***8.8 Tìm kiếmđặc trưng* 299**



Chúng tôi nhận được ngoại lệ sau: “Kết quảcửa sổ quá lớn, từ + kích thước phải là

nhỏ hơn hoặc bằng: [10000] nhưng là [10001].”

Mặc dù 10.000 là đủ cho hầu hết các tìm kiếm, nhưng nếu chúng ta cần lấy nhiều tài liệu hơn thế, chúng ta phải đặt lạicửa sổ kết quả tối đatrên chỉ mục.cửa sổ kết quả tối đalà một thiết lập động, vì vậy chúng ta có thể thay đổi nó bằng cách thực hiện truy vấn sau trên một chỉ mục trực tiếpvới những thay đổi cần thiết:

PUT phim/\_cài đặt

{

"cửa sổ kết quả tối đa":20000

}

**Đặt mức lợi nhuận tối đa**

**kết quả kích thước đến 20000**

Nói như vậy, nó là*không*nên sử dụng hình thức tìm kiếm này khi tìm kiếm dữ liệu lớn

bộ dữ liệu. Thay vào đó, tốt nhất là sử dụngtìm kiếm sautính năng mà chúng ta sẽ thảo luận sau trong phần này. Elasticsearch cung cấp mộtcuộnAPI để lấy các tập dữ liệu lớn, nhưng tôi khuyên bạn nên sử dụngtìm kiếm sautính năng trêncuộnGiao diện lập trình ứng dụng (API).

Ngoài tham số kích thước, giúp tạo ra kết quả hàng loạt, Elasticsearch còn cómột tham số khác được gọi là from để bù trừ kết quả. Là một offset, thiết lập from giúp bỏ qua một số lượng kết quả nhất định. Ví dụ, nếu from được đặt thành 200, 200 kết quả đầu tiên sẽ bị bỏ qua và kết quả sẽ được trả về bắt đầu từ 201. Danh sách sau đây cho thấy cách phân trang kết quả bằng cách đặt thuộc tính size và from.

**Liệt kê 8.18 Phân trang kết quả bằng cách sử dụngkích cỡVàtừ**

NHẬN phim/\_tìm kiếm

{

"kích thước": 100,

"từ": 3,

"truy vấn": {

"match\_all": {}

**Lấy mọi trang với 100 kết quả**

**Lấy từ trang thứ ba (trong tổng số 100 kết quả), bỏ qua hai trang đầu tiên**

}

}

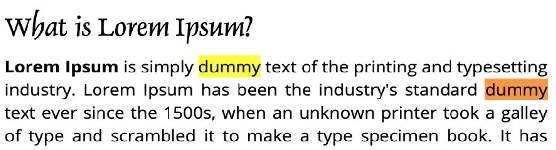
Trong danh sách, bằng cách đặt kích thước thành 100, chúng tôi lấy 100 tài liệu trong mỗi trang. Ngoài ra, chúng tôi lấy kết quả từ trang thứ tư (vì from được đặt thành 3).

Nếu tập kết quả quá lớn (hơn 10.000), thay vì làm việc với phân trangnation sử dụng các thuộc tính size và from, chúng ta cần sử dụng thuộc tính search\_after. Chúng ta thấy một ví dụ về điều này (phân trang sâu) trong chương 9. Bây giờ, chúng ta hãy xem một tính năng tìm kiếm phổ biến khác, tô sáng.

* + 1. Làm nổi bật

Khi chúng ta tìm kiếm một từ khóa trên một trang web trong trình duyệt internet của mình bằng cách sử dụng Ctrl-F, các kết quả được tô sáng để chúng nổi bật. Ví dụ, từ dummy được tô sáng trong hình 8.4. Việc tô sáng các từ khóa trong kết quả cho khách hàng của chúng tôi rất hấp dẫn và bắt mắt.

**300 CPHẦN8*Giới thiệu tìm kiếm***



**Hình 8.4 Một ví dụ về văn bản được tô sáng**

Trong Query DSL, chúng ta có thể thêm một đối tượng nổi bật ở cùng cấp độ với truy vấn cấp cao nhất

sự vật:

NHẬN sách/\_tìm kiếm

{

"truy vấn": { ... },

"điểm nổi bật": { ... }

}

Đối tượng nổi bật mong đợi một khối trường, trong đó chúng ta cung cấp các trường riêng lẻ để làm nổi bật trong kết quả:

NHẬN sách/\_tìm kiếm

{

"truy vấn": { ... }, "làm nổi bật": {

"các trường": { "trường1": {},

"trường2": {}

}

}

}

Khi kết quả được trả về từ máy chủ, chúng ta có thể yêu cầu Elasticsearch làm nổi bật các kết quả khớp với các thiết lập mặc định của nó bằng cách bao gồm văn bản khớp trong các thẻ nhấn mạnh (<em>khớp</em>). Mã trong danh sách tiếp theo tạo ra một đối tượng làm nổi bật, chỉ ra văn bản cần làm nổi bật trong trường tiêu đề của kết quả.

**Liệt kê 8.19 Làm nổi bật kết quả khi khớp**

NHẬN phim/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"thuật ngữ": {

"tiêu đề": {

**Ngăn chặn nguồn được trả về**

"giá trị": "cha đỡ đầu"

}

} **Bao gồm một đối tượng nổi bật dọc theo**

},"làm nổi bật": { "các trường": {

**với các trường được tô sáng**

***8.8 Tìm kiếmđặc trưng* 301**

"tiêu đề": {}

}

}

}

**Lĩnh vực chúng tôi muốn làm nổi bật**

Mã sau đây làm nổi bật từ Bố già bằng thẻ <em>. Nguồn bị ẩn trong kết quả vì chúng ta đặt \_source thành false trong truy vấn:

{

...

"highlight" : { "title" : ["Bố già"] }

},

{

...

"highlight" : { "title" : ["Bố già II"]}

}

Chúng tôi sử dụng thẻ <em> để nhấn mạnh phông chữ trong trình duyệt dựa trên HTML. Chúng tôi cũng có thể sử dụng thẻ tùy chỉnh. Ví dụ, mã này tạo một cặp dấu ngoặc nhọn ({{ và }}) làm thẻ:

...

"làm nổi bật": { "pre\_tags": "{{",

"post\_tags": "}}", "fields": {

"tiêu đề": {}

}

}

Kết quả là "The {{Godfather}}" (với dấu ngoặc nhọn làm nổi bật). Bây giờ chúng ta đã biết cách làm nổi bật kết quả tìm kiếm, hãy chuyển sự chú ý của chúng ta sang điểm liên quan trong dữ liệu.

* + 1. Giải thích điểm số liên quan

Elasticsearch cung cấp một cơ chế cho chúng ta biết chính xác cách công cụ tính toán tương đốiđiểm evancy. Điều này đạt được bằng cách sử dụng cờ giải thích trên điểm cuối \_search hoặc API giải thích. API giải thích cũng được sử dụng để xác định lý do tại sao một tài liệu có hoặc không khớp với truy vấn. Trong phần này, chúng ta sẽ xem xét cả hai phương pháp này để hiểu điểm chung và điểm khác biệt tinh tế của chúng.

**TANH TA GIẢI THÍCH CỜ**

Bạn có thể đã nhận thấy một số dương (giá trị điểm liên quan) trong một số kết quả truy vấn trước đó. Các giá trị đó đã được tính toán và thiết lập bởi công cụ, nhưng chúng tôi không giải thích cách thực hiện. Nếu chúng ta tò mò về phép tính, Elasticsearch cung cấp một cờđược gọi là giải thích mà chúng ta có thể đặt trong thân truy vấn. Khi chúng ta đặt thuộc tính giải thích thành true, Elasticsearch trả về kết quả với thông tin chi tiết về cách nó đạt được điểm số đó. Nói cách khác, nó giải thích logic và các phép tính được thực hiện bởi công cụ đằng sau hậu trường.

Danh sách sau đây hiển thị truy vấn khớp. Vì chúng tôi muốn biết cách tính điểm, chúng tôi đặt Explain thành True.

**302 CPHẦN8*Giới thiệu tìm kiếm***

**Liệt kê 8.20 Yêu cầu động cơ giải thíchđiểm**

NHẬN phim/\_tìm kiếm

{

**"giải thích": đúng,**"\_source": sai, "truy vấn": {

"cuộc thi đấu": {

"title": "Chúa"

}

}

}

Thuộc tính giải thích được đặt ở cùng cấp độ với đối tượng truy vấn. Kết quả của truy vấn này rất thú vị, như hình 8.5 cho thấy.



**Điểm liên quan cuối cùng = boost \* IDF \* TF**

**Tăng điểm**

**Tài liệu đảo ngược**

**Điểm số tần suất (IDF)**

**Tần suất thuật ngữ**

**(TF) điểm**

**Hình 8.5 Giải thích về cách Elasticsearch tính điểm liên quan**

Điểm liên quan được tính bằng cách nhân ba thành phần: tần suất tài liệu nghịch đảo (IDF), tần suất thuật ngữ (TF) và hệ số tăng cường. Elasticsearch đi sâu vào cách đánh giá và đo lường từng thành phần này. Đối vớiVí dụ, trường mô tả trong phản hồi trả về cho thấy IDF được tính là

log(1 + (N - n + 0.5) / (n + 0.5))

Ở đâu

* *N*là tổng số tài liệu có chứa thuật ngữ này (trong hình 8.5, có ba tài liệu có chứa từ lord).

***8.8 Tìm kiếmđặc trưng* 303**

* + *N*là tổng số tài liệu (hình 8.5 hiển thị 25 tài liệu trong chỉ mục của chúng tôi).

Tương tự như vậy, TF được tính bằng công thức này:

tần số / (tần số + k1 \* (1 - b + b \* dl / avgdl))

Giải thích về cách tính toán từng biến được cung cấp trong phần chi tiết của kết quả. Tôi khuyên bạn nên xem phần này để kiểm tra việc áp dụng các công thức của công cụ để tạo ra điểm số.

**GHI CHÚ**Nếu không có kết quả khớp thì sao? Ví dụ, nếu chúng ta thực hiện tìm kiếm bằng "Lords" thay vì "Lord" thì sao? Chúng ta sẽ nhanh chóng phát hiện ra rằng kết quả là trống. Hãy tự mình thử điều này trong mã để xem kết quả.

**TANH ẤY GIẢI THÍCH****Giao diện lập trình ứng dụng (API)**

Mặc dù chúng tôi sử dụng thuộc tính giải thích để hiểu cơ chế chấm điểm mức độ liên quan, nhưng cũng có một API \_explain cung cấp thông tin chi tiết về lý do tại sao một tài liệu khớp (hoặc không khớp), ngoài việc cung cấp các phép tính chấm điểm. Truy vấn trong danh sách sau sử dụng điểm cuối \_explain với ID tài liệu làm tham số để chứng minh phương pháp này.

**Liệt kê 8.21 Giải thích điểm số bằng cách sử dụng\_giải thíchđiểm cuối**

NHẬN phim/\_giải thích/14

{

"truy vấn":{

"cuộc thi đấu": {

"title": "Chúa"

}

}

}

Truy vấn này giống như truy vấn trong danh sách 8.20, nhưng lần này, để giải thích điểm số, chúng tôigọi điểm cuối \_explain thay vì đặt cờ Explain trên điểm cuối \_search.

Cuối cùng, hãy viết sai từ Lord thành Lords trong trận đấuthuộc tính trong danh sách 8.21 và chạy lại truy vấn. Như bạn có thể mong đợi, chúng ta không nhận được kết quả tương tự. Thay vào đó, chúng ta nhận được một manh mối:

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "14",

"đã khớp" : sai, "giải thích" : {

"giá trị" : 0.0,

**"description" : "không có thuật ngữ phù hợp",**

"chi tiết" : [ ]

}

}

**304 CPHẦN8*Giới thiệu tìm kiếm***

Như mô tả của đối tượng giải thích đã nói, Lords không khớp với dữ liệu được lập chỉ mục. Hiểu được lý do khớp (hoặc không khớp) giúp chúng ta khắc phục sự cố trạng thái của các truy vấn (ví dụ, trong ví dụ trước, chúng ta biết rằng thuật ngữ khớp không tồn tại trong chỉ mục của chúng ta). Nếu bạn thử lại truy vấn trong danh sách 8.21 bằng cờ giải thích, bạn có thể không nhận được bất kỳ thông tin nào từ công cụ ngoài một mảng trống.

Truy vấn tìm kiếm được xây dựng bằng cờ giải thích trên API \_search có thể tạo ra nhiều kết quả. Theo tôi, việc yêu cầu giải thích về tất cả các điểm tài liệu ở cấp độ truy vấn là lãng phí tài nguyên máy tính. Thay vào đó, hãy chọn một tài liệu và yêu cầu giải thích bằng API \_explain.

* + 1. Phân loại

Theo mặc định, kết quả trả về của công cụ được sắp xếp theo điểm liên quan (\_score): điểm càng cao thì thứ hạng trong danh sách kết quả càng cao. Tuy nhiên, Elasticsearch cho phép chúng ta quản lý thứ tự sắp xếp của điểm liên quan (tăng dần hoặc giảm dần) và chúng ta cũng có thể sắp xếp theo các trường khác, bao gồm nhiều trường.

**SKẾT QUẢ ORTING**

Để sắp xếp kết quả, chúng ta phải cung cấp một đối tượng sắp xếp ở cùng cấp độ với truy vấn. Đối tượng sắp xếp bao gồm một mảng các trường, trong đó mỗi trường chứa một vài tham số có thể điều chỉnh:

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"thể loại": "tội phạm"

}

},

"loại": [

{ "xếp hạng" :{ "thứ tự": "desc" } }

]

}

Tại đây, kết quả của truy vấn khớp tìm kiếm tất cả các phim trong thể loại tội phạm được sắp xếp theo xếp hạng phim. Đối tượng sắp xếp xác định trường (xếp hạng) và thứ tự mà kết quả dự kiến ​​được sắp xếp—trong trường hợp này là thứ tự giảm dần.

**SORTING TRÊN ĐIỂM SỐ LIÊN QUAN**

Tài liệu có liên quanđiểm được sắp xếp theo \_score theo thứ tự giảm dần theo mặc định nếu không có sắp xếp nào được chỉ định trong truy vấn. Ví dụ, truy vấn trong danh sách sau đây sắp xếp kết quả theo thứ tự giảm dần vì không đề cập đến thứ tự sắp xếp.

**Danh sách 8.22 Sắp xếp điểm số theo thứ tự giảm dần mặc định**

NHẬN phim/\_tìm kiếm

{

"kích thước": 10,

"truy vấn": {

"cuộc thi đấu": {

***8.8 Tìm kiếmđặc trưng* 305**

"title": "Bố già"

}

}

}

Điều này tương đương với việc đưa ra khối sắp xếp trong truy vấn, như được hiển thị trong danh sách tiếp theo.

**Liệt kê 8.23 ​​Sắp xếp trên\_điểm**

NHẬN phim/\_tìm kiếm

{

"kích thước": 10,

"truy vấn": {

"cuộc thi đấu": {

"title": "Bố già"

}

},

"loại": [

**Cho phép sắp xếp bằng cách đặt khối sắp xếp ở cùng cấp độ với khối truy vấn**

"\_điểm"

]

}

**Không có thứ tự sắp xếp nào được chỉ định, do đó theo mặc định, kết quả sẽ được sắp xếp theo thứ tự giảm dần.**

Nếu chúng ta muốn đảo ngược thứ tự với cách sắp xếp tăng dần để các tài liệu có điểm thấp hơn đượcở đầu danh sách, chúng ta chỉ cần thêm trường \_score để chỉ định thứ tự. Danh sách sau đây cho thấy cách thực hiện việc này.

**Liệt kê 8.24 Sắp xếp kết quả theo thứ tự tăng dần theođiểm**

NHẬN phim/\_tìm kiếm

{

"kích thước": 10,

"truy vấn": {

"cuộc thi đấu": {

"title": "Bố già"

}

},

"loại": [

{"\_score":{"order":"tăng dần"}}

]

}

Bạn có thể đã đoán được truy vấn để sắp xếp trên một trường tài liệu không tính điểm.danh sách tiếp theo sắp xếp dữ liệu dựa trên xếp hạng từ cao nhất đến thấp nhất.

**Liệt kê 8.25 Sắp xếp kết quả theocánh đồng**

NHẬN phim/\_tìm kiếm

{

"kích thước": 10,

"truy vấn": {

"cuộc thi đấu": {

"thể loại": "tội phạm"

**306 CPHẦN8*Giới thiệu tìm kiếm***

}

},

"loại": [

{"xếp hạng":{"thứ tự":"desc"}}

]

}

Khi chúng ta chạy truy vấn này, các kết quả (bỏ qua vì lý do ngắn gọn) được sắp xếp với bộ phim được đánh giá cao nhất ở đầu danh sách. Nếu chúng ta xem xét kỹ các kết quả, chúng ta thấy rằng điểm được đặt thành null—khi chúng ta sử dụng một trường để sắp xếp, Elasticsearch không tính điểm. Tuy nhiên, chúng ta có thể yêu cầu Elasticsearch tính điểm ngay cả khi chúng ta khôngsắp xếp theo \_score. Để làm điều đó, chúng ta có thể sử dụng trường Boolean track\_scores. Danh sách sau đây cho thấy cách thiết lập track\_scores để công cụ tính toán điểm.

**Liệt kê 8.26 Cho phép chấm điểm khi sắp xếp trên các trường**

NHẬN phim/\_tìm kiếm

{

**"track\_scores":đúng,**

"kích thước": 10,

"truy vấn": {

"cuộc thi đấu": {

"thể loại": "tội phạm"

}

},

"loại": [

{"xếp hạng":{"thứ tự":"tăng dần"}}

]

}

Thuộc tính track\_scores được tô sáng trong danh sách yêu cầu công cụ tính điểm liên quan của tài liệu. Tuy nhiên, các tài liệu không được sắp xếp theo thuộc tính \_score, vì trường tùy chỉnh được sử dụng để sắp xếp.

Chúng ta cũng có thể cho phép sắp xếp trên nhiều trường. Truy vấn tiếp theo cho phép sắp xếp trêncác trường rating và release\_date.

**Liệt kê 8.27 Sắp xếp theo nhiều trường theo thứ tự tăng dần**

NHẬN phim/\_tìm kiếm

{

"kích thước": 10,

"truy vấn": {

"cuộc thi đấu": {

"thể loại": "tội phạm"

}

},

"loại": [

{"xếp hạng":{"thứ tự":"tăng dần"}},

{"ngày phát hành":{"thứ tự":"tăng dần"}}

]

}

***8.8 Tìm kiếmđặc trưng* 307**

Khi chúng ta sắp xếp trên nhiều trường, thứ tự sắp xếp rất quan trọng! Kết quả của truy vấn này trước tiên được sắp xếp theo thứ tự tăng dần trên trường xếp hạng. Nếu nhiều phim có cùng xếp hạng, trường thứ hai (ngày phát hành) được sử dụng để phá vỡ sự cân bằng, do đó kết quả vớicùng một xếp hạng được sắp xếp theo ngày phát hành theo thứ tự tăng dần.

**GHI CHÚ**Chúng ta sẽ xem xét geosorting trong chương 12. Việc hiểu geoquery và sắp xếp theo geopoints cần có một chương riêng.

* + 1. Thao tác kết quả

Bạn có thể đã quan sát thấy rằng các truy vấn tìm kiếm trả về kết quả từ các tài liệu gốcđược chỉ định với trường \_source. Thỉnh thoảng, chúng ta có thể chỉ muốn lấy một tập hợp con các trường. Ví dụ, chúng ta có thể chỉ cần tiêu đề và xếp hạng của một bộ phim khi người dùng tìm kiếm một loại xếp hạng nhất định hoặc chúng ta có thể không cần tài liệu được đưa vào phản hồi của công cụ. Elasticsearch cho phép chúng ta thao tác phản hồi, cho dù là lấy các trường đã chọn hay xóa toàn bộ tài liệu.

**S****ĐẶT BỎ TÀI LIỆU**

Để ngăn chặn tài liệu được trả về trong phản hồi tìm kiếm, chúng tôi đặt cờ \_source thành

sai trong truy vấn.Danh sách sau đây trả về phản hồi chỉ có siêu dữ liệu.

**Liệt kê 8.28 Ẩn nguồntài liệu**

NHẬN phim/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"phù hợp": { "giấy chứng nhận": "R"

**Đặt cờ \_source thành false sẽ xóa tài liệu nguồn khỏi kết quả.**

}

}

}

Câu trả lời không đề cập đến tài liệu gốc:

"lượt truy cập" : [

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "1",

"\_điểm" : 0.58394784

},

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "2",

"\_điểm" : 0.58394784

},

...

]

**308 CPHẦN8*Giới thiệu tìm kiếm***

**FKHẮC CÁC TRƯỜNG ĐƯỢC CHỌN**

Chúng ta có thể chỉ lấy một vài trường đã chọn thay vì toàn bộ tài liệu. Elasticsearch cung cấp một đối tượng trường để chỉ ra trường nào sẽ được trả về. Chúng ta định nghĩa các trường một cách rõ ràng trong đối tượng này. Ví dụ, truy vấn sau đây chỉ lấy tiêu đềvà đánh giá các trường trong phản hồi:

NHẬN phim/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"phù hợp": { "giấy chứng nhận": "R"

}

},

"các trường": [

"tiêu đề", "xếp hạng"

]

}

Đoạn trích sau đây cho thấy phản hồi. Tài liệu được sắp xếp với tiêu đề

Vàxếp hạngcác trường, như mong đợi:

{

"\_index" : "phim",

"\_type" : "\_doc",

"\_id" : "1",

"\_score" : 0.58394784, "fields" : {

"xếp hạng" : [ 9.296875

],

"tiêu đề" : [

"Nhà tù Shawshank"

]

}

}

Lưu ý rằng mỗi trường được trả về dưới dạng một mảng thay vì một trường duy nhất. Vì có khả năng có nhiều giá trị, nên kết quả được biểu diễn dưới dạng một mảng json (Elastic-search không có kiểu mảng).

Chúng ta cũng có thể sử dụng ký tự đại diện trong bản đồ của trường. Ví dụ, thiết lậptiêu đề\*lấy lạitiêu đề,tiêu đề.bản gốc,tiêu đề\_mô tả dài,tiêu đề\_mãvà bất kỳ trường nào khác cótiêu đềtiền tố. (Chúng tôi không có tất cả các trường này trong bản đồ của mình,khác vớitiêu đềVàtiêu đề.bản gốc, do đó bạn có thể thêm chúng vào bản đồ để thử nghiệm với thiết lập ký tự đại diện.)

**SCÁC TRƯỜNG ĐÃ ĐƯỢC PHÁT HIỆN**

Đôi khi chúng ta cần tính toán một trường ngay lập tức và thêm nó vào phản hồi. Ví dụ, giả sử chúng ta muốn đặt một bộ phim được đánh giá cao nhất nếu nó nằm trong phạm vi

***8.8 Tìm kiếmđặc trưng* 309**

đánh giá cao nhất được trả về (đánh giá lớn hơn, chẳng hạn, 9). Chúng ta có thể sử dụng các tính năng viết kịch bản khi thêm các trường tùy ý theo yêu cầu.

Để sử dụng tính năng tạo tập lệnh, hãy thêm truy vấn với đối tượng script\_fields ở cùng cấp độ với tên bắt buộc của tệp động mới và logic để điền vào. Danh sách sau đây tạo một trường mới, top\_rated\_movie, bằng cách đặt cờ dựa trên xếp hạng mà phim nhận được.

**Liệt kê 8.29 Lọc nguồn bằngkịch bảncánh đồng**

NHẬN phim/\_tìm kiếm

{

"\_source": ["title\*","synopsis", "rating"], "query": {

"phù hợp": { "giấy chứng nhận": "R"

}

},

"script\_fields": { "phim được đánh giá cao nhất": {

"kịch bản": {

"lang": "không đau",

"nguồn": "nếu (doc['rating'].value > 9.0) 'đúng'; nếu không thì 'sai'"

}

}

}

}

Tập lệnh bao gồm phần tử nguồn trong đó logic điền vào trường mới (top\_rated\_movie) được xác định: chúng tôi đóng dấu phim là được xếp hạng cao nhất nếu xếp hạng của phim đó lớn hơn 9. Để hoàn thiện, chúng ta hãy xem đầu ra (đã chỉnh sửa cho ngắn gọn) với trường top\_rated\_movie mới:

"lượt truy cập" : [{

...

"\_source" : { "xếp hạng" : "9.3",

"tóm tắt": "Hai người đàn ông bị giam cầm gắn kết ...", "tiêu đề": "Nhà tù Shawshank"

},

"các trường" : {

"top\_rated\_movie" : ["đúng"]

}

}

...

**SLỌC NGUỒN**

Trước đó, chúng tôi đặt cờ \_source thành false để ngăn không cho tài liệu được trả về trong phản hồi. Mặc dù chúng tôi đã chỉ ra các kịch bản tất cả hoặc không có gì, có một vài trường hợp sử dụng mà chúng tôi có thể triển khai tùy chọn \_source để điều chỉnh phản hồi thêm nữa. Ví dụ: danh sách sau đây đặt \_source thành ["title\*", "synopsis", "rating"]

**310 CPHẦN8*Giới thiệu tìm kiếm***

để kết quả trả về các trường tóm tắt và xếp hạng cùng với tất cả các trường có

tiêu đềnhư một tiền tố.

**Liệt kê 8.30 Sử dụng\_nguồnthẻ để lấy các trường tùy chỉnh**

NHẬN phim/\_tìm kiếm

{

"\_source": ["title\*","synopsis", "rating"], "query": {

"phù hợp": { "giấy chứng nhận": "R"

}

}

}

Chúng ta có thể lấy\_nguồntùy chọn thậm chí còn xa hơn bằng cách thiết lập một danh sáchbao gồmVà

loại trừđể kiểm soát các trường trả về.

**Liệt kê 8.31 Lọc nguồn sử dụngbao gồmVàloại trừ**

NHẬN phim/\_tìm kiếm

{

"\_nguồn": {

"bao gồm": ["title\*","synopsis","genre"], "không bao gồm": ["title.original"]

},

"truy vấn": {

"phù hợp": { "giấy chứng nhận": "R"

}

}

}

Các\_nguồnđối tượng mong đợi hai mảng:

* bao gồm—Tất cả các trường cần được trả về trong kết quả
* loại trừ—Các trường phải được loại trừ khỏi các trường được trả về từ

bao gồmdanh sách

Trong danh sách 8.31, chúng tôi mong đợi tất cảtiêu đềcác cánh đồng (tiêu đềVàtiêu đề.bản gốc), cũng nhưTóm tắtVàthể loạicác trường, được trả về từ truy vấn. Tuy nhiên, chúng ta có thể ngăn chặntiêu đề.bản gốcbằng cách đưa nó vàoloại trừmảng. Chúng ta có thể chơi vớibao gồmVàloại trừmảng để có được khả năng kiểm soát tốt hơn đối với các trường được trả về và hỗ trợép. Ví dụ, nếu chúng ta thêm một"loại trừ": ["tóm tắt","diễn viên"]mảng đến

\_nguồnđối tượng, tất cả các trường ngoại trừTóm tắtVàdiễn viênđược trả lại.

* + 1. Tìm kiếm trênchỉ mục và luồng dữ liệu

Dữ liệu thường được phân bổ trên các chỉ mục và luồng dữ liệu. May mắn thay, Elasticsearch cho phép chúng ta tìm kiếm dữ liệu trên nhiều chỉ mục và luồng dữ liệu bằng cách thêm các

***8.8 Tìm kiếmđặc trưng* 311**

chỉ mục trong yêu cầu tìm kiếm. Ví dụ, việc bỏ qua tên chỉ mục trong yêu cầu tìm kiếm sẽ yêu cầu công cụ tìm kiếm trên tất cả các chỉ mục:

NHẬN \_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"diễn viên": "Pacino"

}

}

}

Chúng ta cũng có thể sử dụng GET \*/\_search hoặc GET \_all/\_search, tương đương với truy vấn trước đó. Tất cả các biểu mẫu này tìm kiếm trên tất cả các chỉ mục trong cụm.

Khi chúng ta tìm kiếm trên nhiều chỉ mục, chúng ta có thể muốn một tài liệu được tìm thấy trong một chỉ mục được ưu tiên hơn cùng một tài liệu được tìm thấy trong một chỉ mục khác. Nghĩa là, chúng ta có thể muốn tăng cường một số chỉ mục nhất định so với các chỉ mục khác khi thực hiện tìm kiếm trên nhiều chỉ mục. Đối với điều đó, chúng ta có thể đính kèm một đối tượng indices\_boost ở cùng cấp độ với đối tượng truy vấn. Chúng ta có thể nhập nhiều chỉ mục với xếp hạng tăng cường phù hợp được đặt trên đối tượng indices\_boost này.

Để chứng minh, chúng ta có thể tạo hai chỉ mục mới (chỉ số\_đầuVàindex\_mới) và lập chỉ mục cho bộ phim The Shawshank Redemption trong đó (mã có sẵn trong các tệp của cuốn sách). Bây giờ chúng ta có cùng một bộ phim trên ba chỉ mục, hãy tạo truy vấn với yêu cầu nâng cao điểm tài liệu thu được từphim\_topvậy thì The Shawshank Redemption là kết quả cao nhất.

**Liệt kê 8.32 Tăng điểm của mộttài liệu**

NHẬN phim\*/\_tìm kiếm

{

"chỉ số\_tăng cường": [

{ "phim": 0.1},

{ "phim\_mới": 0},

{ "phim\_top": 2.0}

**Giảm chỉ số\_boost xuống 0,1**

**Tăng lên**

**Giảm chỉ số\_boost xuống 0**

],

"truy vấn": {

"cuộc thi đấu": {

"title": "Sự cứu chuộc"

**chỉ số\_tăng lên 2.0**

}

}

}

Truy vấn sẽ nhân đôi điểm nếu tài liệu được tìm thấy trong movies\_top của Query DSL và giảm nó xuống 10% (0,1) giá trị ban đầu của nó đối với các tài liệu được lấy từ moviesindex. Cuối cùng, chúng tôi đặt indices\_boost thành 0 cho movies\_new documents, nghĩa là về cơ bản không có boost nào được áp dụng. Ví dụ, nếu điểm gốc của document là 0.2876821 trong movies\_top, thì điểm mới là 0.5753642 (2 \* 0.2876821). Điểm của các document khác được tính toán dựa trên thiết lập trong đối tượng indexes\_boost.

**312 CPHẦN8*Giới thiệu tìm kiếm***

Và thế là xong! Chương tiếp theo sẽ thảo luận về các truy vấn cấp thuật ngữ khi chúng ta đã hiểu rõ hơn về các tính năng tìm kiếm URI và Query DSL.

#### Bản tóm tắt

* Tìm kiếm có thể được phân loại thành loại tìm kiếm có cấu trúc và không có cấu trúc.
* Dữ liệu có cấu trúc hoạt động với các trường không phải văn bản như trường số và ngày tháng hoặc các trường không được phân tích trong thời gian lập chỉ mục và tạo ra kết quả nhị phân (có hoặc không).
* Dữ liệu phi cấu trúc xử lý các trường văn bản được mong đợi có điểm liên quan. Công cụ chấm điểm kết quả dựa trên mức độ phù hợp của các tài liệu kết quả với tiêu chí.
* Chúng tôi sử dụng tìm kiếm theo cấp thuật ngữ cho các truy vấn có cấu trúc và tìm kiếm toàn văn cho dữ liệu không có cấu trúc.
* Mỗi yêu cầu tìm kiếm được xử lý bởi một nút điều phối. Các nút điều phối có trách nhiệm yêu cầu các nút khác thực hiện truy vấn, trả về dữ liệu một phần, tổng hợp dữ liệu và phản hồi cho máy khách bằng kết quả cuối cùng.
* Elasticsearch phơi bày một\_tìm kiếmđiểm cuối cho các truy vấn và tổng hợp. Chúng ta có thể gọi\_tìm kiếmđiểm cuối bằng cách sử dụng yêu cầu URI có tham số hoặc xây dựng yêu cầu đầy đủ bằng cú pháp đặc biệt gọi là Query DSL.
* Query DSL là lựa chọn ưu tiên để tạo truy vấn tìm kiếm. Chúng ta có thể xây dựng rất nhiều truy vấn, bao gồm cả truy vấn nâng cao, bằng cách sử dụng Query DSL.
* Query DSL cho phép chúng ta tạo các truy vấn lá và truy vấn hợp chất. Truy vấn lá là các truy vấn tìm kiếm đơn giản với một tiêu chí duy nhất. Truy vấn hợp chất được sử dụng cho các truy vấn nâng cao được xây dựng bằng các mệnh đề có điều kiện.
* Các tính năng liên ngành có sẵn cho hầu hết các loại truy vấn: phân trang, làm nổi bật, giải thích về điểm số, thao tác kết quả, v.v.

*Tìm kiếm theo cấp độ thuật ngữ*

***Chương này bao gồm***

* Hiểu các truy vấn cấp thuật ngữ
* Truy vấn cấp độ thuật ngữ trong hành động

Tìm kiếm cấp thuật ngữ được thiết kế để hoạt động với dữ liệu có cấu trúc như số, ngày, địa chỉ IP, liệt kê, loại từ khóa, v.v. Chúng giúp chúng ta tìm câu trả lời nhưng không xem xét mức độ liên quan. Nghĩa là chúng tìm kiếm các kết quả khớp chính xác thay vì mức độ khớp của tài liệu với truy vấn. Một điểm khác biệt cơ bản giữa các truy vấn này và tìm kiếm toàn văn là các truy vấn cấp thuật ngữ không trải qua phân tích văn bản.

Chương này tập trung vào tìm kiếm cấp thuật ngữ chi tiết và xử lý các loại truy vấn khác nhau với các ví dụ. Chúng ta hãy bắt đầu với phần tổng quan và sau đó xem xét các truy vấn cụ thể.

**GHI CHÚ**Mã cho chương này có sẵn trên GitHub ([http://mng.bz/](http://mng.bz/Gyw8) [gà8](http://mng.bz/Gyw8)) và trên trang web của cuốn sách ([https://www.manning.com/books/elas](https://www.manning.com/books/elasticsearch-in-action-second-edition) [ticsearch-in-action-second-edition](https://www.manning.com/books/elasticsearch-in-action-second-edition)).

**313**

**314 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

#### Tổng quan về tìm kiếm cấp thuật ngữ

Tìm kiếm cấp thuật ngữ có cấu trúc: truy vấn trả về kết quả khớp chính xác. Chúng tìm kiếm dữ liệu có cấu trúc như ngày, số và phạm vi. Với loại tìm kiếm này, chúng tôi không quan tâm đến mức độ khớp của kết quả (mức độ tương ứng của tài liệu với truy vấn), chỉ cần truy vấn trả về dữ liệu nếu truy vấn khớp. Do đó, chúng tôi không mong đợi điểm liên quan liên quan đến kết quả tìm kiếm cấp thuật ngữ.

Tìm kiếm ở cấp độ thuật ngữ tạo ra tùy chọn nhị phân có hoặc không tương tự như cơ sở dữ liệuMệnh đề WHERE. Kết quả truy vấn sẽ được lấy nếu điều kiện được đáp ứng; nếu không, truy vấn sẽ không trả về bất kỳ kết quả nào.

Mặc dù các tài liệu có điểm số liên quan đến chúng, nhưng điểm số không quan trọng. Các tài liệu được trả về nếu chúng khớp với truy vấn, nhưng không liên quan. Trên thực tế, chúng ta có thể chạy các truy vấn cấp thuật ngữ với điểm số không đổi. Chúng có thể được lưu vào bộ nhớ đệm bởi máy chủ, do đó cung cấp lợi ích về hiệu suất nếu cùng một truy vấn được chạy lại. Các truy vấn này giống như tìm kiếm cơ sở dữ liệu truyền thống.

* + 1. Các truy vấn cấp độ thuật ngữ không được phân tích

Một đặc điểm quan trọng của truy vấn cấp thuật ngữ là chúng không được phân tích và mã hóa (không giống như truy vấn toàn văn). Ngoại lệ của quy tắc này là khi chúng ta sử dụng trình chuẩn hóa. Các thuật ngữ được khớp với các từ được lưu trữ trong chỉ mục đảo ngược mà không áp dụng trình phân tích để khớp với mẫu lập chỉ mục. Điều này có nghĩa là các từ tìm kiếm phải khớp với các trường được lập chỉ mục trong chỉ mục đảo ngược.

Ví dụ, nếu chúng ta tìm kiếm "Java" trong trường tiêu đề bằng truy vấn cấp thuật ngữ, khả năng là các tài liệu sẽ không khớp. Điều này là do trong quá trình lập chỉ mục, giả sử chúng tađang sử dụng một trình phân tích chuẩn, từ Java được chuyển thành chữ thường (java) và được chèn vào chỉ mục đảo ngược của tiêu đề. Vì các truy vấn cấp thuật ngữ không được phân tích, nên công cụ sẽ cố gắng khớp từ tìm kiếm “Java” với từ “java” trong chỉ mục đảo ngược, do đó việc khớp không thành công. Chúng ta có thể trả về cùng một kết quả từ truy vấn (với “Java” được viết hoa) nếu chúng ta sử dụng một loại từ khóa thay thế (chúng tôi sẽ giải thích ngắn gọn, vì vậy hãy kiên nhẫn).

Truy vấn cấp thuật ngữ phù hợp với tìm kiếm từ khóa, không phải tìm kiếm trường văn bản, vì chúng tôi biết bất kỳ trường nào được xác định là từ khóa đều được thêm vào chỉ mục đảo ngược mà không được phân tích trong quá trình lập chỉ mục. Giống như từ khóa, số, Boolean, phạm vi, v.v. không được phân tích và được thêm trực tiếp vào các chỉ mục đảo ngược tương ứng.

* + 1. Ví dụ truy vấn cấp thuật ngữ

Hãy lấy một ví dụ đơn giản từ bộ phim Bố già. Hình 9.1 minh họa lập chỉ mục và tìm kiếm cấp thuật ngữ. Trình phân tích chuẩn không tìm thấy kết quả vì “Bố già” không tồn tại dưới dạng một mã thông báo duy nhất được lưu trữ trong chỉ mục đảo ngược (nó được trình phân tích chia thành hai mã thông báo). Tương tự như vậy, chỉ sử dụng “Bố già” làm từ tìm kiếm trong truy vấn cấp thuật ngữ sẽ không trả về bất kỳ kết quả nào vì từ Bố già không khớp với từ Bố già viết thường.

* 1. ***Tổng quan về cấp độ học kỳtìm kiếm* 315**

{

"title”:"Bố già"

}

"cha đỡ đầu"

Tìm kiếm “**CácCha đỡ đầu**r”

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"thuật ngữ": {

"title": "Bố già"

}

}

}

tiêu chuẩn

máy phân tích

Không có máy phân tích

trong chỉ số đảo ngược

Cấp độ học kỳ (thuật ngữ) truy vấn

**Hình 9.1 Lập chỉ mục và tìm kiếm theo cấp độ thuật ngữ cho bộ phim Bố già**

Không có trận đấu nào

Như hình minh họa, có hai quy trình: lập chỉ mục tài liệu và tìm kiếm tài liệu. Nếu trường là trường văn bản, giả sử trình phân tích chuẩn được áp dụng, tiêu đề sẽ được chia thành hai mã thông báo và viết thường ["the" "godfather"] trong quá trình lập chỉ mục.

Mặt khác, trong quá trình tìm kiếm ở cấp độ thuật ngữ, các thuật ngữ tìm kiếm được truyền nguyên trạng, không có bất kỳ phân tích văn bản nào. Nếu truy vấn ở cấp độ thuật ngữ tìm kiếm “Bố già”, công cụ sẽ cố gắng tìm kiếm chuỗi chính xác “Bố già” trong chỉ mục đảo ngược.

Chúng ta vẫn có thể chạy các truy vấn cấp thuật ngữ trên các trường văn bản, mặc dù không nên thực hiện trên các trường có văn bản dài. Chúng ta cũng có thể sử dụng các truy vấn cấp thuật ngữ nếu văn bản có các phép liệt kê như các ngày trong tuần, chứng chỉ phim, giới tính, v.v. Nếu chúng ta lập chỉ mục giới tính, chẳng hạn như nam và nữ, các truy vấn cấp thuật ngữ phải sử dụng "nam" và "nữ" đểtrả về thành công bất kỳ kết quả nào vì hoạt động của trình phân tích chuẩn trong quá trình lập chỉ mục. Điểm mấu chốt là các truy vấn cấp thuật ngữ tìm kiếm các từ chính xác.

Elasticsearch hiển thị một số truy vấn cấp thuật ngữ, bao gồm thuật ngữ, thuật ngữ, id, mờ, tồn tại, phạm vi và các truy vấn khác. Chúng tôi sẽ xem xét một số truy vấn quan trọng trong phần tiếp theo và sau đó trình bày các ví dụ thực hành.

**GHI CHÚ**Chúng tôi đã lập chỉ mục dữ liệu phim mẫu trong chương 8 và chúng tôi xây dựng tất cả các truy vấn của mình dựa trên dữ liệu đó. Để hoàn thiện, mã nguồn cho chương này cung cấp các bước để lập chỉ mục ánh xạ phim và dữ liệu mẫu.

**316 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

#### Thuật ngữ truy vấn

Truy vấn thuật ngữ sẽ tìm nạp các tài liệu khớp chính xác với một trường nhất định. Trường không được phân tích. Thay vào đó, nó được khớp với giá trị được lưu trữ trong chỉ mục đảo ngược trong quá trình lập chỉ mục. Ví dụ, sử dụng tập dữ liệu phim của chúng tôi, chúng tôi có thể phát triển truy vấn thuật ngữ để tìm kiếm phim được xếp hạng R.

**Liệt kê 9.1 Lấy phim với mộtxếp hạng**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"thuật ngữ": {

**Thuật ngữ truy vấn**

**tuyên ngôn**

"giấy chứng nhận": "R"

}

}

}

Tên của truy vấn (thuật ngữ trong trường hợp này) xác định rằng chúng ta sắp thực hiện tìm kiếm ở cấp thuật ngữ. Đối tượng mong đợi trường (chứng chỉ, trong trường hợp này) và giá trị tìm kiếm. Hãy nhớ rằng chứng chỉ là kiểu dữ liệu từ khóa, vì vậy trong quá trình lập chỉ mục, giá trị “R” không được bất kỳ trình phân tích nào xử lý (thực tế, đó làcông cụ phân tích từ khóa không thay đổi trường hợp); do đó nó được lưu trữ nguyên trạng.

Nếu bạn chạy truy vấn này, bạn sẽ nhận được tất cả các phim được xếp hạng R (14 trong tập dữ liệu mẫu của chúng tôi được xếp hạng R). Chúng được gói gọn trong phản hồi JSON trả về. Trong phần tiếp theo, chúng tôiquan sát tác động của việc chạy tìm kiếm theo cấp thuật ngữ trên các trường văn bản (thay vì loại từ khóa).

* + 1. Thuật ngữ truy vấn trên các trường văn bản

Hãy xem điều gì xảy ra nếu chúng ta thay đổi truy vấn có giá trị xếp hạng thành “r” từ “R” bằng cách viết thường tiêu chí tìm kiếm của chúng ta (chẳng hạn như "certificate": "r"). Thật ngạc nhiên, truy vấn này không nhận được bất kỳ kết quả nào. Bạn có thể đoán được lý do không?

Nhớ lại từ chương 7 rằng Elasticsearch phân tích các trường văn bản trong quá trình lập chỉ mục và khi tìm kiếm. Trường chứng chỉ là một loại từ khóa, vì vậy trường không bao giờ trải qua quá trình phân tích. Điều này có nghĩa là nó sẽ luôn được khớp với nội dung của chỉ mục đảo ngược. Khi lập chỉ mục tài liệu, giá trị chứng chỉ "R" không bao giờ được mã hóa hoặc chuyển qua bộ lọc; nó được chèn vào chỉ mục đảo ngược nguyên trạng.

Mặt khác của đồng xu là tìm kiếm: các truy vấn thuật ngữ cũng không được phân tích. Không giống nhưmột bộ mã hóa chuẩn mã hóa trường truy vấn thành nhiều mã thông báo và viết thường chúng, trường truy vấn vẫn giữ nguyên. Nếu chúng ta tìm kiếm "R", nó được coi là chữ hoa vì không có chuyển đổi chữ thường nào (thông qua bộ mã hóa chuẩn) được áp dụng ở chế độ nền. Do đó, khi chúng ta tìm kiếm chứng chỉ chữ thường dưới dạng ("r", ví dụ), không có kết quả khớp nào ("R" đã được lập chỉ mục, không phải "r"), do đó không có kết quả nào.

Điều này đưa ra một điểm quan trọng về việc làm việc với các truy vấn thuật ngữ: các truy vấn thuật ngữ không phù hợp khi làm việc với các trường văn bản. Mặc dù không có gì ngăn cản chúng ta sử dụng

* 1. ***Thuật ngữtruy vấn* 317**

chúng, chúng được dùng cho các trường không phải văn bản như từ khóa, số và ngày tháng.

vẫn còn truy vấn thuật ngữtạo ra điểm số khi được gọi trong ngữ cảnh truy vấn.

Nếu bạn muốn sử dụng truy vấn thuật ngữ trên trường văn bản, hãy đảm bảo trường văn bản được lập chỉ mục như một phép liệt kê hoặc hằng số. Ví dụ: trường trạng thái đơn hàng có trạng thái CREATED, CANCELLED và FULFILLED là ứng cử viên tốt để sử dụng truy vấn thuật ngữ ngay cả khi đó là trường văn bản.

Tuy nhiên, nếu các trường văn bản được điền bằng văn bản không có cấu trúc như không phải dạng sốgiá trị theo kiểu tion, chúng ta sẽ không nhận được kết quả mong đợi khi chạy truy vấn thuật ngữ. Hãy cùng xem một ví dụ về những gì xảy ra khi chúng ta chạy truy vấn thuật ngữ trên trường văn bản.

* + 1. Ví dụ thuật ngữ truy vấn

Hãy xem điều gì xảy ra nếu chúng ta tìm kiếm một trường văn bản có tên là tiêu đề bằng cách sử dụng truy vấn thuật ngữ. Danh sách sau đây tìm kiếm "Bố già" trong tiêu đề của bộ phim bằng cách sử dụng truy vấn thuật ngữ. (Hãy nhớ rằng trong các ánh xạ phim của chúng ta, trường tiêu đề có một tập hợp ánh xạ trường văn bản rõ ràng.)

**Liệt kê 9.2 Sử dụng truy vấn cấp thuật ngữ trênchữcánh đồng**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"thuật ngữ": {

"title": "Bố già"

}

}

}

Khi chúng tôi chạy mã này, chúng tôi không nhận được kết quả nào (xem hình 9.1). Điều này là dotrường tiêu đề là một trường văn bản, do đó, nó đã trải qua một quá trình phân tích và được lưu trữ trong chỉ mục trước khi tìm kiếm. Bố già đã được chia nhỏ và lưu trữ dưới dạng các mã thông báo viết thường (vì chúng tôi đang sử dụng trình phân tích chuẩn theo mặc định) trong chỉ mục đảo ngược: ["the", "godfather"]. Truy vấn tìm kiếm không được phân tích cho các truy vấn thuật ngữ; thay vào đó, nó lấy từng từ như hiện trạng và so sánh với chỉ mục đảo ngược. Trong trường hợp này, tiêu chí truy vấn "Bố già" không khớp với các mã thông báo ("the" và "godfather") cho trường tiêu đề.

Chạy lại truy vấn bằng cách sử dụng "the godfather" cũng không trả về bất kỳ kết quả nào (hãy thử chạy truy vấn, viết thường tiêu đề như thế này). Thuật ngữ truy vấn cố gắng khớp với giá trị chính xác "the godfather", không có trong chỉ mục đảo ngược (hãy nhớ rằng, tiêu đề đã được mã hóa và lưu trữ dưới dạng hai từ: "the" và "godfather"). Tuy nhiên, tìm kiếm "godfather" trả về kết quả vì từ godfather đã được phân tích và chèn vào chỉ mục đảo ngược trong quá trình lập chỉ mục dữ liệu và do đó tìm thấy kết quả khớp.

Điểm mấu chốt là chúng ta cần chạy truy vấn thuật ngữ trên các trường không phải văn bản. Nếu bạn muốn sử dụng truy vấn thuật ngữ để tìm kiếm trường văn bản, hãy đảm bảo trường văn bản chứa dữ liệu dưới dạng liệt kê hoặc hằng số.

**318 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

* + 1. Các truy vấn cấp độ thuật ngữ được rút gọn

Thuật ngữ truy vấn trong danh sách 9.1 và 9.2 là phiên bản rút gọn. Mặc dù viết phiên bản rút gọn rất tiện lợi, chúng ta hãy dành chút thời gian để xem phiên bản đầy đủ gốc của truy vấn từ danh sách 9.1.

**Liệt kê 9.3 Cú pháp đầy đủ của một truy vấn**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"term": { "certificate": {

"giá trị": "R", "tăng cường": 2

}

}

}

}

**Trường chứng chỉ**

**có giá trị kèm theo**

**trong mộtsự vật. Giấy chứng nhận củatiêu chí tìm kiếm**

**Ngoài giá trị, chúng ta có thể cung cấp các tham số như boost.**

Trường chứng chỉ mong đợi một đối tượng có giá trị và các tham số khác. giá trị, xuất hiện ở cùng cấp độ với trường trong phiên bản rút gọn, thấp hơn một cấp so với trường. Đối tượng được bao quanh cũng có thể lưu trữ các thuộc tính khác, chẳng hạn như boost, như trong ví dụ này.

Trong khi phiên bản đầy đủ bổ sung các tính năng cho truy vấn, phiên bản rút gọn thì đơn giản và được sử dụng rộng rãi khi truy vấn đơn giản và không cần điều chỉnh thêm. Chúng tôi sử dụng các phiên bản rút gọn trong toàn bộ cuốn sách trừ khi chúng tôi quan tâm đến việc làm việc với các tham số khác.

Cho đến nay, chúng ta đã tìm kiếm các từ trong một trường duy nhất bằng cách sử dụng truy vấn thuật ngữ. Truy vấn thuật ngữ sẽ tìm kiếm các kết quả khớp chính xác cho một từ duy nhất, chẳng hạn như "certificate":"R". Nhưng nếu chúng ta muốn tìm kiếm nhiều giá trị trong một trường duy nhất thì sao? Ví dụ, làm thế nào chúng ta có thể tìm kiếm phim được xếp hạng R và PG-13 trên trường chứng chỉ? Đó là nơi truy vấn thuật ngữ xuất hiện.

#### Các điều khoản truy vấn

Như tên gọi của nó, truy vấn thuật ngữ (lưu ý số nhiều) tìm kiếm nhiều tiêu chí dựa trên một trường duy nhất. Chúng ta có thể đưa vào tất cả các giá trị có thể có của trường mà chúng ta muốn tìm kiếm. Giả sử chúng ta muốn tìm kiếm tất cả các phim có nhiều xếp hạng nội dung, chẳng hạn nhưnhư PG-13 và R. Chúng tôi sử dụng truy vấn thuật ngữ cho mục đích này.

**Liệt kê 9.4 Tìm kiếm nhiều tiêu chí tìm kiếm trong một trường**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"điều khoản": {

**Các điều khoản truy vấn mong đợi một**

**mảng tìm kiếmtiêu chuẩn. Nhiều tiêu chí tìm kiếm**

"giấy chứng nhận": ["PG-13","R"]

}

}

}

**chống lại một lĩnh vực duy nhất**

* 1. ***Các điều khoảntruy vấn* 319**

Truy vấn thuật ngữ mong đợi một danh sách các từ tìm kiếm được truy vấn theo một trường, được truyền vào dưới dạng một mảng cho đối tượng thuật ngữ. Các giá trị mảng được tìm kiếm theo các tài liệu hiện có từng cái một để lấy các kết quả khớp. Mỗi từ được khớp chính xác. Trong danh sách này, chúng tôi tìm kiếm tất cả các phim có xếp hạng PG-13 hoặc R trong trường chứng chỉ. Các tài liệu kết quả kết hợp tất cả các phim PG-13 và R.

Có một giới hạn về số lượng các thuật ngữ mà chúng ta có thể đặt trong mảng: một con số khổng lồ là 65.536 thuật ngữ. Nếu chúng ta cần sửa đổi giới hạn này (để tăng hoặc giảm nó), chúng ta có thể sử dụngthiết lập thuộc tính index.max\_terms\_count động của chỉ mục để thay đổi giới hạn. Truy vấn sau đây đặt max\_terms\_count thành 10.

**Liệt kê 9.5 Đặt lại số lượng tối đa các thuật ngữ**

PUT phim/\_cài đặt

{

"chỉ mục":{ "số lượng\_điều\_khoản\_tối\_đa":10

}

}

Thiết lập này hạn chế thiết lập của người dùng không quá 10 giá trị trong mảng terms. Hãy nhớ rằng, đây là thiết lập động trên chỉ mục, vì vậy chúng ta có thể thay đổi nó trên chỉ mục trực tiếp.

Có một phiên bản hơi khác của truy vấn thuật ngữ: truy vấn tra cứu thuật ngữ.ý tưởng là tạo mảng thuật ngữ từ các giá trị của tài liệu hiện có thay vì thiết lập cụ thể. Phần tiếp theo sẽ thảo luận về nó bằng một ví dụ.

* + 1. Ví dụ thuật ngữ truy vấn

Cho đến nay, chúng tôi đã cung cấp một danh sách các giá trị trong một mảng làm tiêu chí tìm kiếm cho một thuật ngữtruy vấn. Truy vấn tra cứu thuật ngữ là một biến thể của truy vấn thuật ngữ cho phép chúng ta đặt các thuật ngữ bằng cách đọc các giá trị trường của tài liệu hiện có. Tốt nhất là hiểu điều này thông qua một ví dụ.

Để giải thích tính năng này, chúng ta cần phải đi chệch khỏi tập dữ liệu phim của mình, tạo một chỉ mục mới với lược đồ phù hợp và lập chỉ mục một số tài liệu. Danh sách sau đây tạo ra một chỉ mục classic\_movies với hai thuộc tính: tiêu đề và đạo diễn.

**Liệt kê 9.6 Tạo mộtchỉ số**

ĐẶT phim kinh điển

{

"ánh xạ": { "thuộc tính": {

"tiêu đề": {

"loại": "văn bản"

},

"giám đốc": {

"loại": "từ khóa"

}

}

}

}

**Trường tiêu đề là trường văn bản.**

**Khai báo trường giám đốc là loại từ khóa**

**320 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

Không có gì đặc biệt về chỉ mục này—ngoại trừ chúng tôi định nghĩa trường giám đốc là

loại từ khóa, để tránh sự phức tạp. Bây giờ, chúng ta hãy lập chỉ mục một vài bộ phim.

**Liệt kê 9.7 Lập chỉ mục ba bộ phim**

ĐẶT classic\_movies/\_doc/1

{

"title":"Hàm cá mập", "director":"Steven Spielberg"

}

ĐẶT classic\_movies/\_doc/2

{

"title":"Hàm cá mập II", "director":"Jeannot Szwarc"

}

ĐẶT classic\_movies/\_doc/3

{

"title":"Ready Player One", "director":"Steven Spielberg"

}

Các tài liệu này đều có giải thích rõ ràng.

* + 1. Các thuật ngữ tra cứu truy vấn

Bây giờ chúng ta đã lập chỉ mục cho ba tài liệu này, hãy quay lại thảo luận về truy vấn tra cứu thuật ngữ. Giả sử chúng ta muốn lấy tất cả các bộ phim do Spielberg đạo diễn. Tuy nhiên, chúng ta không muốn xây dựng truy vấn thuật ngữ và cung cấp các thuật ngữ trực tiếp ngay từ đầu. Thay vào đó, chúng ta cho truy vấn thuật ngữ biết để lấy các giá trị của các thuật ngữ từ một tài liệu. Danh sách tiếp theo thực hiện chính xác điều này.

**Liệt kê 9.8 Ađiều khoảntra cứu tìm kiếm**

NHẬN classic\_movies/\_search

{

"truy vấn": {

"điều khoản": {

"giám đốc": {

**Một truy vấn thuật ngữ**

**(với một chút thay đổi!)**

**Lĩnh vực mà chúng tôi quan tâm tìm kiếm**

**Trường chỉ mục biểu thị tên của chỉ mục nơi**

"index":"phim\_cổ\_điển","id":"3",



"đường dẫn":"giám đốc"

}

**tài liệu lưu trữ.**

**Tên của trường chứa các thuật ngữ cho truy vấn**

} **Trường tìm kiếm trong**

} **hiện hành tài liệu**

}

Mã này cần một chút giải thích. Chúng ta tạo một truy vấn thuật ngữ với director là trường mà chúng ta muốn tìm kiếm nhiều thuật ngữ tìm kiếm. Trong truy vấn thuật ngữ thông thường, chúng ta sẽ cung cấp một mảng với danh sách tên. Nhưng ở đây, chúng ta yêu cầu truy vấn tra cứu các giá trị director trong một tài liệu khác: tài liệu có giá trị id là 3.

* 1. ***Các idtruy vấn* 321**

Tài liệu có ID này phải được lấy từ chỉ mục classic\_movies, vì trường chỉ mục đề cập đến ID đó trong truy vấn. Và tất nhiên, trường từđể lấy các giá trị là director, được khai báo là trường path. Chạy truy vấn này sẽ lấy hai bộ phim do Spielberg đạo diễn.

Truy vấn tra cứu thuật ngữ giúp xây dựng truy vấn dựa trên các giá trị thu được từ một tài liệu khác thay vì một tập hợp các giá trị được truyền vào truy vấn. Nó cung cấp tính linh hoạt cao hơn khi xây dựng các thuật ngữ truy vấn: chúng ta có thể dễ dàng hoán đổi chỉ mục với bất kỳ chỉ mục nào khác để lấy tài liệu. Ví dụ, giả sử chúng ta có một chỉ mục có tên là movie\_search\_terms\_index chứa một số tài liệu về thuật ngữ tìm kiếm (tài liệu 1 chứa các thuật ngữ đạo diễn, tài liệu 2 chứa các thuật ngữ diễn viên, v.v.). Chúng ta có thể tham chiếu tài liệu này với các thuật ngữ đạo diễn từ movie\_search\_terms\_index trong truy vấn chính của mình và lấy kết quả. Theo cách này, truy vấn chính có thể là hằng số trong khi các tài liệu tra cứu được thay đổi theo yêu cầu. Bây giờ chúng ta đã hiểu các truy vấn thuật ngữ, hãy chuyển sang loại truy vấn trong đó chúng ta lấy các tài liệu được cung cấp một tập hợp ID.

#### Truy vấn id

Đôi khi chúng ta muốn lấy các tài liệu có những ID đó từ Elasticsearch.Như tên gọi của nó, truy vấn ids sẽ lấy các tài liệu khớp với một tập hợp ID tài liệu. Đây là cách đơn giản hơn nhiều để lấy nhiều tài liệu cùng lúc. Danh sách sau đây cho biết cách lấy tài liệu bằng danh sách ID tài liệu.

**Liệt kê 9.9 Lấy nhiều tài liệu bằng cách sử dụngidtruy vấn**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"ID": {

**Tên của**

**truy vấn**

"giá trị": [10,4,6,8]

}

}

}

**Cung cấp ID tài liệu dưới dạng một mảng**

Truy vấn này trả về bốn tài liệu có bốn ID tương ứng. Mỗi tài liệu được lập chỉ mục có một trường \_id bắt buộc.

**GHI CHÚ**Các trường siêu dữ liệu không được phép là một phần của định nghĩa lược đồ.Trường \_id, cùng với các trường khác như \_source, \_size, \_routing, v.v., là các trường siêu dữ liệu và do đó không thể là một phần của bài tập ánh xạ chỉ mục.

Chúng ta cũng có thể sử dụng truy vấn thuật ngữ để tìm các tài liệu có một tập hợp ID tài liệu, thay vì sử dụng truy vấn id, như chúng ta đã thấy trong danh sách 9.4. Danh sách tiếp theo hiển thị một ví dụ khác.

**Liệt kê 9.10 Ađiều khoảntruy vấn sử dụng một tập hợp ID để lấy tài liệu**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"điều khoản": {

**322 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

"\_id":[10,4,6,8]

}

}

}

Ở đây, chúng ta sử dụng truy vấn thuật ngữ và đặt mảng ID tài liệu trong trường \_id làm tiêu chí tìm kiếm của chúng ta. Bây giờ, hãy xem một loại truy vấn cấp thuật ngữ khác: truy vấn exists.

#### Truy vấn tồn tại

Đôi khi, các tài liệu dự án có hàng trăm trường. Việc lấy tất cả các trường trong phản hồi là lãng phí băng thông và việc biết liệu một trường có tồn tại trước khi cố gắng lấy nó là một bước kiểm tra trước tốt. Truy vấn tồn tại sẽ lấy các tài liệu cho một trường nhất định nếu trường đó tồn tại. Ví dụ, nếu chúng ta chạy truy vấn sau, chúng ta sẽ nhận được phản hồi liên quangiữ lại tài liệu vì tài liệu có trường tiêu đề đã tồn tại.

**Liệt kê 9.11 Chạy mộttồn tạitruy vấn để kiểm tra xem một trường có tồn tại hay không**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"tồn tại": {

"trường": "tiêu đề"

}

**Xác định truy vấn**

**gõ như tồn tại**

**Cung cấp trường mà chúng ta muốn kiểm tra trong tài liệu**

}

}

Nếu trường không tồn tại, truy vấn sẽ trả về một mảng hits trống (hits[]). Nếu bạn tò mò, hãy thử truy vấn này với một trường không tồn tại như title2 và bạn sẽ thấy mảng trống.Có một trường hợp sử dụng tinh tế khác của truy vấn exists: để truy xuất tất cả các tài liệu không có trường cụ thể (một trường không tồn tại). Ví dụ, danh sách tiếp theo sẽ kiểm tra tất cả các tài liệu không được phân loại là bí mật (giả sử các tài liệu được phân loại

có một trường được gọi làbảo mậtđặt thànhĐÚNG VẬY).

**Liệt kê 9.12 Tìm kiếm tài liệu không bí mật**

ĐẶT top\_secret\_files/\_doc/1



{

"code":"Chim bay", "bí mật":đúng

}

**Thêm hai tài liệu, một tài liệu có cờ bảo mật**

ĐẶT top\_secret\_files/\_doc/2

{

"mã":"Đá Lạnh"

}

NHẬN top\_secret\_files/\_search

{

"truy vấn": {

"bool": {

"must\_not": [{

**Truy vấn hợp chất lấy các tài liệu không có trường bí mật**

* 1. ***Truy vấn phạm vi***

"tồn tại": {

"trường": "bí mật"

**323**

}

}

]

}

}

}

Chúng tôi thêm hai tài liệu vào chỉ mục top\_secret\_files: một trong các tài liệu có một trường bổ sung được gọi là bí mật. Sau đó, chúng tôi viết một truy vấn exists trong mệnh đề must\_not của truy vấn bool để lấy tất cả các tài liệu không được phân loại là bí mật. (Chúng tôi thảo luận về các truy vấn hợp chất trong chương 11.)

Đôi khi chúng ta muốn làm việc với dữ liệu nằm trong một phạm vi được xác định trước: lấy phim từ tháng trước, doanh số trong một quý, doanh thu cao nhất, v.v. Các truy vấn này được nhóm dưới tiêu đề truy vấn phạm vi, mà chúng ta sẽ xem xét tiếp theo.

#### Truy vấn phạm vi

Thông thường, chúng ta cần một tập hợp dữ liệu nằm trong một phạm vi: các chuyến bay bị hoãn giữa các ngày nhất định, lợi nhuận bán hàng trong một ngày cụ thể, học sinh có chiều cao trung bình trong một lớp, v.v.bật. Elasticsearch cung cấp truy vấn phạm vi cho các loại truy vấn này.

Truy vấn phạm vi trả về các tài liệu cho một phạm vi trong một trường. Truy vấn chấp nhận các giới hạn trên và dưới của trường. Ví dụ, để lấy tất cả các phim có xếp hạng trong khoảng

9.0 và 9.5, chúng ta có thể thực hiện truy vấn phạm vi sau.

**Liệt kê 9.13 Tìm kiếm phim có phạm vi xếp hạng được chỉ định**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"phạm vi": {

"xếp hạng": {

"gte": 9.0,

"lte": 9.5

}

}

}

}

Truy vấn phạm vi này lấy các bộ phim tronggiới hạn xếp hạng được chỉ định. Trường xếp hạng là đối tượng chấp nhận các giới hạn, được định nghĩa là toán tử. Bảng 9.1 hiển thị các toán tử mà chúng ta có thể sử dụng để chỉ định phạm vi.

Chúng tôi sử dụng truy vấn phạm vi để tìm kiếm trong phạm vi ngày hoặc số. Ví dụ, để lấy tất cả các bộ phim được thực hiện sau năm 1970, chúng tôi chỉ cần ghép truy vấn này lại với nhau.

**Bảng 9.1 Các nhà điều hành trong mộtphạm vitruy vấn**

|  |  |
| --- | --- |
| **Người điều hành** | **Nghĩa** |
| gt | Lớn hơn |
| gte | Lớn hơn hoặc bằng |
| nó | Ít hơn |
| lte | Nhỏ hơn hoặc bằng |

**324 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

**Liệt kê 9.14 Lấy phim sau năm 1970 bằng cách sử dụngphạm vitruy vấn**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"phạm vi": { "ngày phát hành": {

"gte": "01-01-1970"

}

}

},

"loại": [

{

"ngày phát hành": { "thứ tự": "tăng dần"

}

}

]

}

Trường release\_date khai báo toán tử gte với yêu cầu tìm kiếm: trong trường hợp này là năm 1970. Lưu ý rằng chúng tôi cũng sắp xếp các bộ phim theo thứ tự tăng dần theo ngày phát hành bằng cách sử dụng thuộc tính sort trong truy vấn. Do đó, các bộ phim được trả về được sắp xếp từ cũ nhất đến mới nhất.

Vì chúng ta đang thảo luận về truy vấn phạm vi, hãy tận dụng cơ hội này để xem lại toán học ngày trong truy vấn phạm vi. Chúng ta sẽ thảo luận về điều này trong phần sau.

**Toán dữ liệu trong truy vấn phạm vi**

Elasticsearch hỗ trợ toán học dữ liệu phức tạp trong các truy vấn. Ví dụ, chúng ta có thể yêu cầu công cụ thực hiện các tác vụ như sau:

* Tìm vềdoanh số bán sách của hai ngày trước (ngày hiện tại trừ đi hai ngày).
* Tìm lỗi bị từ chối truy cập trong 10 phút qua (giờ hiện tại trừ 10 phút).
* Nhận tweet theo tiêu chí tìm kiếm cụ thể từ năm ngoái.

Elasticsearch mong đợi một biểu thức dữ liệu cụ thể xử lý toán học dữ liệu. Phần đầu tiên của biểu thức được gọi là*ngày neo*, theo sau là hai thanh dọc (||) Hiện nay-nghĩa là ngày neo đang bị thao túng bằng cách thêm hoặc bớt một số đơn vị thời gian nhất định (phút, giây, năm, ngày, v.v.) và sau đó là thời gian chúng ta muốn thêm vào hoặc trừ đi khỏi ngày neo.

Giả sử chúng ta muốn lấy các bộ phim đã phát hành cách đây hai ngày. Chúng ta đặt ngày neo là hôm nay(tại thời điểm viết, 22-05-2023) và trừ đi hai ngày:

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"phạm vi": { "ngày phát hành": {

* 1. ***Ký tự đại diệntruy vấn* 325**

"lte": "22-05-2023||-2 ngày"



}

}

}

}

**Ngày neo theo sau là || trừ hai ngày**

Cácphạm vitruy vấn củaltengười điều hànhlấy giá trị ngày được biểu thị dưới dạng toán ngày. Trong trường hợp này, 22-05-2023 là ngày neo và chúng ta trừ hai ngày khỏi ngày đó (-2d). (Elasticsearch có một từ điển các chữ cái cho ngày toán học:*Và*trong nhiều năm,*Tôi*trong nhiều tháng,*TRONG*trong nhiều tuần,*ngày*trong nhiều ngày,*h*trong nhiều giờ,*tôi*trong vài phút,*S*trong vài giây, v.v.)

Thay vì chỉ định ngày hiện tại, Elasticsearch cho phép chúng ta sử dụng từ khóa:Hiện nay. CácHiện naytừ khóa biểu thị ngày hiện tại. Ví dụ, sử dụngbây giờ-1yđặt ngàyđến một năm trước:

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"phạm vi": { "ngày phát hành": {

"gte": "bây giờ-1y"

}

}

}

}

**Lấy tất cả các bộ phim từ năm ngoái: ngày hiện tại (được biểu thị là bây giờ) trừ đi một năm**

Chúng tôi xây dựngngày phát hànhbiểu hiện bằng cách sử dụngHiện nayvà trừ đi một năm.

Có nhiều tùy chọn để thao tác ngày tháng trong Elasticsearch và tôi khuyên bạn nênxem tài liệu để tìm hiểu thêm. Lưu ý rằng Elasticsearch không lưu trữ bộ nhớ đệm các truy vấn có chứa toán học ngày, do đó có những tác động về hiệu suất khi sử dụng toán học ngày trongphạm vitruy vấn.

Trong phần tiếp theo, chúng ta sẽ xem xét các truy vấn ký tự đại diện. Các truy vấn này không làm chúng ta thất vọng nếu chúng ta cung cấp tiêu chí tìm kiếm một phần, vì chúng ta sử dụng ký tự đại diện để xây dựng biểu thức.

#### Truy vấn ký tự đại diện

Như tên gọi của chúng, truy vấn ký tự đại diện cho phép chúng ta tìm kiếm các từ bị thiếu ký tự, hậu tố và tiền tố. Ví dụ, giả sử chúng ta muốn tìm kiếm tất cả các kết hợp có thể có của các bộ phim có tiêu đề kết thúc bằng father hoặc god, ngay cả khi chúng bị thiếu một ký tự duy nhất như god?ather. Đây là nơi chúng ta có thể sử dụng truy vấn ký tự đại diện. Truy vấn ký tự đại diện chấp nhận dấu hoa thị (\*) hoặc dấu chấm hỏi (?) trong từ tìm kiếm, như được mô tả trong bảng 9.2.

**Bảng 9.2 Các loại ký tự đại diện**

|  |  |
| --- | --- |
| **Tính cách** | **Sự miêu tả** |
| \*(dấu sao)  ?(dấu chấm hỏi) | Tìm kiếm không hoặc nhiều ký tự  Tìm kiếm một ký tự duy nhất |

**326 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

Hãy tìm kiếm những tài liệu có tiêu đề phim chứa từ bắt đầu bằng chữ god.

**Liệt kê 9.15 Tìm kiếm ký tự đại diện tronghoạt động**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"ký tự đại diện": {

"tiêu đề": {

"giá trị": "thần\*"

}

**Ký tự đại diện**

**loại truy vấn**

**Tìm kiếm trường giá trị bằng toán tử đại diện**

}

}

}

Batôitrứng(*Các Bố già*,T*Anh ta Bố giàII*, Và *Thành phố của Chúa*) nên là đã trả lại vìcái nàyký tự đại diệntruy vấn. Tất nhiên, bất kỳ bộ phim nào (Godzilla, God's Waiting List, v.v.) cũng có thể được tìm thấy vì chúng tôi mong đợi tất cả các tựa phim có tiền tố god (những bộ phim đó không có trong chỉ mục của chúng tôi, nhưng nếu có, truy vấn cũng sẽ trả về chúng).

**Truy vấn ký tự đại diện trên các trường văn bản**

Truy vấn trong danh sách 9.15 được chạy trêntiêu đềcánh đồng, đó là mộtchữkiểu dữ liệu. Vì các truy vấn cấp thuật ngữ không được phân tích, chúng tôi sử dụng chữ thường “thần”. Ngoài ra,tiêu đềtrường đã được lập chỉ mục vớitiêu chuẩnmáy phân tích, theo mặc định, sẽ viết thường chữ cái đầu tiên.

Nếu bạn muốn sử dụng mộttừ khóatrường thay vì mộtchữcánh đồng, thay đổitiêu đềĐẾNtiêu đề

.nguyên bản(cáctiêu đề.bản gốctrường được định nghĩa làtừ khóatrong chúng tôiphimlược đồ)và chạy truy vấn với giá trị “The God\*”. Nhưng nếu bạn bỏ “The” khỏi “The God\*” và chạy truy vấn, bạn sẽ không nhận được kết quả nào. Bởi vìtiêu đề.bản gốclà mộttừ khóatrường được nhập, giá trị được duy trì trong quá trình lập chỉ mục mà không cần bất kỳ phân tích văn bản nào.

Chúng ta có thể điều chỉnh các truy vấn của mình bằng cách đặt các ký tự đại diện ở bất kỳ đâu trong một từ. Ví dụ,truy vấn “g\*d” lấy hai bộ phim từ kho lưu trữ của chúng tôi: The Good, the Bad, and the Ugly và City of God. Nếu chúng ta muốn tìm sự khớp của một tiêu chí truy vấn nhất định trong một tài liệu trả về, chúng ta có thể sử dụng tô sáng (được thảo luận trong chương 8). Danh sách sau đây cho thấy cách tiếp cận này.

**Liệt kê 9.16 Tìm kiếm bằng toán tử đại diện trong một từ**

NHẬN phim/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"ký tự đại diện": {

"tiêu đề": {

"giá trị": "g\*d"

**Toán tử đại diện giữa các chữ cái của một từ**

}

}

* 1. ***Tiền tốtruy vấn* 327**

},

"điểm nổi bật": {

"các trường": {

"tiêu đề": {}

**Khối nổi bật cho phép chúng ta hình dung kết quả.**

}

}

}

Kết quả cho thấy có hai bộ phim trùng khớp:

"title": [ "Cái <em>Thiện</em>, Cái <em>Xấu</em> và Cái <em>Tệ</em>" ] "title": [ "Thành phố của <em>Chúa</em>"]

Chúng ta chỉ sử dụng ký tự đại diện ? nếu chúng ta muốn khớp với một ký tự. Ví dụ, "value": "go?ather" tìm kiếm tất cả các từ khớp với ký tự thứ ba tại vị trí của ký tự đại diện. Chúng ta cũng có thể kết hợp nhiều ký tự ?, chẳng hạn như “g???ather”.

**Các truy vấn tốn kém**

Một số truy vấn tốn kém cho công cụ chạy vì cách chúng tôi triển khai chúng.ký tự đại diệntruy vấn là một, cùng vớiphạm vi,tiền tố,mờ nhạt,biểu thức chính quy, Vàtham giatruy vấn. Sử dụng truy vấn như vậy đôi khi có thể không ảnh hưởng đến hiệu suất máy chủ,nhưng việc sử dụng quá nhiều các truy vấn tốn kém này có thể làm mất ổn định cụm, dẫn đến trải nghiệm không tốt cho người dùng.

Elasticsearch cho phép chúng ta thực hiện các truy vấn tốn kém trên cụm theo ý muốn của chúng ta. Tuy nhiên, nếu chúng ta muốn dừng việc thực hiện các truy vấn này trên cụm, chúng ta có thể thiết lậpcho phép\_truy\_cập\_đắt\_tiềnthuộc tính choSAIvề cài đặt cụm:

ĐẶT \_cluster/cài đặt

{

"tạm thời": { "search.allow\_expensive\_queries": "sai"

}

}

Bằng cách tắtcho phép\_truy\_cập\_đắt\_tiền, chúng tôi bảo vệ cụm khỏi tình trạng quá tải.

Lưu ý rằng nếu chúng ta đặtcho phép\_truy\_cập\_đắt\_tiềnĐẾNSAI,ký tự đại diệntruy vấn không được thực hiện. Lỗi sau sẽ được đưa ra nếu chúng ta thử:"lý do": "[ký tự đại diện] truy vấn

không thể thực hiện được khi 'search.allow\_expensive\_queries' được đặt thành false."

Ký tự đại diện lấy kết quả với các ký tự bị thiếu trong một từ hoặc một câu. Đôi khi chúng tamuốn truy vấn các từ có tiền tố—đó là lúc các truy vấn tiền tố xuất hiện, như sẽ thảo luận tiếp theo.

#### 9.8 Truy vấn tiền tố

Chúng ta có thể muốn truy vấn các từ bằng cách sử dụng phần đầu của một từ (tiền tố), như Leo cho Leonardo hoặc Mar cho Marlon Brando, Mark Hamill hoặc Martin Balsam. Chúng ta có thể sử dụng truy vấn tiền tố để tìm các bản ghi khớp với tiền tố, như được hiển thị trong danh sách sau.

**328 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

**Liệt kê 9.17 Sử dụng mộttiền tốtruy vấn**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"tiền tố": {

**Chỉ định tiền tố**

**loại truy vấn**

"diễn viên.bản gốc": { "giá trị": "Mar"

}

}

}

}

**Truy vấn các từ bắt đầu bằng “Mar”**

Truy vấn này lấy ba bộ phim có diễn viên tên là Marlon, Mark và Martin khi chúng tôi tìm kiếm tiền tố “Mar”. Lưu ý rằng chúng tôi đang chạy truy vấn tiền tố trêntrường actors.original là kiểu dữ liệu từ khóa.

**GHI CHÚ**Truy vấn tiền tố tốn kém và có thể làm mất ổn định cụm. Xem thanh bên “Truy vấn tốn kém” trong phần 9.7 để biết cách ngăn cụm quá tải và phần 9.8.2 để tăng tốc truy vấn tiền tố.

* + 1. Rút gọn truy vấn

Như chúng ta đã thảo luận ở đầu chương này, chúng ta không cần thêm đối tượng bao gồm các giá trị ở cấp khối trường. Thay vào đó, chúng ta có thể tạo phiên bản rút gọn để ngắn gọn.

**Liệt kê 9.18 Rút gọntiền tốsử dụng truy vấn**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"tiền tố": { "diễn viên.bản gốc": "Leo"

}

}

}

Vì chúng tôi muốn tìm các trường khớp trong kết quả, chúng tôi thêm phần tô sáng vào truy vấn. Thêm khối tô sáng vào truy vấn tiền tố sẽ làm nổi bật một hoặc nhiều trường khớp.

**Liệt kê 9.19 Atiền tốtìm kiếm với tô sáng**

NHẬN phim/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"tiền tố": { "diễn viên.bản gốc": {

"giá trị": "Mar"

* 1. ***Tiền tốtruy vấn* 329**

}

}

}, **Làm nổi bậtđã khớp**

"điểm nổi bật": {

"các trường": {

**các diễn viên trong kết quả**

"diễn viên.bản gốc": {}

}

}

}

Vì chúng tôi không muốn nguồn được trả về trong phản hồi ("\_source": false), nên kết quả sau đây sẽ đánh dấu vị trí tiền tố khớp với từ:

"lượt truy cập" : [{

..

"điểm nổi bật" : {

"diễn viên.origen" : ["<em>Marlon Brando</em>"]

}

},

{

..

"điểm nổi bật" : {

"diễn viên.origen" : ["<em>Martin Balsam</em>"]

}

},

{

..

"điểm nổi bật" : {

"actors.original": ["<em>Mark Hamill</em>"]

}

}]

Chúng tôi đã đề cập trước đó rằng việc chạy các truy vấn tiền tố gây ra sức ép tính toán. May mắn thay, có một cách để tăng tốc các truy vấn kém hiệu suất như vậy.

* + 1. Tăng tốc truy vấn tiền tố

Truy vấn tiền tố chạy chậm vì công cụ phải lấy kết quả dựa trên tiền tố (bất kỳ từ nào có chữ cái). Nhưng có một cơ chế để tăng tốc: sử dụng tham số index\_ prefixes trên trường.

Chúng ta có thể thiết lậptiền tố chỉ mụckhi phát triển lược đồ ánh xạ. Ví dụ, định nghĩa ánh xạ trong danh sách sau đây đặttiêu đềlĩnh vực (hãy nhớ,tiêu đềlà mộtchữkiểu dữ liệu) với một tham số bổ sung,tiền tố chỉ mục, trên một chỉ mục mới (phim\_hit\_boxoffice) mà chúng ta đang tạo cho bài tập này.

 **Liệt kê 9.20 Chỉ mục mới vớitiền tố chỉ mụctham số**

ĐẶT boxoffice\_hit\_movies

{

"ánh xạ": { "thuộc tính": {

"tiêu đề":{

**Tạo một chỉ mục mới chỉ có một thuộc tính, tiêu đề**

**330 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

"kiểu": "văn bản", "tiền tố\_chỉ\_mục":{}

}

}

}

}

**Đặt tham số index\_prefixes trên trường tiêu đề**

Thuộc tính tiêu đề duy nhất bao gồm một thuộc tính bổ sung, index\_prefixes. Thuộc tính này chỉ ra cho công cụ rằng trong quá trình lập chỉ mục, nó sẽ tạo trường với các tiền tố được xây dựng sẵn và lưu trữ các giá trị đó. Ví dụ, giả sử chúng ta lập chỉ mục một tài liệu mới:

ĐẶT boxoffice\_hit\_movies/\_doc/1

{

"title":"Đấu sĩ"

}

Bởi vì chúng tôi đặt index\_prefixes trên trường tiêu đề trong danh sách 9.20, Elasticsearch lập chỉ mục các tiền tố có kích thước ký tự tối thiểu là 2 và kích thước ký tự tối đa là5 theo mặc định. Theo cách này, khi chúng ta chạy truy vấn tiền tố, nó không cần phải tính toán tiền tố. Thay vào đó, nó lấy chúng từ bộ nhớ.

Chúng ta cũng có thể thay đổi kích thước min và max mặc định của các tiền tố mà Elasticsearch tạo ra trong quá trình lập chỉ mục. Điều này được thực hiện bằng cách điều chỉnh kích thước của đối tượng index\_prefixes.

**Liệt kê 9.21 Độ dài ký tự tùy chỉnh chotiền tố chỉ mục**

ĐẶT boxoffice\_hit\_movies\_custom\_prefix\_sizes

{

"ánh xạ": { "thuộc tính": {

"tiêu đề":{

"kiểu": "văn bản", "tiền tố\_chỉ\_số":{

"min\_chars":4, "max\_chars":10

}

**Đặt số lượng tối thiểucủa các ký tự cho tiền tố**

**Đặt số lượng ký tự tối đa cho tiền tố**

}

}

}

}

Trong danh sách, chúng tôi yêu cầu công cụ tạo trước các tiền tố có giá trị tối thiểu và tối đađộ dài ký tự lần lượt là 4 và 10 ký tự. Lưu ý rằng min\_chars phải lớn hơn 0 và max\_chars phải nhỏ hơn 20 ký tự. Theo cách này, chúng ta có thể tùy chỉnh các tiền tố mà Elasticsearch sẽ tạo trước trong quá trình lập chỉ mục.

#### Truy vấn mờ

Lỗi chính tả thường gặp trong quá trình tìm kiếm. Chúng tôi có thể tìm kiếm một từ có một hoặc nhiều chữ cái không đúng: ví dụ, tìm kiếm phim “rama” thay vì phim “drama”. Tìm kiếm có thể sửa truy vấn này và trả về phim “drama” thay thế

***9.9 Mờtruy vấn* 331**

của việc thất bại và không trả về gì cả. Nguyên tắc đằng sau loại truy vấn này được gọi là fuzz-*không có*và Elasticsearch sử dụng các truy vấn mờ để bỏ qua lỗi chính tả.

Độ mờ là quá trình tìm kiếm các thuật ngữ tương tự dựa trên thuật toán khoảng cách Levenshtein (còn được gọi là khoảng cách chỉnh sửa). Khoảng cách Levenshtein là số ký tự cần được hoán đổi để tìm các từ tương tự. Ví dụ, tìm kiếm "cake" có thể tìm ra "take", "bake", "lake", "make", v.v. nếu chúng ta sử dụng truy vấn mờ với độ mờ (khoảng cách chỉnh sửa) được đặt thành 1. Truy vấn sau đây sẽ trả về tất cả các phim chính kịch vì áp dụng độ mờ là 1 cho "rama" sẽ trả về "drama".

**Liệt kê 9.22 Amờ nhạttruy vấn trong hành động**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"mờ": {

"thể loại": { "giá trị": "rama", "mờ nhạt": 1

}

}

},

"điểm nổi bật": {

"các trường": {

"thể loại": {}

}

}

}

Trong ví dụ này, chúng tôi sử dụng khoảng cách chỉnh sửa là 1 (một ký tự) để lấy các từ tương tự. Bạn cũng có thể thử xóa một ký tự khỏi giữa từ, như "dama" hoặc“dram”; các truy vấn này cũng đưa ra kết quả dương khi độ mờ được đặt thành 1.

**GHI CHÚ**Không giống như truy vấn ký tự đại diện, sử dụng toán tử ký tự đại diện (\* hoặc ?), truy vấn mờ không sử dụng toán tử. Thay vào đó, nó lấy các từ tương tự bằng thuật toán khoảng cách chỉnh sửa Levenshtein.

Nếu chúng ta bỏ thêm một chữ cái nữa (ví dụ, "value": "ama" với độ mờ được đặt thành 1), truy vấn mờ trong danh sách 9.22 không trả về bất kỳ kết quả nào. Vì chúng ta thiếu hai chữ cái, chúng ta cần đặt khoảng cách chỉnh sửa thành 2 để giải quyết vấn đề này.

**Liệt kê 9.23 Amờ nhạttruy vấn với hai chữ cái bị thiếu trong một từ**

NHẬN phim/\_tìm kiếm

{

"truy vấn": {

"mờ": {

"thể loại": { "giá trị": "ama",

**Một từ bị thiếu chữ cái**

**332 CPHẦN9*Tìm kiếm theo cấp độ thuật ngữ***

"mờ": 2

}

}

}

}

**Thiết lập độ mờ thành 2 sẽ bỏ qua việc thay thế/sửa đổi hai chữ cái.**

Đây có thể là một cách vụng về để xử lý mọi thứ vì chúng ta có thể không biết liệu người dùng đã nhập sai một chữ cái hay một vài chữ cái. Đây là lý do tại sao Elasticsearch cung cấp một thiết lập mặc địnhđối với độ mờ: cài đặt TỰ ĐỘNG. Nếu thuộc tính độ mờ không được cung cấp, cài đặt mặc định là TỰ ĐỘNG sẽ được giả định. Cài đặt TỰ ĐỘNG xác định khoảng cách chỉnh sửa dựa trên độ dài của từ, như thể hiện trong bảng 9.3. Tốt nhất là nên sử dụng cài đặt mặc định là TỰ ĐỘNG cho thuộc tính độ mờ trừ khi chúng ta biết chính xác trường hợp sử dụng đang có.

**Bảng 9.3 Sử dụngcáiTự động làm mờcài đặt**

|  |  |  |
| --- | --- | --- |
| **Độ dài từ tính bằng ký tự** | **Độ mờ (khoảng cách chỉnh sửa)** | **Giải thích** |
| 0 đến 2 | 0 | Nếu từ ngắn hơn hai ký tự, tính năng mờ không được áp dụng. Điều này có nghĩa là không thể sửa các từ viết sai chính tả. |
| 3 đến 5 | 1 | Nếu độ dài của từ là 3 đến 5 ký tự, khoảng cách chỉnh sửa là 1 sẽ được áp dụng. |
| Hơn 5 | 2 | Nếu độ dài của từ dài hơn 5 ký tự, khoảng cách chỉnh sửa sẽ được áp dụng là 2. |

Chúng ta hãy kết thúc ở đây. Chúng ta đã học được rất nhiều về tìm kiếm cấp thuật ngữ trong chương này. Trong khi tìm kiếm cấp thuật ngữ giúp tìm câu trả lời trong dữ liệu có cấu trúc, sức mạnh thực sự của công cụ tìm kiếm nằm ở khả năng tìm kiếm dữ liệu phi cấu trúc. Trong Elasticsearch, dữ liệu phi cấu trúc được liên kết với tìm kiếm toàn văn bản—tìm kiếm các trường văn bản và đưa ra kết quả với điểm liên quan. Chúng ta sẽ xem xét chi tiết các truy vấn toàn văn bản trong chương tiếp theo.

#### Bản tóm tắt

* Tìm kiếm theo cấp thuật ngữ được thực hiện trên dữ liệu có cấu trúc như số, từ khóa, Boolean và ngày tháng.
* Tìm kiếm cấp thuật ngữ tạo ra kết quả nhị phân: tìm kiếm dẫn đến kết quả hoặc không có kết quả nào. Không có kịch bản khớp nào có thể xảy ra.
* Các truy vấn cấp độ thuật ngữ không được phân tích, nghĩa là khi áp dụng chochữCác trường đang được phân tích văn bản trong quá trình lập chỉ mục có thể mang lại kết quả không chính xác hoặc không có kết quả.
* Có nhiều truy vấn tìm kiếm theo cấp thuật ngữ:thuật ngữ,điều khoản,tiền tố,phạm vi,mờ nhạtvà những người khác.
* MỘTthuật ngữtruy vấn tìm kiếm một thuật ngữ duy nhất đối với một trường, trong khi mộtđiều khoản(số nhiều) truy vấn tìm kiếm nhiều giá trị trong một trường duy nhất.
* MỘTphạm vitruy vấngiúp tìm kiếm dữ liệu trong một phạm vi nhất định: ví dụ, tìm kiếm các tội phạm xảy ra ở London trong tháng trước.

***Bản tóm tắt* 333**

* + MỘTký tự đại diệntruy vấn sử dụng\*Và?các toán tử để lấy kết quả.
  + Fuzziness sử dụng khoảng cách chỉnh sửa của Levenshtein để lấy các từ có vẻ ngoài tương tự. Elasticsearch sử dụngmờ nhạttruy vấn để hỗ trợ lỗi chính tả của người dùng.
  + MỘTtiền tốtruy vấn lấy kết quả được cung cấp tiền tố (không cần chỉ định toán tử đại diện). Vì các hoạt động tiền tố tốn kém để thực hiện trên chỉ mục trực tiếp, chúng ta có thể yêu cầu Elasticsearch tạo trước chúng tại thời điểm lập chỉ mục để tránh tìm thấy chúng trong giai đoạn truy vấn trực tiếp.

*Tìm kiếm toàn văn*

***Chương này bao gồm***

* Tổng quan về truy vấn toàn văn
* Làm việc thông qua các truy vấn khớp
* Xem các cụm từ khớp, nhiều khớp và các cụm từ kháctruy vấn
* Xem xét các chuỗi truy vấn và các truy vấn chuỗi truy vấn đơn giản

Trong chương trước, chúng ta đã xem xét tìm kiếm theo cấp thuật ngữ, cơ chế chúng ta sử dụng để tìm kiếm dữ liệu có cấu trúc. Mặc dù tìm kiếm dữ liệu có cấu trúc rất quan trọng, nhưng sức mạnh của các công cụ tìm kiếm hiện đại được phát huy hiệu quả và hiệu quả để chạy khi chúng ta tìm kiếm dữ liệu phi cấu trúc. Elasticsearch là một trong những công cụ tìm kiếm hiện đại như vậy, dẫn đầu trong việc tìm kiếm dữ liệu phi cấu trúc có liên quan.

Elasticsearch cung cấp khả năng tìm kiếm dữ liệu phi cấu trúc thông qua các truy vấn tìm kiếm toàn văn. Tìm kiếm toàn văn là tất cả về tính liên quan: lấy các tài liệu có liên quan đến tìm kiếm của người dùng. Ví dụ, khi tìm kiếm từ Java trong một hiệu sách trực tuyến, người ta không nên mong đợi nhận được thông tin chi tiết về đảo Java của Indonesia hoặc cà phê ép ướt được trồng trên hòn đảo này.

**334**

***10.1 Tổng quan* 335**

Trong chương này, chúng tôi xem xét cơ chế tìm kiếm dữ liệu phi cấu trúc bằng cách sử dụng API tìm kiếm toàn văn. Elasticsearch cung cấp một số truy vấn toàn văn trongdạng match, query\_string và các dạng khác. Vì match queries là các truy vấn được sử dụng phổ biến nhất khi làm việc với toàn văn, chúng tôi dành một phần lớn chương này cho các match queries khác nhau. Chúng tôi cũng làm việc thông qua các tìm kiếm chuỗi truy vấn, tương đương với việc sử dụng tìm kiếm yêu cầu URI nhưng có nội dung yêu cầu tương tự như Query DSL.

#### Tổng quan

Khi chúng ta tìm kiếm trên một trang web bán lẻ như Amazon hoặc eBay, chúng ta thấy kết quả gần với những gì chúng ta đang tìm kiếm. Nếu kết quả không như mong đợi, chúng ta sẽ bày tỏ sự thất vọng và thề sẽ không bao giờ quay lại trang web hoặc ứng dụng đó nữa, phải không? Trải nghiệm tìm kiếm của người dùng rất quan trọng để giữ chân khách hàng hài lòng.

Tính liên quan là tất cả về mức độ phù hợp của kết quả tìm kiếm và mức độ liên quan chặt chẽ của chúng với những gì người dùng đang tìm kiếm. Elasticsearch phát triển mạnh mẽ nhờ tạo ra kết quả có liên quan với tốc độ và độ chính xác bằng cách sử dụng các thuật toán liên quan tinh vi.

Khi chúng ta nói về sự liên quan, hai biện pháp thường xuất hiện trong đầu: độ chính xác và khả năng thu hồi. Sự liên quan được đo bằng hai yếu tố này và việc hiểu chúng ở mức độ cao là rất quan trọng (mặc dù không phải là yếu tố quyết định).

* + 1. Độ chính xác

*Độ chính xác*là tỷ lệ phần trăm các tài liệu có liên quan trong tổng số tài liệu được trả về. Khi một truy vấn trả về kết quả, không phải tất cả các kết quả đều liên quan trực tiếp đến truy vấn. Có thể có một số tài liệu không liên quan trong kết quả.

Ví dụ, giả sử chúng ta đang tìm kiếm một chiếc tivi 4K từ một nhà sản xuất cụ thể (ví dụ như LG) và chúng ta nhận được 10 kết quả. Không phải tất cả các kết quả này đều có liên quan: một số là máy ảnh 4K (xem hình 10.1) và một số là máy chiếu, vì LG cũng sản xuất những sản phẩm đó.

**Tất cả các TV trả về đều có kết quả chính xác. Chúng được gọi là kết quả dương tính thật.**

**Kết quả bao gồm một số sản phẩm không phải TV như máy ảnh và máy chiếu không nên nằm trong tập kết quả. Chúng được gọi là kết quả dương tính giả.**

Tivi

Độ chính xác =

TV + các sản phẩm khác như máy ảnh và máy chiếu

**Hình 10.1 Một ví dụ chính xác trong hành động, trả về kết quả tìm kiếm cho tivi 4K**

**336 CPHẦN10*Tìm kiếm toàn văn***

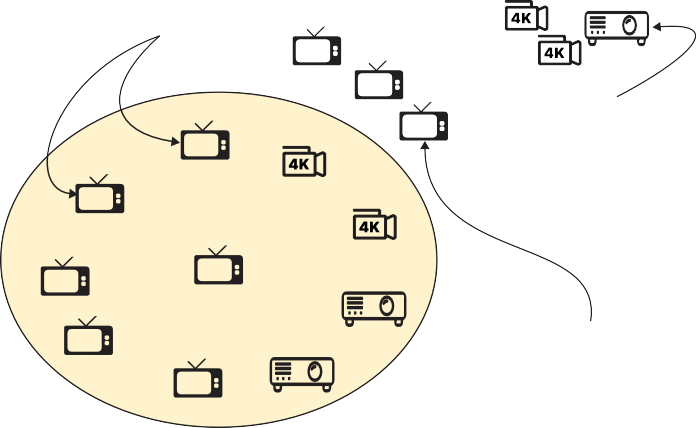
Trong hình, kết quả bao gồm các kết quả dương tính thật (tài liệu có liên quan) cũng như các kết quả dương tính giả (tài liệu không liên quan). Độ chính xác là về số lượng tài liệu được truy xuất có liên quan. Trong trường hợp này, chúng ta biết 6 tài liệu là TV trong số 10 tài liệu kết quả, do đó 6 tài liệu có liên quan và 4 tài liệu còn lại không liên quan. Do đó, độ chính xác là

độ chính xác = 6 / 10 × 100% = 60%

Điều này cho chúng ta biết rằng chỉ có 60% tài liệu được trả về là có liên quan. Một số tài liệu không liên quan trực tiếp đến truy vấn (tài liệu không liên quan) cũng xuất hiện trong kết quả vì nhiều lý do khác nhau.

* + 1. Nhớ lại

Recall là mặt trái của đồng xu. Nó đo lường số lượng tài liệu được trả về có liên quan. Ví dụ, có thể có một số kết quả có liên quan hơn (TV) không được trả về (tức là bị bỏ qua) như một phần của tập kết quả. Phần trăm tài liệu có liên quan được truy xuất được gọi là recall (xem hình 10.2).

**Tất cả các TV trả về đều có kết quả chính xác. Chúng được gọi là kết quả dương tính thật.**

**Những sản phẩm không phải TV này không liên quan và không nên trả lại. Chúng được gọi là hàng âm tính thật.**

**Những chiếc TV này đáng lẽ phải được trả lại nhưng đã bị bỏ qua. Chúng được gọi là âm tính giả.**

TV đã được thu hồi

Thu hồi =

Tất cả các TV đủ điều kiện

**Hình 10.2 Biện pháp thu hồi đang được áp dụng cho ví dụ tìm kiếm tivi 4K của chúng tôi**

Như hình minh họa, ba chiếc TV đã rơi vào thùng tìm kiếm nhưng không được trả về. Những thứ này được gọi là kết quả âm tính giả. Mặt khác, một số sản phẩm như máy ảnh và máy chiếu không được trả về vì chúng thực sự không liên quan và do đó không được mong đợi

* 1. ***Tổng quan* 337**

là một phần của kết quả. Đây là những kết quả âm tính thực sự. Trong kịch bản hiện tại, khả năng thu hồi được tính là

Thu hồi = 6 / (6+3) × 100% = 66,6%

Về mặt lý tưởng, chúng ta muốn độ chính xác và độ thu hồi phải khớp hoàn hảo, không bỏ sót (không bỏ sót tài liệu liên quan nào). Tuy nhiên, điều này gần như không thể vì các biện pháp này luôn chống lại nhau. Chúng tỷ lệ nghịch với nhau: độ chính xác càng cao (số lượng tài liệu khớp nhất), độ thu hồi càng thấp (số lượng tài liệu trả về). Hình 10.3 cho thấy mối quan hệ nghịch đảo giữa hai yếu tố này.

Nhớ lại

**Độ chính xác càng cao thì**

**ít kết quả trả về hơn. Ngược lại, khi độ chính xác bị giảm, khả năng thu hồi sẽ lấy được nhiều tài liệu hơn.**

số

Độ chính xác

ber của tài liệu

**Hình 10.3 Độ chính xác và khả năng thu hồi luôn trái ngược nhau.**

Chúng ta cần đảm bảo kết quả trả về cân bằng giữa chiến lược độ chính xác và chiến lược thu hồi. Hình 10.4 tóm tắt độ chính xác và thu hồi.

**Hình 10.4 Các công thức cho độ chính xácvà nhớ lại**

Tài liệu có liên quan đã được thu thập

Độ chính xác

**=**

Đã lấy lại tài liệu trong tập kết quả

Tài liệu có liên quan đã được thu thập

Nhớ lại

**=**

Các tài liệu có liên quan trong kho dữ liệu

Chúng ta có thể điều chỉnh kết quả để có độ chính xác và khả năng thu hồi khi thiết kế và thực hiện truy vấn.Chúng ta cũng có thể sử dụng truy vấn khớp, bộ lọc và tăng cường để điều chỉnh độ chính xác và độ thu hồi để tinh chỉnh sự cân bằng của chúng. Trong chương này, chúng ta không điều chỉnh trực tiếp các tham số này mà thay vào đó, chúng ta sẽ làm việc trên các truy vấn để xem kết quả bị ảnh hưởng như thế nào. Nếu bạn muốn tìm hiểu thêm về độ chính xác và độ thu hồi trong bối cảnh truy xuất thông tin, tôi khuyên bạn nên xem bài viết này từ Google dành cho Nhà phát triển:<http://mng.bz/a1eo>.

**338 CPHẦN10*Tìm kiếm toàn văn***

Bây giờ chúng ta đã hiểu khái niệm về sự liên quan và các yếu tố kiểm soát (độ chính xác và khả năng thu hồi), phần còn lại của chương này sẽ dành để làm việc với các truy vấn toàn văn. Chúng ta bắt đầu bằng cách lấy một số dữ liệu mẫu.

#### Dữ liệu mẫu

Trong chương này, chúng ta làm việc với một hiệu sách hư cấu. Chúng ta lập chỉ mục một tập hợp gồm 50 cuốn sách kỹ thuật vào một chỉ mục có tên là books bằng cách gọi API \_bulk. Chúng ta không điều chỉnh ánh xạ cho phần dữ liệu mẫu này, vì vậy bạn có thể lập chỉ mục các cuốn sách như hiện tại. Dữ liệu cho các cuốn sách có sẵn trên GitHub tại<http://mng.bz/zX2g>và tập lệnh để lập chỉ mục các cuốn sách là tại<http://mng.bz/0KzW>; các tập tin cũng có sẵn trên trang web của cuốn sách (<https://www.manning.com/books/elasticsearch-in-action-second-edition>).

Elasticsearch cung cấp một số truy vấn toàn văn. Vì mỗi loại truy vấn có nhiều chi tiết triển khai nên chúng được mô tả trong các phần riêng biệt để dễ dàng thực hiện.Truy vấn đầu tiên chúng ta xem xét là truy vấn match\_all, trả về mọi thứ.

#### Truy vấn match\_all

Như tên gọi của nó, truy vấn match all (match\_all) sẽ lấy tất cả các tài liệu trong chỉ mục. Vì truy vấn này được mong đợi sẽ trả về tất cả các tài liệu có sẵn, nên đây là đối tác hoàn hảo để tôn trọng 100% khả năng thu hồi.

* + 1. Xây dựng truy vấn match\_all

Chúng tôi tạo truy vấn với đối tượng match\_all, không truyền tham số nào. Mã trong danh sách sau đây cho thấy cách xây dựng truy vấn match\_all.

**Liệt kê 10.1 Lấy tất cả các tài liệu bằng cách sử dụngkhớp\_tất\_cả**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"match\_all": { }

**Truy vấn match\_all**

**không có tham số**

}

}

Truy vấn này trả về tất cả các tài liệu có trong chỉ mục sách. Điểm đáng chú ý là phản hồi cho biết mỗi cuốn sách có điểm là 1.0 theo mặc định:

{

**"điểm\_tối\_đa" : 1.0,**

...

"lượt truy cập" : [{

"\_index" : "sách",

"\_type" : "\_doc",

"\_id" : "2",

**"\_điểm" : 1.0,**

"\_nguồn" : {

"title" : "Java hiệu quả",

...

}

* 1. ***Trận đấu không cótruy vấn* 339**

},

{

},

...

}]

...

**"\_điểm" : 1.0,**

"\_nguồn" : {

"title" : "Java: Hướng dẫn cho người mới bắt đầu",

...

Chúng ta có thể tăng điểm nếu cần bằng cách chỉ cần sửa đổi truy vấn. Danh sách tiếp theo sẽ cho biết cách thực hiện.

**Liệt kê 10.2 Tăng cường truy vấn với điểm số được xác định trước**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"khớp\_tất\_cả": {

"tăng cường": 2

**Truy vấn match\_all đặt điểm là 2 cho tất cả các tài liệu được trả về.**

}

}

}

Chúng tôi thêm tham số tăng cường vào truy vấn để trả về tất cả các tài liệu có điểm tăng cường.

10.3.2 Dạng rút gọn của truy vấn match\_all

Chúng tôi đã viết một truy vấn match\_all với một thân truy vấn trong danh sách 10.1. Tuy nhiên, việc cung cấp thân là thừa. Truy vấn tương tự có thể được viết lại theo định dạng ngắn hơn như sau:

NHẬN sách/\_tìm kiếm

Đằng sau hậu trường, Elasticsearch thực hiện truy vấn match\_all với boost mặc định là 1 khi thân truy vấn không được cung cấp. Trừ khi chúng ta muốn thay đổi giá trị boost, chúng ta có thể gọi điểm cuối tìm kiếm mà không cần body.

#### Truy vấn match\_none

Trong khi truy vấn match\_all trả về tất cả kết quả từ một chỉ mục hoặc nhiều chỉ mục, truy vấn ngược lại, match\_none, không trả về kết quả nào. Danh sách sau đây hiển thị cú pháp.

**Liệt kê 10.3không có trận đấutruy vấn**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"không khớp": {}

**Một truy vấn mà**

**không khớp với bất cứ thứ gì**

}

}

**340 CPHẦN10*Tìm kiếm toàn văn***

Truy vấn match\_none hữu ích hơn trong các tình huống mà bạn muốn loại trừ có điều kiện tất cả các tài liệu, thường dựa trên một số logic bên ngoài trong ứng dụng của bạn. Ví dụ, nếu một điều kiện nhất định trong ứng dụng của chúng tôi được đáp ứng, chúng tôi có thể chèn truy vấn match\_none vào mệnh đề must của truy vấn bool của bạn để đảm bảo không có tài liệu nào được trả về.

Giả sử chúng ta có một tính năng ứng dụng cho phép người quản trị "khóa" cơ sở dữ liệu phim để bảo trì hoặc nâng cấp. Khi cơ sở dữ liệu bị khóa, bạn muốn tất cả các truy vấn tìm kiếm không trả về kết quả nào thay vì đưa ra kết quả sai. Trongtrường hợp này, bạn có thể chèn truy vấn match\_none dựa trên điều kiện khóa trong ứng dụng của bạn. Truy vấn bool được viết trong đoạn mã dưới đây minh họa trường hợp sử dụng như vậy:

{

"truy vấn": {

"bool": {

"phải": [{

"không khớp": {}

}

]

}

}

}

Truy vấn sẽ được chèn match\_none trong các điều kiện cụ thể (cơ sở dữ liệu đang được nâng cấp), nghĩa là không có truy vấn nào trả về kết quả.

Trong phần tiếp theo, chúng ta sẽ tìm hiểu về truy vấn khớp. Hầu hết các truy vấn chúng ta sử dụng khi làm việc với Elasticsearch đều là truy vấn khớp ở dạng này hay dạng khác.

#### Truy vấn khớp

Truy vấn khớp là truy vấn phổ biến và mạnh mẽ nhất cho nhiều trường hợp sử dụng. Đây là truy vấn tìm kiếm toàn văn trả về các tài liệu khớp với tiêu chí đã chỉ định. Truy vấn khớp có thể được sửa đổi để bao gồm nhiều tùy chọn khác nhau.

* + 1. Định dạng của truy vấn khớp

Trước tiên chúng ta hãy xem định dạng của truy vấn khớp:

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

**Kiểu truy vấn**

**là trận đấu.**

"FIELD": "TÌM KIẾM VĂN BẢN"

}

}

}

**Truy vấn này mong đợi tiêu chí được chỉ định dưới dạng cặp tên-giá trị.**

Truy vấn khớp mong đợi tiêu chí tìm kiếm được xác định là giá trị trường. Trườngcó thể là bất kỳ trường văn bản nào trong tài liệu có giá trị cần khớp. Giá trị có thể là một từ hoặc nhiều từ và có thể là chữ hoa, chữ thường hoặc CamelCase.

***10.5 Trận đấutruy vấn* 341**

Chúng ta cũng có thể truyền một số tham số bổ sung trong dạng đầy đủ của truy vấn khớp. Cho đến nay, chúng ta đã thảo luận về dạng rút gọn của truy vấn khớp. Sau đây là một ví dụ về dạng đầy đủ:

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"CÁNH ĐỒNG": {

**Khai báo FIELD là một đối tượng**

**với bổ sungtham số Thuộc tính truy vấn**

"query":"<VĂN BẢN TÌM KIẾM>",

"<tham số>":"<MY\_PARAM>",

}

}

}

}

**giữ văn bản tìm kiếm.**

**Tham số (trình phân tích, toán tử, prefix\_length, độ mờ, v.v.) mong đợi một giá trị được thiết lập.**

Chúng ta có thể tìm kiếm trên nhiều chỉ mục bằng cách cung cấp các chỉ mục được phân tách bằng dấu phẩy trong URL tìm kiếm:

NHẬN sách\_mới,kinh\_tế,bán\_hết,tội\_hình\* /\_tìm\_kiếm

{

...

}

Có thể cung cấp bất kỳ số lượng chỉ mục nào khi gọi điểm cuối \_search, bao gồm cả ký tự đại diện.

**GHI CHÚ**Nếu chúng ta bỏ qua chỉ mục (hoặc các chỉ mục) trong yêu cầu tìm kiếm, chúng ta thực sựtìm kiếm toàn bộ chỉ mục. Ví dụ: GET \_search { ... } tìm kiếm trên tất cả các chỉ mục trong cụm.

* + 1. Tìm kiếm bằng truy vấn khớp

Bây giờ chúng ta đã biết định dạng cho truy vấn khớp, hãy xem một ví dụ trong đó chúng ta muốn tìm kiếm sách Java với Java trong trường tiêu đề. Danh sách sau đây đặt trường tiêu đề thành từ “Java” làm văn bản để tìm kiếm.

**Liệt kê 10.4 Tìm kiếm Javasách**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tiêu đề": "Java"

}

**Truy vấn khớp**

**đang hành động**

**Đặt tiêu chí tìm kiếm, tìm kiếm từ “Java” trong trường tiêu đề**

}

}

Chúng tôi tạo truy vấn khớp với tiêu chí tìm kiếm một từ trong trường tiêu đề. Elasticsearch sẽ tìm nạp tất cả các tài liệu khớp với từ Java trong trường tiêu đề, như mong đợi.

**342 CPHẦN10*Tìm kiếm toàn văn***

* + 1. Phân tích truy vấn khớp

Trong chương 9, chúng ta thấy rằng các truy vấn cấp thuật ngữ không được phân tích. Mặt khác,các truy vấn khớp hoạt động trên các trường văn bản được phân tích. Các trình phân tích giống nhau được sử dụng trong quá trình lập chỉ mục (trừ khi các truy vấn tìm kiếm của chúng tôi được định nghĩa rõ ràng bằng các trình phân tích khác nhau) xử lý các từ tìm kiếm trong các truy vấn khớp. Nếu một trình phân tích chuẩn (trình phân tích mặc định) được sử dụng trong quá trình lập chỉ mục của một tài liệu, các từ tìm kiếm được phân tích bằng cùng một trình phân tích chuẩn trước khi thực hiện tìm kiếm.

Ngoài ra, trình phân tích chuẩn áp dụng cùng một bộ lọc mã thông báo chữ thường (hãy nhớ rằng, bộ lọc mã thông báo chữ thường được áp dụng trong quá trình lập chỉ mục) cho các từ tìm kiếm. Do đó, nếu chúng ta cung cấp các từ khóa tìm kiếm chữ hoa, chúng sẽ được chuyển đổi thành chữ thường và được tìm kiếm theo chỉ mục đảo ngược. Ví dụ, nếu chúng ta thay đổigiá trị tiêu đề để sử dụng tiêu chí viết hoa như "tiêu đề": "JAVA" và chạy lại truy vấn, kết quả sẽ giống như truy vấn tìm kiếm trong danh sách 10.4. Nếu chúng ta thay đổi giá trị tiêu đề thành chữ thường hoặc sửa đổi theo bất kỳ cách nào khác (java, jaVA, v.v.), truy vấn vẫn trả về kết quả tương tự.

* + 1. Tìm kiếm nhiều từ

Trong danh sách 10.4, chúng tôi sử dụng một từ duy nhất (“Java”) làm tiêu chí tìm kiếm đối với trường tiêu đề. Chúng tôi có thể mở rộng tiêu chí này để phù hợp với việc tìm kiếm nhiều từ hoặc câu trong một trường duy nhất. Ví dụ, chúng tôi có thể tìm kiếm các từ “Java Complete Guide” trong trường tiêu đề, “Concurrency and Multithreading” trong trường tóm tắt, v.v. Thật vậy, việc tìm kiếm một chuỗi từ (như một câu bị ngắt quãng) phổ biến hơn là tìm kiếm một từ duy nhất. Truy vấn trong danh sách sau đây thực hiện chính xác điều đó.

**Liệt kê 10.5 Acuộc thi đấutruy vấn cho một tập hợp các từ**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tiêu đề": {

"query": "Hướng dẫn đầy đủ về Java"

}

}

},

"điểm nổi bật": {

"các trường": {

"tiêu đề": {}

}

}

}

Ở đây, chúng tôi dự định tìm kiếm một tiêu đề cụ thể (Java Complete Guide). Nghĩa là, chúng tôi muốn lấy một cuốn sách có tiêu đề Java Complete Guide, nếu có, và không trả về bất kỳ kết quả nào khác. Nhưng nếu bạn thực hiện truy vấn này, bạn có thể ngạc nhiên khi thấy nhiều tài liệu được trả về hơn là tài liệu khớp chính xác với truy vấn tìm kiếm.

* 1. ***Trận đấutruy vấn* 343**

Điều này là do Elasticsearch sử dụng toán tử OR Boolean theo mặc định cho truy vấn này, lấy tất cả các tài liệu khớp với bất kỳ từ tìm kiếm nào. Các từ được khớp riêng lẻ thay vì dưới dạng cụm từ: trong ví dụ của chúng tôi, Elasticsearch tìm kiếm "Java" và trả về các tài liệu có liên quan, tiếp theo là một tìm kiếm khác cho "Complete" để thêm các kết quả đó vào danh sách, v.v. Truy vấn trả về "Java", "Complete" hoặc "Guide", bao gồm các kết hợp của các từ, làm kết quả của nó. Có thể viết lại cùng một tìm kiếm (mặc dù toán tử OR là thừa) như danh sách tiếp theo hiển thị.

**Liệt kê 10.6 Chỉ địnhHOẶCtoán tử rõ ràng**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tiêu đề": {

"query": "Hướng dẫn đầy đủ về Java",

"toán tử": "HOẶC"

}

}

}

}

**Chỉ định toán tử OR một cách rõ ràng (mặc dù nó được thiết lập theo mặc định)**

Để thay đổi hành vi này nhằm tìm kiếm các tài liệu có cả ba từ trong tiêu đề, chúng ta cần bật toán tử AND.

**Liệt kê 10.7 Chỉ địnhVÀtoán tử rõ ràng**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tiêu đề": {

"query": "Hướng dẫn đầy đủ về Java",

"toán tử": "VÀ"

}

}

}

}

**Chỉ định rõ ràng một mệnh đề AND**

Truy vấn này cố gắng tìm một hoặc nhiều cuốn sách khớp với cả ba từ (tiêu đề phải bao gồm “Java” và “Complete” và “Guide”). Tuy nhiên, tập dữ liệu của chúng tôi không chứa một cuốn sách có tên là Java Complete Guide, do đó không có kết quả nào được trả về.

* + 1. Phù hợp với ít nhất một vài từ

Các toán tử OR và AND là các điều kiện đối lập. Điều kiện OR lấy một trong hai từ tìm kiếm và điều kiện AND lấy các tài liệu khớp với chính xác tất cả các từ. Nếu chúng ta muốn tìm các tài liệu khớp với ít nhất một vài từ trong tập hợp đã cho thì sao? Trong ví dụ trước, giả sử chúng ta muốn có ít nhất hai từ trong số ba từ để

**344 CPHẦN10*Tìm kiếm toàn văn***

cuộc thi đấu (nói, *Java* VàHướng dẫn). Cái này là Ở đâu cái tối thiểu\_phải\_phù hợp thuộc tính rất hữu ích.

Thuộc tính minimum\_should\_match chỉ ra số lượng từ tối thiểu cần sử dụng để khớp với các tài liệu. Danh sách tiếp theo cho thấy điều này trong hành động.

**Liệt kê 10.8 Phù hợp với ít nhất haitừ**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tiêu đề": {

"query": "Hướng dẫn đầy đủ về Java", "operator": "OR", "minimum\_should\_match": 2

}

}

}

}

**Đặt số lượng từ tối thiểu phải khớp**

Truy vấn này khớp với ít nhất hai từ (thuộc tính minimum\_should\_match được đặt thành 2) và lấy các tài liệu có sự kết hợp của hai trong ba từ đã cho. Toán tử OR là thừa ở đây vì nó được áp dụng theo mặc định.

**GHI CHÚ**Đặt giá trị thành 3 trong danh sách 10.8 cũng giống như thay đổi toán tử thành AND: tất cả các từ phải khớp nhau.

* + 1. Sửa lỗi đánh máy bằng cách sử dụng từ khóa fuzziness

Khi tìm kiếm thứ gì đó, đôi khi chúng ta nhập sai tiêu chí tìm kiếm (tất cả chúng ta đều từng gặp trường hợp này); ví dụ, thay vì tìm kiếm sách Java, chúng ta có thể đăng truy vấn với "Kava" làm tiêu chí tìm kiếm. Chúng ta biết mục đích của mình là tìm kiếm sách Java—và Elasticsearch cũng vậy.

Tính mờ làm thay đổi ký tự trong chuỗi đầu vào để nó giống với chuỗi có thể tồn tại trong chỉ mục. Nó sử dụng thuật toán khoảng cách Levenshtein để sửa lỗi chính tả. Chúng ta sẽ xem xét tính mờ sâu hơn trong phần 10.12, nhưng hãy xem cách sử dụng nó với các truy vấn khớp.

Truy vấn khớp cho phép chúng ta thêm tham số fuzziness để sửa lỗi chính tả. Chúng ta có thể đặt nó dưới dạng giá trị số, trong đó giá trị mong đợi là 0, 1 hoặc 2, nghĩa là không, một hoặc hai ký tự thay đổi (chèn, xóa, sửa đổi), tương ứng. Ngoài việc đặt các giá trị này, chúng ta có thể sử dụng cài đặt AUTO để cho công cụ xử lý các thay đổi. Danh sách sau đây cho thấy cách chúng ta có thể sử dụng fuzziness (với giá trị là 1) để sắp xếp lỗi đánh máy "Kava" của mình.

**Liệt kê 10.9 Sửa lỗi chính tả bằnghơi quăntham số**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

* 1. ***Cụm từ match\_phrasetruy vấn* 345**

"cuộc thi đấu": {

"title": { "query": "Kava", "fuzziness": 1

}

**Đặt độ mờ thành 1 sẽ thay thế một chữ cái bằng tất cả các kết hợp khác.**

}

}

}

Khi tìm kiếm chuỗi văn bản “Java Complete Guide”, chúng ta sử dụng một tập hợp các từ để tìm kiếm một cuốn sách (hoặc nhiều cuốn sách) và thường mong đợi các từ được xử lý riêng lẻ (như một tập hợp các từ tìm kiếm). Tuy nhiên, đôi khi chúng ta muốn tìm kiếm một cụm từ hoặc mộttence. Đó là lúc truy vấn match\_phrase trở nên hữu ích.

#### Truy vấn match\_phrase

Truy vấn cụm từ khớp (match\_phrase) tìm các tài liệu khớp chính xác với một cụm từ nhất định. Ý tưởng là tìm kiếm cụm từ (nhóm từ) trong một trường nhất định theo thứ tự nhất định. Ví dụ, nếu chúng ta đang tìm cụm từ “book for every Java pro-grammer” trong phần tóm tắt của một cuốn sách, các tài liệu sẽ được tìm kiếm theo thứ tự đó.

Trong phần trước (phần 10.5) về truy vấn khớp, chúng ta đã thấy rằng các từ có thể được tách riêng lẻ và tìm kiếm bằng toán tử AND/OR khi sử dụng truy vấn khớp. Truy vấn match\_phrase thì ngược lại. Nó trả về kết quả khớp chính xác với cụm từ tìm kiếm. Danh sách sau minh họa truy vấn match\_phrase đang hoạt động.

**Liệt kê 10.10match\_phrasetruy vấn trong hành động**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "cụm từ khớp": {

**Chỉ định một**

**truy vấn match\_phrase**

"synopsis": "sách dành cho mọi lập trình viên Java"

}

}

}

**Chỉ định cụm từ (nhóm từ) cần khớp**

Truy vấn match\_phrase mong đợi một cụm từ. Nó trả về chính xác một tài liệu vì chỉ có một tài liệu trong chỉ mục sách của chúng tôi có cụm từ đó trong trường tóm tắt.

Nếu chúng ta bỏ một hoặc hai từ trong cụm từ tìm kiếm thì sao? Ví dụ, giả sử chúng taxóa for hoặc every (hoặc cả hai) khỏi cụm từ “book for every Java programmer” và chạy lại truy vấn. Thật không may, truy vấn sẽ không trả về bất kỳ kết quả nào! Điều này là do match\_phrase mong đợi các từ trong cụm từ tìm kiếm khớp với cụm từ chính xác, từng từ một. Tìm kiếm “book Java programmer” không trả về kết quả nào. Nhưng có một cách khắc phục cho vấn đề này: sử dụng tham số slop.

Tham số slop cho phép chúng ta bỏ qua số lượng từ giữa các từ trong một cụm từ. Chúng ta có thể bỏ các từ ở giữa, nhưng chúng ta phải cho công cụ biết số lượng từ cần bỏ. Điều này được thực hiện bằng cách đặt giá trị cho tham số slop. Tham số slop

**346 CPHẦN10*Tìm kiếm toàn văn***

thuộc tính là một giá trị số nguyên biểu thị số lượng từ có thể bị bỏ qua trong mộtcụm từ trong quá trình tìm kiếm match\_phrase. Ví dụ, slop có giá trị 1 sẽ bỏ qua một từ bị thiếu trong cụm từ, slop có giá trị 2 sẽ bỏ qua hai từ, v.v. Giá trị mặc định của slop là 0, nghĩa là chúng ta không được tha thứ khi cung cấp cụm từ bị thiếu từ.

Quay trở lại ví dụ trước đó, chúng ta hãy bỏ từ for khỏi cụm từ đã cho để thay vì “book for every Java programming”, chúng ta tìm kiếm “book every Java pro-“ngữ pháp”. Vì chúng ta bỏ một từ duy nhất nên chúng ta cần đặt tham số slop thành 1. Chúng ta cũng cần mở rộng truy vấn bằng cách cung cấp hai tham số bổ sung trong đối tượng truy vấn và slop cho trường tóm tắt.

**Liệt kê 10.11 Amatch\_phrasetruy vấn vớibùn lầyđiều đó làm rơi một từ**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "cụm từ khớp": {

"Tóm tắt": {

**Chúng tôi mở rộng lĩnh vực này để**

**bao gồm một đối tượng với**

**thêm vàotham số. Cụm từ với một**

"query": "đặt lịch cho mọi lập trình viên Java",

**từ còn thiếu:vì**

"chảy": 1

}

}

}

}

**Đặt slop thành 1, nghĩa là truy vấn tìm kiếm các cụm từ có một từ bị thiếu**

Nếu chúng ta muốn sử dụng tham số slop, cả query và slop phải được cung cấp cùng với đối tượng của trường, như được hiển thị trong danh sách 10.11 (dạng dài của truy vấn). Vì slop được đặt thành 1, truy vấn sẽ khớp nếu thiếu một từ trong toàn bộ cụm từ trong trường tóm tắt. Truy vấn này trả về cuốn sách khớp với toàn bộ cụm từ của chúng ta. Điểm mấu chốt là truy vấn match\_phrase tìm kiếm một cụm từ chính xác, nhưng nếu chúng ta không chắc chắn về tìm kiếm của mình, chúng ta có thể sử dụng tham số slop để chỉ ra mức độ dễ tha thứ của truy vấn.

Truy vấn match\_phrase có một chút thay đổi: truy vấn match\_phrase\_prefix. Ngoài việc khớp với một cụm từ chính xác, chúng ta có thể khớp với từ cuối cùng dưới dạng tiền tố. Chúng ta hãy thảo luận về điều này tiếp theo với một ví dụ.

#### Truy vấn match\_phrase\_prefix

Truy vấn match\_phrase\_prefix giống như truy vấn match\_phrase ở chỗ, ngoài việc khớp chính xác cụm từ, truy vấn còn khớp với tất cả các từ, sử dụng từ cuối cùng trong cụm từ tìm kiếm làm tiền tố. Điều này dễ hiểu hơn thông qua một ví dụ. Danh sách sau đây tìm kiếm tiền tố “found” trong các thẻ, có thể áp dụng cho “foundation”, “found”, v.v.

**Liệt kê 10.12 Sử dụng mộttiền tố\_cụm\_phù\_hợptruy vấn**

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

* 1. ***Đa\_trận\_đấutruy vấn* 347**

**Chỉ định**

**tiền tố chotìm kiếm**

"match\_phrase\_prefix": { "thẻ": {

"query": "khái niệm và tìm thấy"

}

}

},

**Chỉ định truy vấn match\_phrase\_prefix**

"điểm nổi bật": {

"các trường": {

"thẻ": {}

}

}

}

Truy vấn này sẽ lấy tất cả các sách có thẻ khớp với “found”. Nó khớp với “foundational” trong chỉ mục sách của chúng tôi.

Tương tự như mộtmatch\_phrasetruy vấn, thứ tự từ cũng quan trọng trong mộtmatch\_cụm từ\_tiền tốtruy vấn. Và một lần nữa,bùn lầyđến để giải cứu. Ví dụ, để lấy lại những cuốn sách có cụm từ khái niệm và nền tảng trongthẻtrường, chúng ta có thể bỏ một từ bằng cách thêmbùn lầytừ khóa.

**Liệt kê 10.13 Sử dụng mộttiền tố\_cụm\_phù\_hợptruy vấn vớibùn lầy**

# Phù hợp với tiền tố cụm từ GET books/\_search

{

"truy vấn": { "tiền tố cụm từ khớp": {

"thẻ": {

"query": "khái niệm được tìm thấy",

**Cụm từ này thiếu một từ (“và”) và có tiền tố (“tìm thấy”).**

"lỏng lẻo":1

}

}

}

}

**Đặt slop thành 1 vì một từ bị loại khỏi cụm từ**

Đặt từ khóa slop thành 1 truy vấn sách có thẻ khái niệm và tìm thấy\*, nhưng nó bỏ qua từ và. Truy vấn sẽ trả về cuốn sách Lập trình Kotlin làm kết quả vì truy vấn khớp với cụm từ “Kotlin conceptual and foundational APIs” trong trường thẻ.

Cho đến nay, chúng tôi đã truy vấn tiêu chí tìm kiếm trên một trường duy nhất. Tuy nhiên, giả sử chúng tôi muốnđể tìm các từ “Phát triển phần mềm” trong các trường tiêu đề, tóm tắt và thẻ. Đó là những gì xảy ra khi chúng ta sử dụng truy vấn multi\_match, được thảo luận trong phần tiếp theo.

#### 10.8 Truy vấn multi\_match

Như tên gọi của nó, truy vấn đa khớp (multi\_match) tìm kiếm trên nhiều trường. Ví dụ, để tìm kiếm từ “Java” trong tiêu đề, tóm tắt vàtags fields, truy vấn multi\_match là câu trả lời. Danh sách sau đây hiển thị truy vấn tìm kiếm “Java” trên ba trường này.

**348 CPHẦN10*Tìm kiếm toàn văn***

**Liệt kê 10.14 Tìm kiếm nhiều trường bằng cách sử dụngnhiều\_trận\_hợp**

NHẬN sách/\_tìm kiếm



{

"\_source": sai, "truy vấn": {

"multi\_match": { "query": "Java", "fields": [

"tiêu đề", "tóm tắt", "thẻ"

**Ngăn chặn tài liệu nguồn xuất hiện trong kết quả**

**Chỉ định truy vấn multi\_match**

**Xác định tiêu chí tìm kiếm là từ “Java”**

**Tìm kiếm trên nhiều trường được cung cấp trong một mảng**

]

}

},

"điểm nổi bật": {

"các trường": { "tiêu đề": {},

"thẻ": {}

**Điểm nổi bật của các trận đấutrả về trong kết quả**

}

}

}

Truy vấn multi\_match mong đợi một mảngcủa các trường cùng với tiêu chí tìm kiếm. Chúng tôi nhận được kết quả tổng hợp bằng cách kết hợp tất cả các kết quả cho từng trường.

* + 1. Các lĩnh vực tốt nhất

Bạn có thể tự hỏi mức độ liên quan của tài liệu là bao nhiêu khi chúng ta tìm kiếm nhiều trường. Các trường có nhiều từ khớp nhau hơn sẽ có điểm cao hơn. Nếu chúng ta tìm kiếm "Java Collections" trên nhiều trường, một trường (ví dụ, tóm tắt) có hai từ khớp nhau sẽ có mức độ liên quan cao hơn một trường có một (hoặc không) từ khớp. Trong trường hợp này, tài liệu có trường tóm tắt này sẽ có điểm liên quan cao hơn.

Các trường khớp với tất cả các tiêu chí tìm kiếm được gọi là các trường tốt nhất. Trong phần trướcVí dụ, giả sử synopsis chứa cả hai từ, Java và Collections, chúng ta có thể nói rằng synopsis là trường tốt nhất. Multi-match sử dụng kiểu best\_fields khi chạy truy vấn. Kiểu này là mặc định cho truy vấn multi\_match. Tất nhiên, còn có các trường khác, như chúng ta sẽ thấy sau đây.

Hãy viết lại truy vấn từ danh sách 10.14. Lần này, thay vì để Elasticsearchsử dụng cài đặt mặc định (kiểu best\_fields), chúng tôi ghi đè cụ thể trường kiểu.

**Liệt kê 10.15 Chỉ địnhtrường tốt nhấtgõ rõ ràng**

NHẬN sách/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"nhiều\_trùng\_hợp": {

"query": "Mẫu thiết kế", "type": "best\_fields",

**Đặt loại truy vấn multi\_match thành best\_fields**

***10.8 Đa\_trậntruy vấn* 349**

"các trường": ["tiêu đề","tóm tắt"]

}

},

"điểm nổi bật": {

"các trường": { "các thẻ": {},

"tiêu đề": {}

**Ngăn chặn nguồnnhưng cho thấy những điểm nổi bật**

}

}

}

Chúng tôi truy vấn “Mẫu thiết kế” trêntiêu đềVàTóm tắtcác trường. Lần này, chúng tôi hướng dẫn rõ ràngnhiều\_trận\_hợptruy vấn để sử dụngtrường tốt nhấtkiểu.

**GHI CHÚ**Kiểu mặc định cho mộtnhiều\_trận\_hợptruy vấn làtrường tốt nhất. Cáctrường tốt nhấtThuật toán xếp hạng trường có nhiều từ nhất cao hơn trường có ít từ nhất.

Nếu bạn nhìn vào phản hồi và điểm số (xem đoạn mã sau), bạn sẽ thấyrằng cuốn sách Head First Design Patterns có số điểm là 6,9938974 so với cuốn Head First Object-Oriented Analysis Design có số điểm là 2,9220228:

"lượt truy cập" : [{

"\_index" : "sách",

"\_id" : "10",

**"\_điểm" : 6.9938974,**

"điểm nổi bật" : {

"tiêu đề" : [

"Thiết kế mẫu <em>Head First</em>"

]

}

},

{

"\_index" : "sách",

"\_id" : "8",

**"\_score" : 2.9220228,**

"điểm nổi bật" : {

"tiêu đề" : [

"Thiết kế phân tích hướng đối tượng đầu tiên"

]

}

}

...]

Có nhiều loại truy vấn đa khớp khác, bao gồm cross\_fields, most\_fields, phrase và phrase\_prefix. Chúng ta có thể sử dụng tham số type để đặt loại truy vấn nhằm tìm kiếm kết quả khớp nhất giữa nhiều trường. Tuy nhiên, chúng ta sẽ không đi sâu vào tất cả các loại này ở đây; hãy tham khảo tài liệu của Elasticsearch để biết thêm thông tin.

Elasticsearch thực hiện truy vấn multi\_match như thế nào? Nó được viết lại bằng cách sử dụng truy vấn dis-junction max (dis\_max) ẩn. Chúng ta sẽ thảo luận về loại truy vấn này sau.

**350 CPHẦN10*Tìm kiếm toàn văn***

* + 1. Truy vấn dis\_max

Trong phần trước, chúng ta đã xem xét truy vấn multi\_match, tìm kiếm tiêu chí trong nhiều trường. Để thực hiện loại truy vấn này ở chế độ nền, Elasticsearch viết lại truy vấn multi\_match bằng truy vấn disjunction max (dis\_max). Truy vấn dis\_max chia mỗi trường thành một truy vấn match riêng biệt, như danh sách sau đây cho thấy.

**Liệt kê 10.16 Phân tách max (dis\_max) truy vấn trongsử dụng**

NHẬN sách/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"dis\_max": {

"thắc mắc": [

**Chỉ định loại truy vấn dis\_max**

**Xác định tập hợp các truy vấn để đưa vào khối truy vấn dis\_max**

{"match": {"title": "Mẫu thiết kế"}},

{"match": {"synopsis": "Mẫu thiết kế"}}]

}

}

}

**Chỉ định một truy vấn khớp**

Nhiều trường được chia thành hai truy vấn khớp trong truy vấn dis\_max. Truy vấn trả về các tài liệu có \_score liên quan cao cho từng trường.

**GHI CHÚ**Truy vấn dis\_max được phân loại là truy vấn hợp chất: truy vấn bao gồm các truy vấn khác. Chúng tôi thảo luận về truy vấn hợp chất trong chương 11.

Trong một số trường hợp, điểm liên quan của các trường trong truy vấn multi\_match là như nhau. Trong trường hợp đó, điểm sẽ hòa. Để phá vỡ thế hòa, chúng tôi sử dụng một tiebreaker.

* + 1. Máy cắt dây buộc

Điểm liên quan dựa trên điểm của một lĩnh vực duy nhất, nhưng nếu điểm số bằng nhau, chúng ta có thểchỉ định tie\_breaker để giải quyết vấn đề hòa. Khi chúng ta sử dụng tie\_breaker, Elasticsearch tính toán điểm tổng thể hơi khác một chút, như chúng ta sẽ thấy ngay sau đây; trước tiên, hãy cùng xem một ví dụ.

Danh sách sau đây truy vấn hai từ so với hai trường: tiêu đề và thẻ. Tuy nhiên, mã cũng thêm tham số tie\_breaker.

**Liệt kê 10.17 Anhiều\_trận\_hợptruy vấn với một tiebreaker**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "multi\_match": {

**Chỉ định một**

**nhiều\_trận\_hợptruy vấn Các truy vấn cho**

**Đặt ra sự phá vỡ thế bế tắc**

}

"query": "Mẫu thiết kế", "type": "best\_fields", "fields": ["title","tags"], "tie\_breaker": 0.9

}

}

**"Thiết kế Các mẫu”**

**Đặt loại truy vấn thành best\_fields**



**Xác định tập hợp các trường để tìm kiếm**

* 1. ***Đa\_trận\_đấutruy vấn* 351**

Khi chúng ta tìm kiếm “Design Patterns” bằng cách sử dụng loại best\_fields và chỉ định nhiều trường (tiêu đề và tóm tắt), chúng ta có thể cung cấp giá trị tie\_breaker để vượt qua bất kỳ điểm hòa nào khi có số điểm bằng nhau. Khi chúng ta cung cấp một tiebreaker, tổng điểm được tính như sau:

Tổng điểm = điểm của trường phù hợp nhất + điểm của các trường phù hợp khác \* tie\_breaker

Trước đó, chúng tôi đã làm việc với mộtdis\_maxtruy vấn. Elasticsearch chuyển đổi tất cảnhiều\_trận\_hợptruy vấn đếndis\_maxtruy vấn. Ví dụ,nhiều\_trận\_hợptruy vấn từ danh sách 10.17 có thể được viết lại thànhdis\_maxtruy vấn.

**Liệt kê 10.18dis\_maxtruy vấn với một tiebreaker**

NHẬN sách/\_tìm kiếm

{

"\_source": sai, "truy vấn": {

"dis\_max": {

"thắc mắc": [

{"match": {"title": "Mẫu thiết kế"}},

{"match": {"synopsis": "Mẫu thiết kế"}}], "tie\_breaker": 0,5

} **Người phá vỡ thế bế tắc**

},

"điểm nổi bật": {

"các trường": { "tiêu đề": {},

"Tóm tắt": {},

"thẻ": {}

}

}

}

Truy vấn multi\_match hiện được viết dưới dạng truy vấn dis\_max. Đó chính xác là những gì Elastic-search thực hiện đằng sau hậu trường.

Khi chúng ta tìm kiếm nhiều trường, đôi khi chúng ta muốn đưa thêm trọng số cho một trường cụ thể (ví dụ, việc tìm kiếm từ khóa trong tiêu đề có liên quan hơncùng một từ tìm kiếm xuất hiện trong một bản tóm tắt dài hoặc trường thẻ). Làm thế nào để chúng ta cho Elasticsearch biết để cung cấp thêm trọng số cho trường tiêu đề? Chúng ta có thể tăng cường các truy vấn riêng lẻ, như đã thảo luận trong phần sau.

* + 1. Tăng cường các lĩnh vực riêng lẻ

Các trang web và ứng dụng thường cung cấp thanh tìm kiếm để người dùng tìm kiếm sản phẩm, sách, đánh giá, v.v. Khi người dùng nhập một vài từ, điều đó không có nghĩa là họ chỉ muốn tìm kiếm những từ đó trong một trường cụ thể. Ví dụ, khi chúng tôi tìm kiếm "sách C#" trên Amazon, chúng tôi không yêu cầu Amazon chỉ tìm kiếm trong một danh mục cụ thể, chẳng hạn như tiêu đề hoặc tóm tắt. Chúng tôi chỉ cần nhập chuỗi vào hộp văn bản và để Amazon tính toán kết quả. Đó là những gì chúng tôi có thể làm bằng cách sử dụng các trường tăng cường riêng lẻ!

**352 CPHẦN10*Tìm kiếm toàn văn***

Trong truy vấn multi\_match, chúng ta có thể đẩy lên (boost) các tiêu chí cho một trường cụ thể. Giả sử khi tìm kiếm “C# Guide”, chúng ta quyết định rằng việc tìm từ trong tiêu đề quan trọng hơn việc tìm từ trong thẻ. Trong trường hợp này, chúng ta có thể boost trường quan trọng của trường bằng cách sử dụng dấu mũ và một số: title^2, ví dụ. Danh sách sau đây hiển thị toàn bộ truy vấn cho tình huống này.

**Liệt kê 10.19 Tăng điểm cho một trường trongnhiều\_trận\_hợptruy vấn**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "multi\_match": {

"query": "Hướng dẫn C#", "fields": ["title^2", "tags"]

}

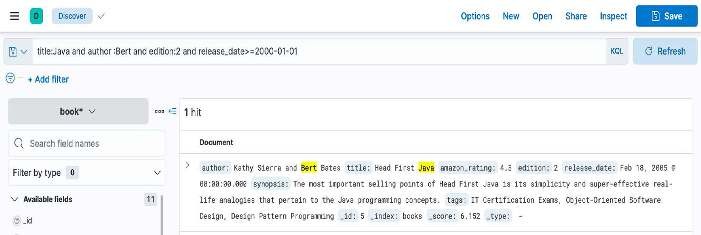
**Tăng gấp đôi tầm quan trọng của trường tiêu đề**

}

}

Trong danh sách này, chúng tôi nhân đôi tầm quan trọng của trường tiêu đề. Vì vậy, nếu văn bản “C# Guide” được tìm thấy trong trường tiêu đề, tài liệu đó có điểm cao hơn tài liệu có văn bản trong trường thẻ.

Tiếp theo, hãy xem truy vấn query\_string. Loại truy vấn này giúp chúng ta xây dựng tìm kiếm URL yêu cầu, mô phỏng Ngôn ngữ truy vấn Kibana (KQL) theo định dạng DSL truy vấn. Chúng ta xem xét nhu cầu về truy vấn query\_string và làm việc với nó trong phần sau.



**Chuỗi truy vấn và KQL**

Trong chương 8, chúng ta đã xem xét phương pháp tìm kiếm URI (một phương pháp truy vấn tìm kiếm ngoài Query DSL). Chúng ta đã tạo một yêu cầu bằng cách chuyểntìm kiếmtruy vấn và các tham số của nó tới URL, thay vì trong một thân yêu cầu. Chúng tôi cũng thấy rằng mặc dù phương pháp URIrất đơn giản, nhưng dễ xảy ra lỗi khi độ phức tạp của tiêu chí truy vấn tăng lên.

Trên tab Discover của Kibana, chúng ta thường sử dụng KQL để tạo tiêu chí tìm kiếm bằng các toán tử. Ví dụ, để tìm kiếm “sách Java phiên bản thứ 2 do Bert viết, phát hành sau năm 2010”, truy vấn tương đương được viết trong hộp KQL của tab Discover như sau:

tiêu đề:Java và tác giả:Bert và phiên bản:2 và ngày phát hành>=2000-01-01

* 1. ***Chuỗi truy vấntruy vấn* 353**

**Tab Khám phá của Kibana hiển thị truy vấn KQL**

Elastic giới thiệu KQL để truy vấn nhật ký và số liệu trong chỉ mục Elasticsearchthông qua Kibana. KQL là một ngôn ngữ linh hoạt và trực quan hỗ trợ nhiều khả năng tìm kiếm, bao gồm tìm kiếm cấp trường, hoạt động logic và truy vấn phức tạp. Chúng ta có thể tìm kiếm bằng ký tự đại diện và, như được hiển thị trong truy vấn ví dụ, kết hợp nhiều truy vấn tìm kiếm với các toán tử logic nhưVÀ,HOẶC, VàKHÔNG. KQL cũng cung cấp chức năng tự động hoàn thành và tô sáng cú pháp, giúp việc xây dựng và hiểu các truy vấn dễ dàng hơn. Nguyên lý đằng sau KQL sử dụng API yêu cầu URI. Thật tiện lợi khi sử dụng các truy vấn kiểu URI mà không phải lo lắng về việc xây dựng truy vấn ở chế độ DSL truy vấn—nếu chỉ có một phương pháp cung cấp cho chúng ta những điều tốt nhất của cả hai thế giới.

Tin tốt là chúng ta có thể đạt được chức năng URI tương tự bằng cách sử dụng truy vấn đặc biệt được gọi làchuỗi truy vấn. Kiểu truy vấn này cho phép chúng ta định nghĩa một truy vấn bằng cách sử dụng toán tử logicngười dùng trong một cơ thể yêu cầu.

#### 10.9 Truy vấn query\_string

Kiểu query\_string cho phép chúng ta xây dựng truy vấn bằng các toán tử như AND hoặc OR, cũng như các toán tử logic như > (lớn hơn), <= (nhỏ hơn hoặc bằng), \* (chứa trong), v.v. Điều này dễ hiểu với một ví dụ, vì vậy chúng ta hãy bắt đầu ngay vào mã.

Trong thanh bên ở phần trước, chúng tôi đã lấy sách của Bert bằng cách sử dụng truy vấn dài trên tab Khám phá của Kibana:tiêu đề:Java và tác giả:Bert và phiên bản:2 vàngày phát hành>=2000-01-01.Chúng ta có thể đạt được điều này bằng cách viết mộtchuỗi truy vấntruy vấn.

**Liệt kê 10.20 Tạo chuỗi truy vấn với các toán tử**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

**Chỉ định**

**chuỗi truy vấn truy vấn**

"query": "tác giả:Bert VÀ phiên bản:2 VÀ ngày phát hành>=2000-01-01"

} **Cung cấp tìm kiếm**

} **tiêu chí trong truy vấn**

}

Truy vấn query\_string mong đợi một tham số truy vấn mà chúng ta cung cấp tiêu chí. Truy vấn được xây dựng dưới dạng cặp tên-giá trị: trong ví dụ này, tác giả là trường và Bert là giá trị. Xem mã, hãy chú ý những điều sau:

* Truy vấn tìm kiếm được xây dựng bằng cú pháp Query DSL (LẤYyêu cầu có nội dung).
* Tiêu chí tìm kiếm được viết để nối các trường bằng các toán tử.

Truy vấn này đơn giản như viết một câu hỏi bằng tiếng Anh thông thường. Chúng ta có thể tạo các tiêu chí phức tạp bằng cách sử dụng các toán tử này (bằng tiếng Anh thông thường) và cung cấp chúng cho công cụ để có được kết quả cho chúng ta.

**354 CPHẦN10*Tìm kiếm toàn văn***

Đôi khi chúng tôi không biết người dùng muốn tìm kiếm những lĩnh vực nào; họ có thể muốntruy vấn để tập trung vào trường tiêu đề, trường tóm tắt hoặc thẻ hoặc tất cả các trường đó. Trong phần tiếp theo, chúng ta sẽ xem xét các cách chỉ định trường.

10.9.1 Các trường trong truy vấn query\_string

Trong hộp tìm kiếm thông thường, người dùng không cần phải chỉ định trường khi tìm kiếm thứ gì đó. Ví dụ, hãy xem truy vấn sau.

**Liệt kê 10.21 Chuỗi truy vấn không có trường nào được chỉ định**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"query": "Mẫu"

}

**Từ khóa tìm kiếm truy vấn (không đề cập đến các trường)**

},

"điểm nổi bật": {

"các trường": { "tiêu đề": {},

"Tóm tắt": {},

"thẻ": {}

**Làm nổi bật các phản hồi**

}

}

}

Chúng tôi muốn tìm kiếm từ khóa “Patterns” trong truy vấn trước đó. Hãy nhớ rằng, truy vấn không yêu cầu chúng tôi tìm kiếm bất kỳ trường nào. Đây là truy vấn chung cần được thực hiện trên tất cả các trường. Phản hồi cho thấy một số kết quả được tô sáng trên một trường khác nhau cho từng tài liệu:

"điểm nổi bật" : {

"synopsis": ["Mẫu thiết kế <em>Head First</em> là một trong những ..."], "title": ["Mẫu thiết kế <em>Head First</em>"]

},

...

"điểm nổi bật" : {

"tóm tắt": [ "tạo .. sử dụng các mẫu ứng dụng hiện đại"]

},

...

Thay vì để công cụ tìm kiếm truy vấn tất cả các trường có sẵn, chúng ta có thể hỗ trợ Elasticsearch bằng cách cung cấp các trường để chạy tìm kiếm. Danh sách tiếp theo cho biết cách thực hiện việc này.

**Liệt kê 10.22 Chỉ định các trường một cách rõ ràng trongchuỗi truy vấntruy vấn**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"query": "Mẫu",

**Tiêu chí truy vấn không có trường nào được đề cập**

* 1. ***Truy vấn query\_string***

"các trường": ["tiêu đề","tóm tắt","thẻ"]

}

}

}

**Tuyên bố rõ ràng các trường là một mảng chuỗi**

**355**

Ở đây, trong một mảng trong tham số fields, chúng ta chỉ định rõ ràng các trường mà tiêu chí này sẽ được tìm kiếm. Nếu chúng ta không chắc chắn về các trường khi xây dựng truy vấn, chúng ta có thể sử dụng một tham số khác, default\_field.

**Liệt kê 10.23 Chuỗi truy vấn có trường mặc định**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"query": "Mẫu", "default\_field": "title"

}

**Mặc địnhkhai báo trường**

}

}

Nếu một trường không được đề cập trong truy vấn, tìm kiếm sẽ được thực hiện theo tiêu đề

cánh đồng. Đó là bởi vìtiêu đềtrường được khai báotrường mặc định.

* + 1. Toán tử mặc định

Trong danh sách 10.23, chúng tôi đã tìm kiếm một từ duy nhất, “Patterns”. Nếu chúng tôi mở rộng tìm kiếm đó để bao gồm một từ bổ sung, chẳng hạn như “Design”, chúng tôi có thể nhận được nhiều cuốn sách (hai cuốn sách, với bộ dữ liệu hiện tại) thay vì kết quả đúng: Head First Design Patterns. Lý do là Elasticsearch sử dụng toán tử OR theo mặc định khi tìm kiếm. Do đó, nótìm những cuốn sách có cả hai từ “Thiết kế” hoặc “Mẫu” trong trường tiêu đề.

Nếu đây không phải là ý định của chúng ta (ví dụ, chúng ta muốn lấy một cuốn sách có cụm từ chính xác Design Patterns trong tiêu đề), chúng ta có thể sử dụng toán tử AND. Truy vấn chuỗi query\_ sau có một tham số bổ sung, default\_operator, trong đó chúng ta có thể đặt toán tử thành AND.

**Liệt kê 10.24 Chuỗi truy vấn vớiVÀngười điều hành**

NHẬN sách/\_tìm kiếm

{

}

"truy vấn": { "chuỗi\_truy vấn": {

"query": "Mẫu thiết kế", "default\_field": "title", "default\_operator": "AND"

}

}

**Thay đổi toán tử từ OR thành AND**

Cái nàychuỗi truy vấntruy vấn được khai báo vớiVÀtoán tử. Do đó, chúng tôi mong đợi các Mẫu thiết kế được coi là một từ duy nhất.

**356 CPHẦN10*Tìm kiếm toàn văn***

* + 1. Truy vấn query\_string với một cụm từ

Nếu bạn đang thắc mắc liệu có hỗ trợ tìm kiếm cụm từ bằng chuỗi query\_ hay không, thì thực sự là có. Chúng ta có thể viết lại mã trong danh sách 10.24 bằng cách sử dụng cụm từ thay vì thay đổi toán tử. Điều duy nhất chúng ta cần lưu ý là cụm từ phải được đặt trong dấu ngoặc kép. Điều đó có nghĩa là dấu ngoặc kép tương ứng với cụm từ phảiđã thoát: ví dụ, "query": "\"Design Patterns\"". Truy vấn tiếp theo sẽ tìm kiếm một cụm từ.

**Liệt kê 10.25 Achuỗi truy vấntruy vấn với một cụm từ**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"query": "\"làm cho mã tốt hơn\"", "default\_field": "synopsis"

**Dấu ngoặc kép quanh câu khiến nó trở thành một cụm từ truy vấn.**

}

}

}

Mã này tìm kiếm cụm từ "làm cho mã tốt hơn" trong trường tóm tắt và lấy cuốn sách Effective Java. Chúng ta cũng có thể sử dụng tham số slop (đã thảo luận trước đó trong chương này ở phần 10.5.2 và trong chương 8) nếu chúng ta thiếu một hoặc hai từ trong cụm từ. Ví dụ, danh sách sau đây cho thấy cách tham số phrase\_slop cho phép một từ bị thiếu trong cụm từ (the bị loại khỏi cụm từ) và vẫn nhận được kết quả thành công.

**Liệt kê 10.26 Achuỗi truy vấntruy vấn với một cụm từ vàcụm từ\_slop**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"query": "\"làm cho mã tốt hơn\"", "default\_field": "tóm tắt",

**Cụm từ có một từ bị xóa**

"cụm từ\_lỗi": 1

}

}

}

**Đặt phrase\_slop thành 1 để tôn trọngcụm từ với một mất tích**

Truy vấn thiếu một từ, nhưng thiết lập phrase\_slop bỏ qua lỗi thiếu sót này và chúng ta có được kết quả mong muốn.

Khi xây dựng dịch vụ tìm kiếm, chúng ta phải cân nhắc hỗ trợ lỗi chính tả. Các ứng dụng phải xử lý những lỗi này một cách khéo léo và thay vì trả về kết quả không chính xác hoặc không có kết quả, hãy xác định vấn đề chính tả để cải thiện và điều chỉnh kết quả. Điều này cải thiện trải nghiệm tìm kiếm của người dùng. Elasticsearch cung cấp hỗ trợ xử lý lỗi chính tả thông qua các truy vấn mờ.

***10.11 Chuỗi đơn giảntruy vấn* 357**

#### Truy vấn mờ

Chúng ta có thể yêu cầu Elasticsearch bỏ qua lỗi chính tả bằng cách sử dụng các truy vấn mờ với truy vấn chuỗi query\_. Tất cả những gì chúng ta cần làm là thêm hậu tố tiêu chí truy vấn bằng toán tử dấu ngã (~). Điều này được hiểu rõ nhất thông qua một ví dụ.

**Liệt kê 10.27 Một truy vấn chuỗi mờ**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"query": "Pattenrs~", "default\_field": "title"

}

}

}

**Viết sai chính tả “Pattenrs” là từ khóa tìm kiếm**

Bằng cách đặt hậu tố với toán tử ~, chúng tôi yêu cầu công cụ xem xét truy vấn mờ. Theo mặc định, khoảng cách chỉnh sửa là 2 được sử dụng khi làm việc với các truy vấn mờ. Khoảng cách chỉnh sửa là số lượng đột biến cần thiết để chuyển đổi một chuỗi thành một chuỗi khác. Ví dụ, các từ cat và cap chỉ khác nhau một chữ cái. Do đó, chúng tôi yêu cầu khoảng cách chỉnh sửa là 1 để chuyển đổi cat thành cap.

Trong chương 9, chúng ta đã tìm hiểu về các truy vấn mờ trong ngữ cảnh cấp thuật ngữ, sử dụng thuật toán khoảng cách Levenshtein. Một loại thuật toán khoảng cách chỉnh sửa khác là thuật toán khoảng cách Damerau–Levenshtein, được sử dụng để hỗ trợ các truy vấn mờ trong ngữ cảnh toàn văn. Thuật toán này hỗ trợ chèn, xóa hoặc thay thế tối đa hai ký tự cũng như chuyển vị các ký tự liền kề.

**GHI CHÚ**Thuật toán khoảng cách Levenshtein xác định số lượng đột biến tối thiểu cần thiết để một chuỗi được chuyển đổi thành một chuỗi khác. Các đột biến này bao gồm các phép chèn, xóa và thay thế. Thuật toán khoảng cách Damerau–Levenshtein tiến thêm một bước nữa. Ngoài việc có tất cả các đột biến được Levenshtein xác định, thuật toán Damerau-Levenshtein còn xem xét việc chuyển vị các ký tự liền kề (ví dụ: TB > BT > BAT).

Theo mặc định, khoảng cách chỉnh sửa trong truy vấn query\_string là 2, nhưng chúng ta có thể giảm khoảng cách này nếu cần bằng cách đặt thành 1 sau dấu ngã: "Pattenrs~1". Trong hai phần tiếp theo, chúng ta sẽ xem xét một số truy vấn đơn giản hơn.

#### Truy vấn chuỗi đơn giản

Truy vấn query\_string nghiêm ngặt về cú pháp và các lỗi trong đầu vào không được tha thứ. Ví dụ, truy vấn sau đây đưa ra lỗi vì đầu vào có vấn đề về phân tích cú pháp (có chủ đích—chúng tôi đã thêm một trích dẫn vào tiêu chí đầu vào).

**358 CPHẦN10*Tìm kiếm toàn văn***

**Liệt kê 10.28 Achuỗi truy vấntruy vấn với một ký tự trích dẫn bất hợp pháp**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy vấn": {

"truy vấn": "tiêu đề:Java\""

}

**Truy vấn có lỗi cú pháp (không có trích dẫn tương ứng)**

}

}

Truy vấn này không được phân tích cú pháp. Elasticsearch đưa ra một ngoại lệ nêu rằng cú pháp đã bị vi phạm ("lý do": "Không thể phân tích cú pháp 'title:Java\"': Lỗi từ vựng ở dòng 1, cột 12. Đã gặp phải: <EOF> sau: \"\""). Việc ném ngoại lệ phân tích cú pháp JSON này cho người dùng sẽ xác thực cú pháp chặt chẽ hơn chochuỗi truy vấntruy vấn. Tuy nhiên, nếu chúng ta muốn Elasticsearch bỏ qua các lỗi cú pháp và tiếp tục công việc, thì có một giải pháp thay thế:chuỗi\_truy\_cập\_đơn\_giảntruy vấn, được thảo luận trong phần sau.

#### Truy vấn simple\_query\_string

Như tên gọi của nó,chuỗi\_truy\_cập\_đơn\_giảntruy vấn là một biến thể củachuỗi truy vấntruy vấn với cú pháp đơn giản, hạn chế. Chúng ta có thể sử dụng các toán tử như+,-,|,\*, Và

~ để xây dựng truy vấn. Ví dụ, tìm kiếm "Java + Cay" sẽ cho ra một cuốn sách Java do Cay viết, như danh sách tiếp theo hiển thị.

**Liệt kê 10.29 Truy vấn simple\_query\_string**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy\_cập\_đơn\_giản": {

"truy vấn": "Java + Cay"

}

**Chỉ định loại truy vấn**

**như simple\_query\_string**

**Tìm kiếm truy vấn bằng toán tử AND**

}

}

Toán tử + trong truy vấn cho phép truy vấn tìm kiếm “Java” và “Cay” trên tất cả các trường. Chúng ta có thể chỉ định các trường nếu chúng ta muốn kiểm tra một tập hợp các trường thay vì tất cả các trường bằng cách thiết lập mảng các trường. Bảng 10.1 mô tả các toán tử chúng ta có thể sử dụng trong

chuỗi\_truy\_cập\_đơn\_giản.

**Bảng 10.1 Các nhà điều hành cho mộtchuỗi\_truy\_cập\_đơn\_giảntruy vấn**

|  |  |
| --- | --- |
| **Người điều hành** | **Sự miêu tả** |
| | | HOẶC |
| + | VÀ |
| - | KHÔNG |
| ~ | Truy vấn mờ |

***Bản tóm tắt* 359**

**Bảng 10.1 Các nhà điều hành cho mộtchuỗi\_truy\_cập\_đơn\_giảntruy vấn (tiếp theo)**

|  |  |
| --- | --- |
| **Người điều hành** | **Sự miêu tả** |
| \*  " | Truy vấn tiền tố  Truy vấn cụm từ |

Không giống như truy vấn query\_string, truy vấn simple\_query\_string không trả về lỗi nếu có lỗi cú pháp trong tiêu chí đầu vào. Nó có cách tiếp cận nhẹ nhàng hơn là không trả về bất kỳ thứ gì nếu có lỗi cú pháp trong truy vấn, như danh sách tiếp theo cho thấy.

**Liệt kê 10.30 Không có vấn đề gì với saicú pháp**

NHẬN sách/\_tìm kiếm

{

"truy vấn": { "chuỗi\_truy\_cập\_đơn\_giản": {

"truy vấn": "tiêu đề:Java\""

}

**MỘT chuỗi\_truy\_cập\_đơn\_giản**

**truy vấn**

**Truy vấn với cú pháp không đúng**

}

}

Mặc dù chúng tôi đưa ra cùng một truy vấn với cú pháp không chính xác (một dấu ngoặc kép ở cuối), không có lỗi nào được trả về cho người dùng ngoại trừ việc không có tài liệu nào được trả về.Truy vấn simple\_query\_string hữu ích trong những tình huống như vậy.

Và thế là xong! Chương này nói về các truy vấn toàn văn: các truy vấn trên dữ liệu phi cấu trúc. Trong chương 11, chúng ta sẽ xem xét chi tiết các truy vấn hợp chất. Các truy vấn hợp chất là các truy vấn tìm kiếm nâng cao bao gồm các truy vấn lá như truy vấn cấp thuật ngữ và truy vấn toàn văn.

#### Bản tóm tắt

* Elasticsearch tập trung vào việc tìm kiếm dữ liệu phi cấu trúc bằng truy vấn toàn văn. Truy vấn toàn văn mang lại sự liên quan, nghĩa là các tài liệu được khớp và trả về có điểm liên quan dương.
* Elasticsearch cung cấp một\_tìm kiếmAPI phục vụ mục đích truy vấn.
* Một số loạicuộc thi đấutruy vấn hoạt động cho nhiều trường hợp sử dụng khác nhau khi tìm kiếm toàn văn có liên quan. Truy vấn phổ biến nhất làcuộc thi đấutruy vấn.
* Cáccuộc thi đấutruy vấn tìm kiếm trên các trường văn bản để tìm kiếm tiêu chí và chấm điểm tài liệu bằng thuật toán liên quan.
* Phù hợp với tất cả (khớp\_tất\_cả) truy vấn tìm kiếm trên tất cả các chỉ mục và không yêu cầu nội dung.
* Để tìm kiếm một cụm từ, chúng tôi sử dụngmatch\_phrasetruy vấn hoặc biến thể của nó,match\_cụm từ\_tiền tố. Cả hai loại truy vấn đều cho phép chúng ta tìm kiếm các từ cụ thể theo thứ tự xác định. Ngoài ra, chúng ta có thể sử dụngcụm từ\_sloptham số nếu cụm từ bị thiếu từ.

**360 CPHẦN10*Tìm kiếm toàn văn***

* Việc tìm kiếm tiêu chí người dùng trên nhiều trường được kích hoạt bằng cách sử dụngnhiều\_trận đấutruy vấn.
* Một chuỗi truy vấn (chuỗi truy vấn) truy vấn sử dụng các toán tử logic nhưVÀ,HOẶC, VàKHÔNG. Tuy nhiên,chuỗi truy vấntruy vấn có cú pháp nghiêm ngặt, do đó chúng tôi nhận được ngoại lệ nếu cú ​​pháp đầu vào không chính xác.
* Nếu chúng ta cần Elasticsearch ít nghiêm ngặt hơn về cú pháp chuỗi truy vấn, thì thay vì sử dụngchuỗi truy vấntruy vấn, chúng ta có thể sử dụng mộtchuỗi\_truy\_cập\_đơn\_giảntruy vấn. Với kiểu truy vấn đó, tất cả các lỗi cú pháp đều được công cụ loại bỏ.

*Truy vấn hợp chất*

***Chương này bao gồm***

* Làm việc với các truy vấn hợp chất
* Truy vấn tìm kiếm Boolean
* Truy vấn điểm số liên tục
* Tăng cường truy vấn
* Truy vấn tối đa phân tách
* Truy vấn điểm số chức năng

Trong hai chương trước, chúng ta đã xem xét các truy vấn cấp độ thuật ngữ và toàn văn. Chúng ta đã thảo luận về việc tìm kiếm dữ liệu có cấu trúc và không có cấu trúc bằng các truy vấn, một số tạo ra điểm liên quan và một số khác hoạt động trong bối cảnh bộ lọc, trong đó điểm không liên quan. Hầu hết các truy vấn cho phép thiết lập tiêu chí tìm kiếm đơn giản và làm việc trên một tập hợp các trường giới hạn, chẳng hạn như tìm sách do một tác giả viết hoặc tìm kiếm sách bán chạy nhất.

Ngoài việc cung cấp các truy vấn cho các tiêu chí phức tạp, đôi khi chúng tôi cần tăng điểm dựa trên các tiêu chí nhất định trong khi đồng thời phủ nhận điểm cho các kết quả trùng khớp tiêu cực (ví dụ: tất cả các cuốn sách được ra mắt trong một chương trình đào tạo có thể nhận được sự thúc đẩy tích cực trong khi đồng thời, các cuốn sách đắt tiền bị loại bỏ

**361**

**362 CPHẦN11*Truy vấn hợp chất***

[bị phủ định]). Hoặc có thể chúng ta muốn đặt điểm dựa trên các yêu cầu tùy chỉnh thay vì sử dụng các thuật toán liên quan tích hợp sẵn của Elasticsearch.

Các truy vấn lá riêng lẻ mà chúng tôi đã làm việc cho đến nay bị hạn chế ở chỗ chúng có thể tìm kiếm dựa trên một hoặc nhiều tiêu chí nhưng không dựa trên các yêu cầu phức tạp hơn (ví dụ: tìm kiếm sách do một tác giả cụ thể viết, được xuất bản trong một số ngày nhất định, được liệt kê là sách bán chạy nhất hoặc được đánh giá 4,5 trên 5, với số trang cụ thể). Các truy vấn nâng cao như vậy yêu cầu khả năng truy vấn tìm kiếm nâng cao. Chúng tôi sẽ xem xét các truy vấn hợp chất này trong các chương này.

*Truy vấn hợp chất*là các cấu trúc tìm kiếm nâng cao để truy vấn các tiêu chí tìm kiếm phức tạp trong Elasticsearch. Chúng được tạo thành từ các truy vấn lá riêng lẻ được gói trong các mệnh đề điều kiện và các cấu trúc khác để cung cấp các khả năng như cho phép người dùng đặt điểm tùy chỉnh bằng các hàm được xác định trước, tăng cường các kết quả tìm kiếm tích cực trong khi loại bỏ các mệnh đề tiêu cực, phát triển điểm bằng các tập lệnh, v.v. Chúng cho phép chúng ta sử dụng các truy vấn lá riêng lẻ để phát triển các truy vấn nâng cao hoàn chỉnh thuộc nhiều loại khác nhau.

Trong chương này, chúng tôi sẽ xem xét các yêu cầu mà các truy vấn hợp chất đáp ứng, cũng như ngữ nghĩa và cách sử dụng của chúng. Chúng tôi xem xét các truy vấn Boolean, trong đó nhiều truy vấn lá được đúc trong một vài mệnh đề có điều kiện để thiết kế một truy vấn tìm kiếm nâng cao và chúng tôi sử dụngcác mệnh đề như must, must\_not, should và filter để sắp xếp các truy vấn lá thành một truy vấn hợp chất. Chúng ta xem xét việc tăng cường các truy vấn để nâng cao điểm của truy vấn khi có sự trùng khớp dương tính đồng thời giảm điểm cho các phủ định. Sau đó, chúng ta tìm hiểu về một truy vấn chấm điểm tĩnh được xác định trước có tên là constant\_score, giúp đặt điểm tĩnh cho các kết quả trả về.

Chúng tôi cũng chạy qua các truy vấn điểm số hàm để giúp thiết lập các thuật toán chấm điểm tùy chỉnh do người dùng xác định bằng cách sử dụng một tập hợp các hàm. Chúng tôi xem xét cơ chế thiết lập điểm số bằng cách sử dụng các tập lệnh và trọng số dựa trên các giá trị trường khác có trong tài liệu, cũng như các số ngẫu nhiên. Nhưng trước tiên, hãy lấy một số dữ liệu mẫu.

#### Dữ liệu sản phẩm mẫu

Trong các ví dụ của chương này, chúng tôi làm việc với một tập dữ liệu các sản phẩm điện và điện tử như tivi (TV), máy tính xách tay, điện thoại di động, tủ lạnh (tủ lạnh), v.v. Dữ liệu sản phẩm có sẵn trên GitHub (<http://mng.bz/Rxwa>) và trên trang web của cuốn sách ([https://www.manning.com/books/elasticsearch-in-action-second-edTôi](https://www.manning.com/books/elasticsearch-in-action-second-edition) [sự kiện](https://www.manning.com/books/elasticsearch-in-action-second-edition)). Trong phần này, chúng tôi sẽ xem xét định nghĩa và quy trình lập chỉ mục.

* + 1. Sơ đồ sản phẩm

Bước đầu tiên trong việc phát triển chỉ mục sản phẩm là tạo lược đồ dữ liệu xác định các trường và kiểu dữ liệu của chúng. Danh sách sau đây hiển thị lược đồ tổng quan (lược đồ đầy đủ có sẵn trong các tệp của sách).

**Liệt kê 11.1 Định nghĩacác sản phẩmsơ đồ**

Sản phẩm PUT

{

"ánh xạ": {

* 1. ***Sản phẩm mẫudữ liệu* 363**

"thuộc tính": { "thương hiệu": {

"loại": "văn bản"

},

"màu sắc": {

"loại": "văn bản"

},

"energy\_rating": { "type": "văn bản"

},

...

"user\_ratings": { "type": "gấp đôi"

},

"giá": {

"loại": "đôi"

}

}

}

}

Không có bất ngờ nào trong định nghĩa sản phẩm ngoại trừ một vài thuộc tính(giá và user\_ratings) được định nghĩa là double nhưng phần còn lại được khai báo là trường văn bản. Hầu hết các trường được khai báo là nhiều kiểu dữ liệu (ví dụ: văn bản và từ khóa) để phù hợp với việc làm việc với các truy vấn cấp thuật ngữ trên dữ liệu (ví dụ: xếp hạng năng lượng hoặc màu sắc).

Việc lập bản đồ trong danh sách 11.1 tạo ra một chỉ mục sản phẩm trống với lược đồ liên quan cho các sản phẩm điện tử thương mại điện tử mà chúng ta sắp lập chỉ mục. Tiếp theo, chúng ta lập chỉ mục một tập hợp các sản phẩm mẫu.

* + 1. Sản phẩm lập chỉ mục

Bây giờ khi lược đồ đã sẵn sàng, hãy lập chỉ mục cho tập dữ liệu sản phẩm của chúng ta cho Elasticsearch, có sẵn trong các tệp của cuốn sách. Sao chép nội dung của products.txt và dán chúng vào Kibana. Chúng ta sử dụng API \_bulk để lập chỉ mục cho dữ liệu này. Đoạn mã này hiển thị một mẫu dữ liệu:

ĐẶT \_số lượng lớn

{"index":{"\_index":"sản phẩm","\_id":"1"}}

{"sản phẩm": "TV", "thương hiệu": "Samsung", "model": "UE75TU7020", "kích thước": "75",

"resolution": "4k", "type": "smart tv", "price": 799, "colour": "silver", "energy\_rating": "A+", "overview": "Hãy an tọa để tận hưởng một trải nghiệm tuyệt vời..", "user\_ratings": 4.5, "images": ""}

{"index":{"\_index":"sản phẩm","\_id":"2"}}

{"sản phẩm": "TV", "thương hiệu": "Samsung", "model": "QE65Q700TA", "kích thước": "65",

"resolution": "8k", "type": "QLED", "price": 1799, "colour": "black", "energy\_rating": "A+", "overview": "Chiếc TV 65 inch tuyệt vời này ..", "user\_ratings": 5, "images": ""}

{"index":{"\_index":"sản phẩm","\_id":"3"}}

...

Bây giờ chúng ta đã lập chỉ mục cho tập dữ liệu sản phẩm, hãy bắt đầu bằng cách thảo luận về nhu cầu sử dụng truy vấn hợp chất và lý do cũng như cách chúng giúp xây dựng các truy vấn nâng cao.

**364 CPHẦN11*Truy vấn hợp chất***

#### Truy vấn hợp chất

Chúng tôi đã làm việc với các truy vấn lá trong hai chương cuối (truy vấn cấp độ thuật ngữ và truy vấn toàn văn): các truy vấn hoạt động trên các trường đơn lẻ (riêng lẻ). Nếu yêu cầu của chúng tôi là tìm những cuốn sách bán chạy nhất trong một giai đoạn, chúng tôi có thể sử dụng truy vấn lá để lấy các kết quả đó. Truy vấn lá giúp tìm câu trả lời cho các câu hỏi đơn giản mà không cần hỗ trợ các mệnh đề điều kiện. Tuy nhiên, thế giới thực hiếm khi liên quan đến các yêu cầu truy vấn đơn giản.

Hầu hết các yêu cầu đều yêu cầu phát triển các truy vấn phức tạp với nhiều mệnh đề và điều kiện. Ví dụ, một truy vấn phức tạp có thể bao gồm việc tìm kiếm những cuốn sách bán chạy nhất do một tác giả cụ thể viết và xuất bản trong một khoảng thời gian cụ thể hoặc một phiên bản cụ thể; hoặc trả về tất cả các cuốn sách được phân loại theo các khu vực địa lý khác nhau ngoại trừ một quốc gia cụ thể, được sắp xếp theo tổng doanh thu cao nhất.

Đây là nơi các truy vấn hợp chất tỏa sáng: chúng giúp phát triển các truy vấn tìm kiếm phức tạp bằng cách kết hợp một hoặc nhiều truy vấn lá. May mắn thay, chúng ta có thể sử dụng Query DSL (đã thảo luậntrong chương 8) để viết truy vấn hợp chất: chúng ta sử dụng cùng một điểm cuối \_search với một thân yêu cầu bao gồm một đối tượng truy vấn. Hình 11.1 cho thấy cú pháp cho một truy vấn hợp chất.

**Loại truy vấn: chọn loại truy vấn hợp chất cần thiết.**

NHẬN <index>/\_search

{

"query": { "bool|constant\_score|boosting|dis\_max|function\_score": {

// truy vấn tìm kiếm

}

}

}

**Truy vấn có tiêu chí phức tạp sẽ được đưa vào đây.**

**Có năm loại truy vấn để lựa chọn:**

**bool,điểm số hằng số,tăng cường,dis\_max, Vàđiểm\_hàm.**

**Hình 11.1 Cú pháp truy vấn hợp chất**

Cú pháp cơ bản cho truy vấn hợp chất không khác gì các truy vấn khác. Tuy nhiên, phần thân của đối tượng truy vấn được tạo thành từ các thành phần khác nhau, tùy thuộc vào loại truy vấn hợp chất mà chúng ta muốn sử dụng.

Elasticsearch cung cấp năm truy vấn như vậy cho các yêu cầu tìm kiếm khác nhau: Boolean (bool), điểm hằng số, điểm hàm, tăng cường và disjunction max. Bảng 11.1 mô tả ngắn gọn năm truy vấn hợp chất này. Ví dụ, nếu yêu cầu của chúng ta là phát triển một truy vấn nâng cao sử dụng các mệnh đề điều kiện, chúng ta có thể sử dụng truy vấn Boolean bao gồm nhiều truy vấn lá sử dụng AND, OR và các điều kiện khác. Tương tự, nếu yêu cầu là đặt điểm tĩnh trên tất cả các kết quả, thì truy vấn điểm hằng số là thứ chúng ta cần. Chúng tôi áp dụng năm loại truy vấn này trong các trường hợp sử dụng khác nhau trong chương này.

* 1. ***Boolean (bool)truy vấn* 365**

**Bảng 11.1 Các loại truy vấn hợp chất**

|  |  |
| --- | --- |
| **Truy vấn hợp chất** | **Sự miêu tả** |
| Truy vấn Boolean (bool)  Truy vấn điểm số không đổi (constant\_score)  Truy vấn điểm số hàm (function\_score)  Truy vấn tăng cường (boosting)  Phân ly tối đa (dis\_max)truy vấn | Một sự kết hợp của các mệnh đề điều kiện bao bọc các truy vấn lá riêng lẻ (thuật ngữ và toàn văn). Hoạt động tương tự nhưVÀ,HOẶC, VàKHÔNGngười vận hành.  Ví dụ: sản phẩm = TV VÀ màu sắc = bạc KHÔNG đánh giá < 4,5 VÀ thương hiệu =Samsung hoặc LG  Gói truy vấn bộ lọc để đặt điểm số không đổi cho kết quả. Cũng giúp tăng điểm số.  Ví dụ: Tìm kiếm tất cả các TV có xếp hạng của người dùng cao hơn 5 nhưng đặt điểm cố định là 5 cho mỗi kết quả bất kể điểm tính toán của công cụ tìm kiếm là bao nhiêu.  Các hàm do người dùng xác định để gán điểm tùy chỉnh cho các tài liệu kết quả  Ví dụ: Tìm kiếm sản phẩm và nếu sản phẩm đó là của LG và là TV, hãy tăng điểm lên ba (thông qua tập lệnh hoặc hàm trọng số).  Tăng điểm cho các trận đấu tích cực trong khi phủ nhận điểm cho các trận đấu không phù hợp  Ví dụ: Lấy tất cả các TV nhưng giảm điểm của những TV đắt tiền.  Bao bọc nhiều truy vấn để tìm kiếm nhiều từ trong nhiều trường (tương tự như truy vấn multi\_match)  Ví dụ: Tìm kiếm TV thông minh trong hai trường (ví dụ: tổng quan và  mô tả) và trả về kết quả phù hợp nhất. |

Trong số các loại này, truy vấn bool là truy vấn hợp chất được sử dụng phổ biến nhất do tính linh hoạt và hỗ trợ nhiều mệnh đề điều kiện. Chúng ta có nhiều điều cần đề cập khi thảo luận về truy vấn bool, vì vậy nó xứng đáng có một phần riêng. Nhưng trước khi tiếp tục với phần còn lại của chương, chúng ta phải thiết lập hộp thử nghiệm của mình để thử nghiệm với các truy vấn hợp chất.

#### 11.3 Truy vấn Boolean (bool)

Truy vấn Boolean (bool) là truy vấn hợp chất phổ biến và linh hoạt nhất để tạo ra các tiêu chí phức tạp cho việc tìm kiếm dữ liệu. Như tên gọi của nó, nó kết hợp các mệnh đề Boolean, mỗi mệnh đề có một truy vấn lá được tạo thành từ các truy vấn cấp thuật ngữ hoặc toàn văn. Mỗi mệnh đề cómột sự xuất hiện được gõ của các mệnh đề must, must\_not, should hoặc filter, được giải thích ngắn gọn trong bảng 11.2.

**Bảng 11.2 Các mệnh đề Boolean**

|  |  |
| --- | --- |
| **Điều khoản** | **Sự miêu tả** |
| phải  phải\_không | Truy vấn AND trong đó tất cả các tài liệu phải khớp với tiêu chí truy vấn Ví dụ: Lấy TV (sản phẩm = TV) nằm trong phạm vi giá cụ thể  Một truy vấn KHÔNG trong đó không có tài liệu nào khớp với tiêu chí truy vấn  Ví dụ: Lấy TV (sản phẩm = TV) nằm trong phạm vi giá cụ thể nhưng  ngoại trừ một số trường hợp, chẳng hạn như không phải là một màu nhất định |

**366 CPHẦN11*Truy vấn hợp chất***

**Bảng 11.2 Mệnh đề Boolean (tiếp theo)**

|  |  |
| --- | --- |
| **Điều khoản** | **Sự miêu tả** |
| nên  lọc | Truy vấn OR trong đó một trong các tài liệu phải khớp với tiêu chí truy vấn Ví dụ: Tìm kiếm tủ lạnh không đóng tuyết hoặc có mức năng lượng được đánh giá trên mức C.  Một truy vấn lọc trong đó các tài liệu phải phù hợp với tiêu chí truy vấn (tương tự nhưcáiphảimệnh đề), nhưng mệnh đề bộ lọc không tính điểm các trận đấu  Ví dụ: Lấy TV (sản phẩm = TV) nằm trong phạm vi giá cụ thể (nhưngđiểm của các tài liệu trả về sẽ là 0) |

Một truy vấn hợp thành bao gồm các mệnh đề này có thể chứa nhiều truy vấn lá hoặc thậm chí là các truy vấn hợp thành bổ sung. Chúng ta có thể tạo các truy vấn tìm kiếm nâng cao, phức tạp bằng cách kết hợp các truy vấn lá và truy vấn hợp thành. Hãy thảo luận về cú pháp và cấu trúc truy vấn bool.

11.3.1 Cấu trúc truy vấn bool

Như đã đề cập trước đó, truy vấn bool là sự kết hợp của các mệnh đề Boolean tạo rađầu ra thống nhất. Hình 11.2 minh họa cấu trúc cơ bản của truy vấn bool với các mệnh đề rỗng.

**Boolean (bool) khai báo truy vấn**

NHẬN <index>/\_search

{

"truy vấn": {

"bool": { "phải": [{}],

"must\_not": [{}],

"nên": [{}],

"lọc": [{}]

**Lá riêng lẻ**

**(hoặc hợp chất tiếp theo)truy vấn là được bao gồm trong các điều khoản.**

**Mệnh đề điều kiện**

}

}

}

**Hình 11.2 Cú pháp của một mẫu Boolean (bool) truy vấn với bốn mệnh đề điều kiện**

Như bạn có thể thấy, một booltruy vấn được cấu hình với một tập hợp các điều kiện được ghi lại trongmệnh đề. Một truy vấn bool có thể chấp nhận ít nhất một trong các truy vấn được nhúng trong một mệnh đề. Mỗi mệnh đề sau đó có thể lưu trữ một hoặc nhiều truy vấn lá hoặc truy vấn hợp thành một mảng các truy vấn. Như đoạn mã sau đây cho thấy, chúng ta có thể cung cấp nhiều truy vấn cấp thuật ngữ và toàn văn (được hiển thị bằng chữ in đậm) bên trong bất kỳ mệnh đề nào (in nghiêng):

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"bool": {

***11.3 Truy vấn Boolean (bool)***

"phải": [

{ "phù hợp": {"FIELD": "TEXT"}},

{ "term": {"FIELD": {"value": "VALUE"}}}

],

"must\_not": [

{"bool": { "phải": [{}]}}}

**367**

]

"nên": [

{ "phạm vi": { "LĨNH VỰC": {"gte": 10,"lte": 20}}},

{ "điều khoản": { "TRƯỜNG": [ "GIÁ TRỊ1", "GIÁ TRỊ2" ]}}

]

}

}

}

Ở đây, chúng ta có ba mệnh đề và mỗi mệnh đề chứa các truy vấn lá như match, term,range, v.v. Chúng ta cũng có một mệnh đề hợp chất trong mệnh đề must\_not, nơi chúng ta có thể mở rộng thêm tiêu chí của mình bằng cách sử dụng cùng một tập hợp các mệnh đề. Các truy vấn riêng lẻ này được nối trong các mệnh đề cho phép chúng ta viết một truy vấn tìm kiếm đáp ứng các yêu cầu truy vấn nâng cao. Mặc dù bạn có thể hiểu truy vấn trước đó theo góc độ lý thuyết, nhưng trừ khi bạn viết và thực thi các truy vấn, bạn không thể đánh giá hết tiềm năng của chúng. Hãy cùng xem xét truy vấn bool từ đầu, xây dựng truy vấn từng mệnh đề một

và phát triển nó trong quá trình thực hiện. Chúng ta bắt đầu cuộc chạy nước rút với mệnh đề bắt buộc.

* + 1. Mệnh đề phải

Tiêu chí mà chúng ta khai báo trong mệnh đề must của truy vấn bool sẽ cho kết quả dương khi các truy vấn được xác định trong khối (clause) thỏa mãn tiêu chí. Nghĩa là, đầu ra chứa tất cả các tài liệu khớp với các điều kiện trong mệnh đề must.

Hãy cùng khám phá một truy vấn đơn giản để bắt đầu. Giả sử yêu cầu của chúng ta là tìm tất cảTV trong chỉ mục sản phẩm. Đối với điều đó, chúng tôi viết truy vấn bool với mệnh đề must. Vì chúng tôi chỉ tìm kiếm TV, chúng tôi có thể đặt tiêu chí tìm kiếm trong truy vấn khớp, khớp các sản phẩm là TV. Danh sách sau đây cung cấp mã.

**Liệt kê 11.2 Abooltruy vấn củaphảimệnh đề với mộtcuộc thi đấutruy vấn**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool":{ 

"phải": [

{

**Một booltruy vấn Phảiđiều khoản**

**Tiêu chí tìm kiếm**

}

]

}

}

"cuộc thi đấu": {

"sản phẩm": "TV"

}

**Một truy vấn khớp lệnh truy vấn theo tác giả**

}

**368 CPHẦN11*Truy vấn hợp chất***

Chúng ta hãyphân tích truy vấn này. Boolkhai báo bên trong đối tượng truy vấn chỉ ra rằngđây là truy vấn bool. Đối tượng bool sau đó được bao bọc bằng mệnh đề must với tiêu chí để khớp với từ tìm kiếm đã cho, TV. Khi chúng tôi thực hiện truy vấn này, một vài TV được trả về, như mong đợi (đầu ra được bỏ qua ở đây để ngắn gọn).

**Truy vấn khớp là truy vấn bool**

Cáccuộc thi đấutruy vấn mà chúng tôi đã làm việc cho đến nay là một loại truy vấn Boolean. Ví dụ,booltruy vấn trong danh sách 11.2 có thể được viết lại thànhcuộc thi đấu(toàn văn) truy vấn để lấyTV. Truy vấn lá sau trả về cùng TV được trả về bởi truy vấn lá trước đóbooltruy vấn:

NHẬN sách/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tác giả": "Joshua"

}

}

}

Bạn có thể có xu hướng sử dụng mộtcuộc thi đấutruy vấn cho một tiêu chí đơn giản (duy nhất), nhưng than ôi, có nhiều sự phức tạp hơn trong thế giới thực; vì vậy, bạn có thể phải dựa vàobooltruy vấn.

* + 1. Tăng cường mệnh đề phải

Lấy TV không thú vị lắm, phải không? Hãy làm cho truy vấn của chúng ta thú vị hơn một chút. Ngoài việc lấy TV, hãy thêm một điều kiện: chỉ lấy TV có giá trị nằm trong một phạm vi giá nhất định. Chúng ta cần kết hợp hai truy vấn để đạt được mục tiêu mong muốn.

Yêu cầu này yêu cầu chúng ta sử dụng mệnh đề must với hai truy vấn khớp: một để khớp trên trường sản phẩm và một để khớp trên giá. Hãy nhớ rằng, mệnh đề must chấp nhận một mảng các truy vấn cấp độ lá. Trong khi một truy vấn khớp là đủ để tìm kiếm trên loại sản phẩm, chúng ta có thể thêm một truy vấn phạm vi vào truy vấn bool để lấy tất cả các TV trong một phạm vi giá nhất định.

**Danh sách 11.3 Tìm TV trong phạm vi giá**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [

{

**Phải truy vấn với hai truy vấn lá riêng lẻ**

"cuộc thi đấu": {

"sản phẩm": "TV"

**Một truy vấn khớp để tìm TV**

}

},

{

"phạm vi": {

**Một truy vấn phạm vi với phạm vi giá**

***11.3 Truy vấn Boolean (bool)***

"giá": {

"gte": 700,

"lte": 800

**369**

}

}

}

]

}

}

}

Ở đây, chúng tôi tạo hai truy vấn lá với AND có điều kiện giữa truy vấn khớp và truy vấn phạm vi. Các truy vấn như sau: tìm kiếm TV và lấy những TV có phạm vi giá đã chỉ định. Kết quả là một TV, vì tập dữ liệu của chúng tôi chỉ chứa một TV có giá trong phạm vi ($799).

Tất nhiên, chúng ta có thể thêm tiêu chí khác vào truy vấn. Ví dụ, truy vấn trong danh sách tiếp theo tìm kiếm tất cả TV màu bạc hoặc đen có độ phân giải 4K.

**Liệt kê 11.4 Ba truy vấn lá được gói trong mộtphảiđiều khoản**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [

{

**Phải truy vấn với ba truy vấn lá**

"cuộc thi đấu": {

"sản phẩm": "TV"

}

},

{

"term": { "resolution": "4k"

**Truy vấn lá phù hợp để tìm kiếm TV**

**Thuật ngữ truy vấn để tìm TV có độ phân giải 4K**

}

},

{

"điều khoản": {

"màu sắc": [

"bạc", "đen"

**Các điều khoản truy vấn để lấy TV màu bạc hoặc đen**

]

}

}

]

}

}

}

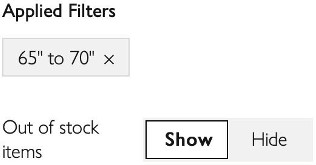
Như bạn có thể tưởng tượng, chúng ta có thể kết hợp nhiều truy vấn toàn văn bản và cấp thuật ngữ (hoặc các truy vấn lá khác) để phát triển giải pháp tìm kiếm cho các tiêu chí phức tạp và phức tạp bằng cách sử dụng truy vấn bool—chúng ta vừa mới bắt đầu. Chúng ta có thể xây dựng nhiều hơn nữatruy vấn nâng cao sử dụng các mệnh đề khác; chúng ta sẽ thảo luận về mệnh đề must\_not tiếp theo.

**370 CPHẦN11*Truy vấn hợp chất***

* + 1. Mệnh đề must\_not

Ngược lại với must, như bạn có thể đoán, là mệnh đề truy vấn must\_not. Ví dụ, trên một trang web mua sắm, chúng ta có thể muốn tìm kiếm các sản phẩm có thông tin chi tiết cụ thể, yêu cầu nhà bán lẻ bỏ qua các sản phẩm không có sẵn. Hình 11.3 cho thấy một ví dụ (từ một nhà bán lẻ ở Anh, John Lewis).

**Chuyển đổi để xóa**



**sản phẩm hết hàng**

**Hình 11.3 Ẩn hết hàngcác mục trong quá trình tìm kiếm**

Công cụ tìm kiếm ẩn tất cả các mặt hàng hết hàng trước khi hiển thị kết quả tìm kiếm. Điều nàylà loại chức năng mà mệnh đề must\_not có thể đáp ứng.

Giống như mệnh đề must, must\_not chấp nhận một mảng các truy vấn lá để xây dựng tiêu chí tìm kiếm nâng cao. Mục đích duy nhất của truy vấn là lọc ra các kết quả khớp không đáp ứng tiêu chí đã chỉ định. Cách tốt nhất để hiểu điều này là thông qua một ví dụ. Truy vấn trong danh sách sau đây tìm kiếm tất cả các TV không phải là một thương hiệu cụ thể (trong trường hợp này là Samsung hoặc Philips).

**Liệt kê 11.5 Lấy TV bằng cách sử dụngphải\_khôngđể loại trừ một số thương hiệu nhất định**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"must\_not": [

{



[

**Một mệnh đề không được phéplưu trữ một truy vấn thuật ngữ**

"điều khoản": { "thương hiệu.từ khóa":

"Samsung", "Philips"

**Các thuật ngữ truy vấn tìm kiếm các thương hiệu cụ thể**

]

}

}

]

}

}

}

Danh sách này hiển thị truy vấn bool được cung cấp với mệnh đề must\_not chứa truy vấn terms. Lúc đầu, như mong đợi, truy vấn terms sẽ lấy các sản phẩm là Samsung hoặc Philips. Vì truy vấn terms này tồn tại trong mệnh đề must\_not, nên hiệu ứng là phủ định những gì

***11.3 Boolean (bool)truy vấn* 371**

truy vấn thực hiện. Nghĩa là, kết quả “không được khớp với những gì khớp với truy vấn trong“mệnh đề”. Do đó, vì truy vấn terms được gói trong mệnh đề must\_not nên kết quả ngược lại: nó sẽ tìm nạp tất cả các sản phẩm không do Samsung và Philips sản xuất.

Vấn đề với truy vấn này là nó lấy tất cả các sản phẩm (TV, tủ lạnh, màn hình, v.v.). Nhưng hãy nhớ rằng, yêu cầu của chúng ta là chỉ lấy TV không do Samsung hoặc Philips sản xuất. Chúng ta nên sửa đổi truy vấn để lấy TV, bỏ qua các thương hiệu cụ thể đó.

Nếu nhưbạn đang nghĩ chúng ta nên thêm mệnh đề must vào mệnh đề must\_not trong danh sách11.5, bạn đúng. Chúng ta có thể tạo truy vấn thuật ngữ được gói trong mệnh đề must để lấy tất cả TV và sau đó xóa (lọc) các thương hiệu cụ thể khỏi kết quả bằng cách sử dụng must\_not. Hãy cập nhật truy vấn để phản ánh điều này.

**Liệt kê 11.6 Lọc ra một số thương hiệu nhất định**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"must\_not": [

{

"điều khoản": {

**Mệnh đề must\_not bỏ qua một tập hợp các thương hiệu**

"thương hiệu.từ khóa": [ "Philips", "Samsung"

]

}

}

],

"phải": [

{

"cuộc thi đấu": {

"sản phẩm": "TV"

**Một điều khoản bắt buộc phải tìm kiếm (phù hợp) TV**

}

}

]

}

}

}

Chúng ta có hai mệnh đề truy vấn: must và must\_not, cả hai đều được gói gọn trong một truy vấn hợp chất bool duy nhất. Bây giờ chúng ta đang tìm kiếm các TV không phải là hai thương hiệu cụ thể. Chúng ta có thể thêm nhiều hơn vào mệnh đề must\_not không? Có; nó có thể được cải thiện bằng nhiều truy vấn lá, như chúng ta thảo luận trong phần sau.

* + 1. Cải thiện mệnh đề must\_not

Tương tự như cách chúng tôi tăng cường truy vấn cho các mệnh đề must (phần 11.3.3), việc cung cấp nhiều tiêu chí truy vấn bên trong mệnh đề must\_not là điều dễ hiểu. Ví dụ, ngoài việc tìm kiếm các sản phẩm không do Philips và Samsung sản xuất, chúng tôi

**372 CPHẦN11*Truy vấn hợp chất***

chỉ có thể truy vấn các TV có xếp hạng bốn sao trở lên (truy vấn must\_not sử dụng

truy vấn phạm vi lọc ra các TV có xếp hạng của người dùng dưới 4,0).

**Liệt kê 11.7 Nâng caophải\_khôngtruy vấn**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"must\_not": [

{



[

**Mệnh đề must\_not với hai truy vấn riêng biệt**

"điều khoản": { "thương hiệu.từ khóa":

"Philips", "Samsung"

**Truy vấn các thuật ngữ khớp với các giá trị đã cho trong trường thương hiệu**

]

}

},

{

"phạm vi": { "xếp hạng người dùng": {

"lte": 4.0

**Truy vấn phạm vi tìm kiếm các TV được đánh giá thấp hơn hoặc bằng 4 sao**

}

}

}

], **Mệnh đề phải với**

"phải": [

{

**một vài câu hỏi lá**

"cuộc thi đấu": {

"sản phẩm": "TV"

}

},

{

"thuật ngữ": { "giải pháp": {

"giá trị": "4k"

}

}

},

{

"phạm vi": {

"giá": {

"gte": 500,

"lte": 700

}

}

}

]

}

}

}

Mặc dù truy vấn trong danh sách 11.7 khá dài dòng, nhưng việc phân tích nó sẽ giúp hiểu được bản chất của nó.Truy vấn bool này bao gồm hai mệnh đề: must và must\_not. Mệnh đề must tìm kiếm tất cả các TV có độ phân giải 4K có giá từ 500 đến 700 đô la. Danh sách này là

***11.3 Boolean (bool)truy vấn* 373**

sau đó đưa vào mệnh đề must\_not, mệnh đề này có hai truy vấn lá hoạt động trên danh sách TV do mệnh đề must tạo ra. Truy vấn đầu tiên lọc ra tất cả TV Philips hoặc Samsung (giữ lại TV của tất cả các thương hiệu khác). Truy vấn lá thứ hai lọc thêm danh sách này bằng truy vấn phạm vi loại bỏ tất cả TV có xếp hạng dưới 4.0.

**GHI CHÚ**Mệnh đề must\_not không ảnh hưởng đến điểm liên quan của kết quả trả về. Điều này là do các truy vấn must\_not được thực hiện trong một ngữ cảnh bộ lọc. Các truy vấn được thực hiện trong ngữ cảnh bộ lọc không tạo ra điểm; chúng đưa ra kết quả nhị phân (có hoặc không). Do đó, điểm được tạo ra bởi các mệnh đề khác, chẳng hạn như mustvà nên, không được sửa đổi bởi các truy vấn được khai báo trong mệnh đề must\_not.

Cho đến nay, chúng ta đã xem xét các mệnh đề must và must\_not để xây dựng các truy vấn hợp chất của mình. Các truy vấn từ các mệnh đề này nằm ở cả hai phía của thang đo: mệnh đề must cụ thể về các tiêu chí khớp chính xác, trong khi mệnh đề must\_not thì ngược lại—nó cung cấp các kết quả không khớp với bất kỳ tiêu chí nào.

Ví dụ của chúng tôi đã lấy tất cả các TV khớp với một điều kiện nhất định trong khi loại bỏ một số thương hiệu nhất định. Giả sử chúng tôi muốn lấy kết quả cho một phần khớp: ví dụ, lấy TV lớn hơn 85 inch hoặc được sản xuất bởi một công ty cụ thể. Loại nàytruy vấn được hỗ trợ bởi mệnh đề thứ ba của truy vấn bool: mệnh đề should.

* + 1. Mệnh đề nên

Nói một cách đơn giản, mệnh đề should là mệnh đề OR đánh giá tìm kiếm dựa trên điều kiện OR (trong khi mệnh đề must dựa trên toán tử AND). Ví dụ, hãy xem truy vấn sau.

**Liệt kê 11.8 Sử dụng mộtnêntruy vấn với một vài tiêu chí để lấy TV**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["sản phẩm","thương hiệu", "tổng quan","giá"], "truy vấn": {

"bool": {

"nên": [

{

**Mệnh đề should với**

**hai truy vấn riêng lẻ**

"phạm vi": {

"giá": {

"gte": 500,

"lte": 1000

**Truy vấn để khớp các sản phẩm trong phạm vi giá cụ thể**

}

}

},

{

"match\_phrase\_prefix": { "tổng quan": "4K Ultra HD"

}

}

]

}

}

}

**Truy vấn để tìm kiếm sản phẩm phù hợpmột cụm từ trong trường tổng quan**

**374 CPHẦN11*Truy vấn hợp chất***

Mệnh đề should được tạo thành từ hai truy vấn tìm kiếm sản phẩm trong phạm vi giá

$500 đến $1.000 hoặc các sản phẩm cócụm từ “4K Ultra HD” trong trường tổng quan. Chúng tôi nhận được nhiều kết quả hơn bạn mong đợi (hãy nhớ điều kiện OR):

{

...

"\_score" : 12.059638, "\_source" : {

"tổng quan": ".. Màn hình 4K Ultra HD ...", "sản phẩm": "TV",

"giá" : 799, "thương hiệu" : "Samsung"

}

},

{

...

"\_score" : 11.199882, "\_source" : {

"tổng quan" : ".. 4K Ultra HD ...",, "sản phẩm" : "TV",

"giá" : 639, "thương hiệu" : "Panasonic"

}

},

{

...

"\_score" : 10.471219, "\_source" : {

"tổng quan": ".. Màn hình 4K Ultra HD..", "sản phẩm": "TV",

"giá" : 1599,

"thương hiệu" : "LG"

}

}

...

Kết quả bao gồm các sản phẩm không nằm trong phạm vi giá đã chỉ định (ví dụ: sản phẩm thứ ba trong kết quả trả về có giá là 1.599 đô la (cao hơn nhiều so với mức chúng tôi yêu cầu); tuy nhiên, đây là kết quả khớp vì tiêu chí thứ hai, 4K Ultra HD, khớp. Chỉ số nàychỉ ra rằng mệnh đề should hoạt động theo điều kiện OR.

Có nhiều hơn là sử dụng điều kiện OR trên các truy vấn khi thực hiện tìm kiếmcho mệnh đề should. Mặc dù chúng ta chỉ chạy truy vấn should, nhưng thường thì nó được kết hợp với các mệnh đề khác như must và must\_not. Ưu điểm của việc sử dụng mệnh đề should với mệnh đề must là kết quả khớp với truy vấn trong mệnh đề should sẽ được tăng điểm. Chúng ta hãy thảo luận chi tiết về điều này trong phần tiếp theo, sử dụng một ví dụ.

**BBỎ ĐIỂM SỐ VỚI MỘT MỆNH ĐỀ NÊN**

Truy vấn trong danh sách 11.8 có một mệnh đề should, trả về kết quả dương bằng cách sử dụngĐiều kiện OR. Khi sử dụng với mệnh đề must, mệnh đề should sẽ tăng thêm trọng số cho điểm liên quan. Giả sử chúng ta đưa ra một mệnh đề must khớp với TV LG.

***11.3 Boolean (bool)truy vấn* 375**

**Liệt kê 11.9 Lấy TV bằng cách sử dụngphảitruy vấn**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["sản phẩm","thương hiệu"], "truy vấn": {

"bool": {

"phải": [

{

"cuộc thi đấu": {

**Mệnh đề phải có hai truy vấn khớp riêng lẻ**

"sản phẩm": "TV"

}

},

{

"cuộc thi đấu": {

"thương hiệu": "LG"

}

}

]

}

}

}

Không có nhiều điều để phân tích trong truy vấn này ngoại trừ \_score được đề cập trong kết quả, đó là 4.4325914:

"lượt truy cập" : [

{

"\_index" : "sản phẩm", "\_id" : "5",

**"\_điểm" : 4.4325914,**

"\_ignored" : [ "tổng quan.từ khóa"

],

"\_source" : { "sản phẩm" : "TV",

"thương hiệu" : "LG"

}

}

]

Bây giờ, hãy thêm mệnh đề should vào truy vấn này. Danh sách sau đây cho thấy tác động lên điểm số.

**Liệt kê 11.10 Thêm mộtnênđiều khoản để tăng điểm**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["sản phẩm","thương hiệu"], "truy vấn": {

"bool": {

"phải": [

**Mệnh đề phải là**

**tìm kiếm TV LG**

**376 CPHẦN11*Truy vấn hợp chất***

{

"cuộc thi đấu": {

"sản phẩm": "TV"

}

},

{

"cuộc thi đấu": {

"thương hiệu": "LG"

}

} **Mệnh đề should kiểm tra**

], **phạm vi giá hoặc sự phù hợp**

"nên": [

{

**cụm từ của TV kết quả**

"phạm vi": {

"giá": {

"gte": 500,

"lte": 1000

}

}

},

{

"match\_phrase\_prefix": { "tổng quan": "4K Ultra HD"

}

}

]

}

}

}

Truy vấn này, sử dụng mệnh đề must với mệnh đề should, sẽ tăng điểm của các tài liệu khớp. Điểm trước đó là 4.4325914 giờ đã lên tới 14.9038105:

"lượt truy cập" : [

{

"\_index" : "sản phẩm", "\_id" : "5",

**"\_điểm" : 14.9038105,**

"\_ignored" : [ "tổng quan.từ khóa"

],

"\_source" : { "sản phẩm" : "TV",

"thương hiệu" : "LG"

}

}

]

Điểm số tăng lên vì truy vấn đã thành công trong mệnh đề must và cũngkhớp trong mệnh đề should. Điểm mấu chốt là nếu truy vấn trong mệnh đề should khớp (ngoài các khớp tích cực trong mệnh đề must), điểm sẽ tăng. Nói như vậy, nó nên khớp với bao nhiêu truy vấn? Có thể có một số truy vấn lá trong mệnh đề should, đúng không? Tất cả các truy vấn lá cần phải khớp hay chúng ta có thể yêu cầu Elasticsearch

***11.3 Boolean (bool)truy vấn* 377**

để kiểm tra xem ít nhất một trong các truy vấn lá có khớp không? Điều này có thể đạt được bằng

tối thiểu\_phải\_phù hợpthuộc tính, sẽ được thảo luận tiếp theo.

**TTỐI THIỂU****\_NÊN\_THIẾT LẬP TRẬN ĐẤU**

Khi chúng ta chạy một tập hợp các truy vấn trong mệnh đề must cùng với các truy vấn trong mệnh đề should, các quy tắc sau đây được áp dụng ngầm định:

* Tất cả các kết quả phải phù hợp với tiêu chí truy vấn được khai báo trongphảiđiều khoản (cáctruy vấn sẽ không trả về kết quả tìm kiếm tích cực nếu một trong cácphải(các truy vấn không khớp với tiêu chí).
* Không cần bất kỳ kết quả nào phải khớp với tiêu chí đã nêu trongnênđiều khoản. Nếu chúng khớp nhau,\_điểmđược tăng cường; nếu không, sẽ không có tác động nào đến điểm số.

Tuy nhiên, đôi khi chúng ta có thể cần ít nhất một trong các tiêu chí should phải khớp trước khi gửi kết quả cho máy khách. Chúng ta muốn điểm được tăng dựa trên các kết quả khớp truy vấn should này. Chúng ta có thể thực hiện điều này bằng cách sử dụng thuộc tính minimum\_should\_match.

Ví dụ, chúng ta có thể tuyên bố truy vấn từ danh sách 11.10 thành công nếu và chỉ nếu ít nhất một trong các kết quả khớp là dương; nó trả về kết quả dương (với điểm tăng) chỉ khi một trong nhiều truy vấn khớp. Danh sách sau đây cho thấy điều này.

**Liệt kê 11.11 Sử dụngtối thiểu\_phải\_phù hợptham số**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["sản phẩm","thương hiệu","tổng quan", "giá","màu sắc"], "truy vấn": {

"bool": {

"phải": [{

"cuộc thi đấu": {

"sản phẩm": "TV"

}

},

{

"cuộc thi đấu": {

"thương hiệu": "LG"

}

}],

"nên": [{

"phạm vi": {

"giá": {

"gte": 500,

"lte": 2000

}

}

},

{

"cuộc thi đấu": {

"màu sắc": "bạc"

}

},

**378 CPHẦN11*Truy vấn hợp chất***

{

"match\_phrase\_prefix": { "tổng quan": "4K Ultra HD"

}

}],

"minimum\_should\_match": 1

}

}

}

**Phù hợp với ít nhất một truy vấn lá trong mệnh đề should**

Trong danh sách, chúng tôi đặt minimum\_should\_match thành 1. Điều này có nghĩa là truy vấn cố gắng khớp với các tiêu chí được xác định trong các truy vấn lá của mệnh đề should nhưng với điều kiện là ít nhất một trong các truy vấn phải khớp với giá trị dương: nếu sản phẩm là bạc hoặc sản phẩm là 4K Ultra HD hoặc giá nằm trong khoảng từ 500 đến 2.000 đô la. Tuy nhiên, nếu không có tiêu chí nào trong mệnh đề should khớp, truy vấn sẽ không thành công vì chúng tôi yêu cầu truy vấn thỏa mãn tham số minumum\_should\_match.

Như bạn có thể đã đoán, mộtbooltruy vấn có thể được khai báo với mộtnênđiều khoản riêng của nó—tức là, không cóphảiĐiều khoản. Dựa trên việc liệu mộtbooltruy vấn bao gồm mộtphảiđiều khoản cùng vớinênmệnh đề, giá trị mặc định củatối thiểu\_phải\_phù hợpthay đổi. Giá trị mặc định củatối thiểu\_phải\_phù hợpđược đặt thành 0 nếubooltruy vấn bao gồm mộtnênvới mộtphảimệnh đề; nó được đặt thành 1 với chỉ mộtnênđiều khoản (xem bảng 11.3).

**Bảng 11.3 Giá trị mặc định củatối thiểu\_phải\_phù hợpthuộc tính**

|  |  |
| --- | --- |
| **Các điều khoản** | **Giá trị mặc định của**  **tối thiểu\_phải\_phù hợp** |
| nênchỉ có điều khoản (khôngphảiđiều khoản) | 1 |
| nênvới mộtphảiđiều khoản | 0 |

Cho đến nay, chúng ta đã xem xét các mệnh đề must, must\_not và should. Mặc dù must\_not chạy trong ngữ cảnh bộ lọc, must và should chạy trong ngữ cảnh truy vấn. Chúng ta đã thảo luận về ngữ cảnh truy vấn so với ngữ cảnh bộ lọc trong chương 8, nhưng để đầy đủ, chúng ta hãy cùng tìm hiểu các khái niệm này tại đây. Các truy vấn chạy trong ngữ cảnh truy vấn thực thi thuật toán liên quan thích hợp, do đó chúng ta có thể mong đợi điểm liên quan liên quan đến các tài liệu kết quả. Các truy vấn trong ngữ cảnh bộ lọc không đưa ra điểm và có hiệu suất cao vì không cần thực thi thuật toán chấm điểm. Điều đó dẫn đến một mệnh đề không sử dụng điểm liên quan nhưng hoạt động trong ngữ cảnh bộ lọc: mệnh đề bộ lọc.

* + 1. Điều khoản lọc

Mệnh đề filter lấy tất cả các tài liệu khớp với tiêu chí, tương tự như mệnh đề must. Điểm khác biệt duy nhất là mệnh đề filter chạy trong ngữ cảnh filter, do đó kết quả không được chấm điểm. Hãy nhớ rằng, chạy truy vấn trong ngữ cảnh filter sẽ tăng tốc hiệu suất truy vấn vì nó lưu trữ đệm các kết quả truy vấn được Elasticsearch trả về. Danh sách sau đây hiển thị truy vấn filter đang hoạt động.

***11.3 Boolean (bool)truy vấn* 379**

**Liệt kê 11.12lọcđiều khoản**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"lọc": [{

"thuật ngữ": {

**Thực hiện truy vấn trong bối cảnh bộ lọc một cách ngầm định**

"sản phẩm.từ khóa": "TV"

},

{

"phạm vi": {

"giá": {

"gte": 500,

"lte": 1000

}

}

}

}

]

}

}

}

Đoạn trích sau đây cho thấy kết quả của truy vấn này. Lưu ý rằng điểm kết quả là 0:

"lượt truy cập" : [{

...

**"\_điểm" : 0.0,**

"\_source" : { "sản phẩm" : "TV",

"màu sắc" : "bạc", "thương hiệu" : "Samsung"

}

},

{

...

"\_điểm" : 0.0,

"\_source" : { "sản phẩm" : "TV",

"màu sắc" : "đen", "thương hiệu" : "Samsung"

}

}

...

]

Mệnh đề bộ lọc không cung cấp điểm. Vì điểm không bắt buộc đối với đầu ra, Elasticsearch có thể lưu trữ bộ nhớ đệm truy vấn/kết quả, điều này có lợi cho hiệu suất của ứng dụng.

Chúng tôi thường kết hợp một mệnh đề lọc với một mệnh đề phải. Kết quả của mệnh đề phải được đưa qua mệnh đề lọc, mệnh đề này lọc dữ liệu không phù hợp. Ví dụ tiếp theo cho thấy cách tiếp cận này.

**380 CPHẦN11*Truy vấn hợp chất***

**Liệt kê 11.13 Alọcmệnh đề với mộtphảiđiều khoản**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["thương hiệu","sản phẩm","màu sắc","giá"], "truy vấn": {

"bool": {

"phải": [{

"cuộc thi đấu": {

"thương hiệu": "LG"

}

}

],

"lọc": [{

"phạm vi": {

"giá": {

"gte": 500,

"lte": 1000

}

}

}

]

}

}

}

Ở đây, chúng tôi lấy tất cả các sản phẩm của LG (chúng tôi có một TV và ba tủ lạnh do LG sản xuất trong kho của mình). Sau đó, chúng tôi lọc chúng theo giá, chỉ để lại hai tủ lạnh nằm trong phạm vi giá của chúng tôi (cả hai đều có giá 900 đô la):

"lượt truy cập" : [{

..

"\_score" : 2.6820748, "\_source" : {

"sản phẩm": "Tủ lạnh", "màu sắc": "Đen mờ", "giá": 900,

"thương hiệu" : "LG"

}

},{

..

"\_score" : 2.6820748, "\_source" : {

"sản phẩm": "Tủ lạnh", "màu sắc": "Đen mờ", "giá": 900,

"thương hiệu" : "LG"

}

}]

Lưu ý rằng cả hai tài liệu được trả về hiện đều có điểm, nghĩa là truy vấn đã đượcđược thực hiện trong ngữ cảnh truy vấn. Như chúng ta đã thảo luận, truy vấn lọc tương tự như truy vấn phải ngoại trừ việc nó không được chạy trong ngữ cảnh truy vấn (truy vấn lọc được chạy trong ngữ cảnh lọc). Do đó, việc thêm bộ lọc không ảnh hưởng đến việc chấm điểm tài liệu.

***11.3 Boolean (bool)truy vấn* 381**

Cho đến nay, chúng ta đã làm việc với các truy vấn bool sử dụng các mệnh đề riêng lẻ với các truy vấn lá. Chúng ta có thể kết hợp tất cả các mệnh đề này để xây dựng một truy vấn nâng cao phức tạp. Hãy xem xét điều đó trong phần tiếp theo.

* + 1. Kết hợp tất cả các mệnh đề

Hãy kết hợp các mệnh đề must, must\_not, should và filter. Yêu cầu là tìm các sản phẩm do LG sản xuất không phải là bạc, là tủ lạnh đông hoặc có xếp hạng năng lượng A++ và nằm trong phạm vi giá cụ thể.

Truy vấn trong danh sách 11.14 lấy các sản phẩm LG với truy vấn khớp trong mệnh đề bắt buộc,bỏ qua màu bạc trong mệnh đề must\_not và truy vấn tủ lạnh đông đá hoặc xếp hạng năng lượng nhất định bằng mệnh đề should. Cuối cùng, chúng tôi sử dụng mệnh đề filter để kiểm tra giá của sản phẩm để xem chúng có phù hợp với phạm vi giá cụ thể hay không.

**Liệt kê 11.14 Tất cả các mệnh đềkết hợp**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{

"cuộc thi đấu": {

"thương hiệu": "LG"

}

}],

"must\_not": [{

"thuật ngữ": {

"màu sắc": "bạc"

}

}],

"nên": [{

"phù hợp": { "đánh giá năng lượng": "A++"

}

},

{

"thuật ngữ": {

"loại": "Tủ lạnh tủ đông"

}

}],

"lọc": [{

"phạm vi": {

"giá": {

"gte": 500,

"lte": 1000

}

}

}

]

}

}

}

**382 CPHẦN11*Truy vấn hợp chất***

Danh sách này kết hợp bốn mệnh đề bằng cách sử dụng truy vấn bắt buộc, khớp với TV LG nhưng không khớp với TV màu bạc (mệnh đề must\_not). Nếu sản phẩm có xếp hạng năng lượng A++ hoặc là tủ lạnh đông (mệnh đề should), thì thậm chí còn tốt hơn. Cuối cùng, chúng tôi lọc sản phẩm theo giá (mệnh đề filter).

Chúng ta có thể tăng cường truy vấn bằng các mệnh đề và truy vấn lá khi các yêu cầu phức tạp hơn được thêm vào danh sách. Không có giới hạn về số lượng truy vấn mà một mệnh đề có thể chứa; hoàn toàn tùy thuộc vào quyết định của chúng ta. Nhưng làm sao chúng ta biết được truy vấn lá nào trong các mệnh đề khác nhau được khớp để có kết quả? Có bất kỳ truy vấn nào bị bỏ qua dựa trên các yêu cầu của chúng ta không? Biết chính xác truy vấn lá nào đã thu được kết quả cuối cùng (hay còn gọi là kết quả) là một cách tuyệt vời để xác định các truy vấn đã được thực thi và tạo ra kết quả. Chúng ta có thể thực hiện việc này bằng cách đặt tên cho mọi truy vấn.

* + 1. Truy vấn được đặt tên

Chúng ta có thể xây dựng hàng chục truy vấn cho một truy vấn phức tạp; tuy nhiên, chúng ta không biết có bao nhiêu truy vấn được sử dụng trong một lần khớp để có được kết quả cuối cùng. Chúng ta có thể đặt tên cho các truy vấn của mình để Elasticsearch đưa ra kết quả cùng với tên của các truy vấn được sử dụng trong quá trình khớp truy vấn. Hãy cùng xem một ví dụ. Danh sách tiếp theo hiển thị một truy vấn phức tạp với tất cả các mệnh đề và một vài truy vấn lá.

**Liệt kê 11.15 Các truy vấn phức tạp được đặt tên riêng**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["sản phẩm", "thương hiệu"], "truy vấn": {

"bool": {

"phải": [

{

"cuộc thi đấu": {

"thương hiệu": { "truy vấn": "LG",

"\_name": "phải\_khớp\_với\_truy\_cập\_hàng\_thương\_hiệu"

**Tên truy vấn phải khớp với thương hiệu trong mệnh đề bắt buộc**

}

}

}

],

"must\_not": [

{

"phù hợp": { "màu sắc.từ khóa": {

"query":"đen", "\_name":"must\_not\_colour\_query"

**Tên truy vấn không khớp với màu cụ thể**

}

}

}

],

"nên": [

{

"thuật ngữ": {

* 1. ***Truy vấn Boolean (bool)***

"loại.từ khóa": {

"value": "Tủ lạnh không đóng tuyết", "\_name":"should\_term\_type\_query"

}

}

},

{

**Tên truy vấn để khớp với kiểu trong mệnh đề should**

**383**

"phù hợp": { "đánh giá năng lượng": {

"query": "A++", "\_name":"should\_match\_energy\_rating\_query"

}

}

}

],

**Tên truy vấn để khớp với xếp hạng năng lượng trong**

**một mệnh đề nên**

"lọc": [

{

"phạm vi": {

"giá": {

"gte": 500,

"lte": 1000,

"\_name":"filter\_range\_price\_query"

}

}

}

]

}

}

}

**Tên truy vấn để khớp với phạm vi giá trong**

**một điều khoản lọc**

Các truy vấn lá riêng lẻ được gắn thẻ với thuộc tính \_name với giá trị do chúng tôi lựa chọn. Sau khi truy vấn được thực thi, phản hồi bao gồm một đối tượng matches\_queries được đính kèm vào mỗi kết quả. Kèm theo trong matches\_queries này là tập hợp các truy vấn được khớp để lấy tài liệu:

"lượt truy cập" : [

{

...

"\_nguồn" : {

"sản phẩm": "Tủ lạnh",

"thương hiệu" : "LG"

},

"matched\_queries": [ "filter\_range\_price\_query", "should\_match\_energy\_rating\_query", "must\_match\_brand\_query", "should\_term\_type\_query"

]

},

{

...

"\_nguồn" : {

"sản phẩm": "Tủ lạnh",

"thương hiệu" : "LG"

**384 CPHẦN11*Truy vấn hợp chất***

},

"matched\_queries": [ "filter\_range\_price\_query", "should\_match\_energy\_rating\_query", "must\_match\_brand\_query", "should\_term\_type\_query"

]

}

]

Kết quả được khớp với bốn truy vấn được đề cập trong khối matches\_queries. Lợi ích thực sự của việc đặt tên truy vấn là loại bỏ các truy vấn dư thừa không liên quan đến kết quả. Theo cách này, chúng ta có thể giảm kích thước truy vấn và tập trung vào việc điều chỉnh các truy vấn là một phần của việc tìm nạp kết quả.

Như vậy là chúng ta đã kết thúc phần xem xét truy vấn bool, một trong những truy vấn phức hợp quan trọng và phức tạp nhất. Trong một vài phần tiếp theo, chúng ta sẽ xem xét các truy vấn phức hợp khác, bắt đầu với điểm số hằng số.

#### Điểm số liên tục

Trước đây, chúng ta đã xem xét truy vấn bộ lọc trong mệnh đề bool. Để hoàn thiện, hãy chạy lại truy vấn bộ lọc mẫu để tìm các sản phẩm có xếp hạng của người dùng từ 4 đến 5.

**Liệt kê 11.16lọcđiều khoản được tuyên bố trong mộtbooltruy vấn**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"lọc": [

{

"phạm vi": { "xếp hạng người dùng": {

"gte": 4,

"lte": 5

}

}

}

]

}

}

}

Truy vấn đưa ra tất cả các sản phẩm khớp với tiêu chí đánh giá của người dùng. Điểm đáng chú ý duy nhất là truy vấn được thực hiện trong ngữ cảnh bộ lọc; do đó, không có điểm số nào (không) được liên kết với kết quả. Tuy nhiên, chúng ta có thể cần đặt điểm số khác không, đặc biệt là khi chúng ta muốn tăng cường một tiêu chí tìm kiếm cụ thể. Đây là nơi một loại truy vấn mới, constant\_ score, xuất hiện.

Như tên gọi của nó, constant\_score gói một truy vấn lọc và tạo ra kết quả với điểm số được xác định trước (được tăng cường). Truy vấn trong danh sách sau đây cho thấy điều này đang diễn ra.

* 1. ***Không thay đổiđiểm số* 385**

**Liệt kê 11.17 Ađiểm số hằng sốtruy vấn tạo ra điểm số tĩnh**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "điểm\_hằng\_số": {

"lọc": {

"phạm vi": { "xếp hạng người dùng": {

"gte": 4,

"lte": 5

**Tuyên bố**

**truy vấn constant\_score**

**Bao bọc một truy vấn lọc**

}

}

},

"tăng cường": 5.0

}

}

}

**Tăng cường kết quả bằng cách sử dụng điểm số được xác định trước**

Truy vấn constant\_score trong danh sách này bao gồm một truy vấn lọc. Nó cũng có một thuộc tính khác, boost, giúp tăng điểm với giá trị đã cho. Do đó, tất cả các tài liệu kết quả đều được đóng dấu với điểm là 5 thay vì 0.

Nếu bạn đang thắc mắc về việc sử dụng thực tế củađiểm số hằng số, không cần tìm đâu xa nữa.Danh sách sau đây cho thấy mộtbooltruy vấn nơi chúng ta gói mộtđiểm số hằng sốchức năng thành mộtphảitruy vấn cùng với mộtcuộc thi đấutruy vấn.

**Liệt kê 11.18 Abooltruy vấn với điểm số không đổi**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [{

"cuộc thi đấu": {

**Mệnh đề must với hai truy vấn: match và constant\_score**

**Truy vấn khớp mà**

"sản phẩm": "TV"

}

},

{

**tìm kiếm TV**

"điểm\_hằng\_số": { "bộ lọc": {

"thuật ngữ": {

"màu sắc": "đen"

**Constant\_score tăng điểm lên 3,5 nếu màu của TV là màu đen**

}

},

"tăng cường": 3.5

}

}

]

}

}

}

**386 CPHẦN11*Truy vấn hợp chất***

Mệnh đề must trong truy vấn bool này chứa hai truy vấn: một match và một constant\_score. Truy vấn constant\_score lọc tất cả các TV dựa trên màu sắc, nhưng có một điều chỉnh: nó tăng điểm lên 3,5 cho tất cả các TV màu đen. Ở đây, chúng tôi yêu cầu công cụ Elasticsearch lấy thông tin đầu vào của chúng tôi khi chấm điểm kết quả bằng cách đặt giá trị boost thành lựa chọn của chúng tôi trong truy vấn đã lọc được gói trong constant\_score.

Trong phần này, chúng ta đã thấy kết quả truy vấn được gán một điểm số tĩnh bằng cách sử dụng hằng số\_hàm điểm. Nhưng nếu chúng ta muốn một số kết quả được chấm điểm cao hơn và một số khác ở cuối trang kết quả thì sao? Đó chính là chức năng của truy vấn tăng cường, như chúng ta sẽ thảo luận tiếp theo.

#### Truy vấn tăng cường

Đôi khi chúng ta muốn có câu trả lời thiên vị. Ví dụ, chúng ta có thể muốn danh sách kết quả có TV LG ở đầu và Sony ở cuối. Kiểu thao túng điểm số thiên vị nàyvì vậy danh sách có các mục được ưu tiên ở đầu được thực hiện bằng truy vấn tăng cường. Truy vấn tăng cường hoạt động với hai bộ truy vấn: một phần tích cực, trong đó bất kỳ số lượng truy vấn nào cũng tạo ra kết quả khớp tích cực; và một phần tiêu cực, khớp các truy vấn để phủ định điểm bằng tăng cường tiêu cực.

Hãy xem xét một ví dụ. Chúng tôi muốn tìm kiếm TV LG, nhưng nếu giá lớn hơn 2.500 đô la, chúng tôi sẽ đưa chúng xuống cuối danh sách bằng cách sử dụng điểm được tính toán bởigiá trị được chỉ định bởi sự tăng cường tiêu cực của truy vấn tiêu cực. Hãy xem cách thực hiện điều này.

**Liệt kê 11.19 Atăng cườngtruy vấn trong hành động**

NHẬN sản phẩm/\_tìm kiếm

{

"kích thước": 50,

"\_source": ["sản phẩm", "giá","màu sắc"], "truy vấn": {

**Một sự thúc đẩy**

**truy vấn trong hành động**

"tăng cường": {

"tích cực": {

"thuật ngữ": {

"sản phẩm":"tv"

**Truy vấn tăng cường**

**phần tích cực**

}

},

"tiêu cực": {

"phạm vi": {

"giá": { "gte": 2500

**Phần tiêu cực của truy vấn tăng cường**

}

}

},

"tăng\_âm\_âm": 0,5

}

}

}

**Tiêu cựctăng**

Như đã trình bày, truy vấn tăng cường có hai phần: tích cực và tiêu cực. Trong phần tích cực, chúng tôi chỉ cần tạo một truy vấn (một truy vấn thuật ngữ, trong trường hợp này) để lấy TV. Mặt khác, chúng tôi không muốn những chiếc TV quá đắt tiền, vì vậy chúng tôi loại bỏ những chiếc TV đắt tiền khỏi

* 1. ***Sự tăng cườngtruy vấn* 387**

kết quả (di chuyển chúng xuống cuối danh sách) bằng cách phủ định điểm của những kết quả khớp trong phần phủ định của truy vấn. Giá trị được thiết lập bởi thuộc tính negative\_boost được sử dụng để tính toán lại điểm của những kết quả khớp trong phần phủ định. Điều này đẩy kết quả từ phần phủ định xuống cuối danh sách.

Truy vấn tăng cường trong danh sách 11.19 rất đơn giản: nó sử dụng các truy vấn lá ở phần tích cực và tiêu cực. Tuy nhiên, chúng ta cũng có thể viết một truy vấn hợp chất như truy vấn tăng cường với các truy vấn hợp chất khác. Chúng ta có thể khai báo truy vấn tăng cường với bool, constant\_score hoặc các truy vấn hợp chất khác, bao gồm các truy vấn lá cấp cao nhất. Truy vấn trong danh sách tiếp theo cho thấy điều này: một truy vấn tăng cường với các truy vấn bool được nhúng.

**Danh sách 11.20 Abooltruy vấn với điểm số không đổi**

NHẬN sản phẩm/\_tìm kiếm

{

"kích thước": 40,

"\_source": ["sản phẩm", "giá","màu sắc","thương hiệu"], "truy vấn": {

"tăng cường": {



"tích cực": {

"bool": {

"phải": [

{

"cuộc thi đấu": {

**Khai báo một truy vấn tăng cường**

**Xác định phần dương của truy vấn tăng cường trong truy vấn bool, cũng có mệnh đề must trong truy vấn khớp**

"sản phẩm": "TV"

}

}

]

}

},

"tiêu cực": {

"bool": {

"phải": [

{

**Xác định phần phủ định của truy vấn với truy vấn bool được nhúng**

"cuộc thi đấu": {

"thương hiệu": "Sony"

}

}

]

}

},

"tăng\_âm\_âm": 0,5

}

}

}

**Đặt negative\_boost thành 0,5 khi khớp thành công với các truy vấn từ phần phủ định**

Truy vấn tăng cường này bao gồm cả phần dương và phần âm, như mong đợi, và phần âm có giá trị negative\_boost được đặt thành 0,5. Truy vấn hoạt động như sau: tìm kiếm TV (như được hiển thị trong khối dương) và nếu thương hiệu TV là Sony, điểm của nó sẽ giảm 0,5. Mặc dù TV Sony có thể tuyệt vời, nhưng chúng được gửi đến cuối tập kết quả vì chúng tôi đã thao túng điểm của chúng bằng cách sử dụng cài đặt negative\_boost.

**388 CPHẦN11*Truy vấn hợp chất***

Do đó, truy vấn tăng cường giúp chúng ta hạ thấp một loại tài liệu nhất định bằng cách sử dụng điểm số âm. Chúng ta có thể chuẩn bị kết quả bằng cách thao tác điểm số dựa trên truy vấn âm và tăng cường âm. Bây giờ chúng ta hãy chuyển sang một truy vấn hợp chất khác: dis-junction max (dis\_max).

#### Truy vấn disjunction max (dis\_max)

Trong chương 10, chúng tôi đã làm việc với một truy vấn có tên là multi\_match, dùng để tìm kiếm các từtrên nhiều trường. Để tìm kiếm một chiếc TV thông minh trên hai trường, nhập và tổng quan, chúng ta có thể sử dụng multi\_match. Để hoàn thiện, đây là truy vấn multi\_match kết quả.

**Liệt kê 11.21 Tìm kiếm nhiều trường bằng cách sử dụngnhiều\_trận\_hợptruy vấn**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "multi\_match": {

"query": "smart tv", "fields": ["type","overview"]

}

}

}

Chúng tôi đưa ra truy vấn multi\_match dưới tiêu đề disjunction max (dis\_max) truy vấn vì multi\_match sử dụng truy vấn dis\_max ở chế độ ẩn. Truy vấn dis\_max gói một số truy vấn và mong đợi ít nhất một trong số chúng khớp. Nếu có nhiều hơn một truy vấn khớp, truy vấn dis\_max trả về tài liệu có điểm liên quan cao nhất. Hãy xem lại truy vấn từ danh sách 11.21 nhưng lần này sử dụng dis\_max để tìm kiếm các từ “smart tv” trong các trường loại và tổng quan.

**Liệt kê 11.22dis\_maxtruy vấn trong hành động**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["type","overview"], "query": {

"dis\_max": {

**Khai báo truy vấn dis\_max**

"truy vấn": [{

"cuộc thi đấu": {

**Khai báo một tập hợp các truy vấn có điều kiện khớp**

**bao gồm một tập hợp các truy vấn**

"loại": "TV thông minh"

}

},

{

"cuộc thi đấu": {

"tổng quan": "TV thông minh"

}

}]

}

}

}

* 1. ***Điểm số hàmtruy vấn* 389**

Truy vấn dis\_max là truy vấn hợp chất mong đợi một số truy vấn láđược định nghĩa trong đối tượng truy vấn. Ở đây, chúng tôi khai báo hai truy vấn khớp, tìm kiếm nhiều từ trong hai trường khác nhau: loại và tổng quan.

Khi tìm kiếm nhiều từ trên nhiều trường, Elasticsearch sử dụng chiến lược trường tốt nhất, ưu tiên tài liệu có tất cả các từ trong các trường đã cho. Đối vớiVí dụ, giả sử chúng ta đang tìm kiếm “smart TV” trên hai trường, tổng quan và loại. Chúng ta có thể mong đợi rằng một tài liệu có cụm từ này trong cả hai trường sẽ có liên quan hơn một tài liệu có “smart” trong trường tổng quan và “TV” trong trường loại.

Khi thực hiệntruy vấn dis\_max trên nhiều trường, chúng ta cũng có thể xem xét điểm từ các truy vấn khớp khác. Trong trường hợp này, chúng ta sử dụng một tiebreaker để thêm điểm từcác trường khác khớp, không chỉ các trường tốt nhất. Hãy thêm thuộc tính tie\_breaker vào truy vấn sau.

**Liệt kê 11.23 Adis\_maxtruy vấn với một tiebreaker**

NHẬN sản phẩm/\_tìm kiếm

{

"\_source": ["type","overview"], "query": {

"dis\_max": {

"truy vấn": [{

"cuộc thi đấu": {

"loại": "TV thông minh"

}

},

{

"cuộc thi đấu": {

"tổng quan": "TV thông minh"

}

},

{

"cuộc thi đấu": {

"sản phẩm": "TV thông minh"

}

}],"người phá vỡ thế cân bằng": 0,5

}

}

}

Giá trị tie\_breaker là số dấu phẩy động dương giữa 0,0 và 1,0 (mặc định là 0,0). Trong trường hợp này, chúng tôi nhân điểm của các trường không tốt nhất với tiebreaker và cộng kết quả vào điểm của mỗi tài liệu khớp với nhiều trường.

Truy vấn hợp chất cuối cùng mà chúng ta xem xét trong chương này là truy vấn điểm số hàm. Nó cung cấp cho chúng ta nhiều sự linh hoạt hơn khi chỉ định điểm số dựa trên nhu cầu của chúng ta bằng cách sử dụng các hàm được xác định trước.

#### Truy vấn function\_score

Đôi khi chúng tôi muốn chỉ định điểm cho một tài liệu được trả về từ truy vấn tìm kiếm dựa trên các yêu cầu nội bộ, chẳng hạn như đưa ra trọng số cho một trường cụ thể hoặc hiển thị

**390 CPHẦN11*Truy vấn hợp chất***

quảng cáo của nhà tài trợ dựa trên điểm liên quan ngẫu nhiên. Các truy vấn điểm hàm (function- tion\_score) giúp tạo điểm dựa trên các hàm do người dùng xác định, bao gồm các hàm ngẫu nhiên, dựa trên tập lệnh hoặc hàm phân rã (Gauss, tuyến tính, v.v.).

Trước khi bắt đầu làm việc với truy vấn function\_score, hãy thực hiện truy vấn trong danh sách sau. Đây là truy vấn thuật ngữ đơn giản trả về các tài liệu.

**Liệt kê 11.24 Tìm kiếm thuật ngữ với một tiêu chuẩnthuật ngữtruy vấn**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": {

"thuật ngữ": {

"sản phẩm": {

"giá trị": "TV"

}

}

}

}

Truy vấn không làm được gì nhiều ngoài việc tìm kiếm TV bằng truy vấn thuật ngữ chuẩn. Điểm duy nhất cần lưu ý là điểm của tài liệu hàng đầu được truy vấn này trả về: 1.6376086.

Mặc dù truy vấn này có mục đích đơn giản, một số truy vấn yêu cầu xử lý nặng để tính điểm liên quan. Có thể chúng ta không quan tâm đến việc lấy điểm được tính bằng thuật toán liên quan Best Match 25 (BM25) của Elasticsearch (hoặc bất kỳ thuật toán tùy chỉnh nào) vì chúng ta muốn tạo điểm dựa trên yêu cầu của riêng mình. Trong trường hợp như vậyTrong trường hợp này, chúng ta có thể gói truy vấn trong cấu trúc function\_score để tạo điểm dựa trên các hàm do người dùng xác định.

**Liệt kê 11.25 Athuật ngữtìm kiếm được gói gọn trong mộtđiểm\_hàm**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "điểm\_hàm": {

"truy vấn": {

"thuật ngữ": {

"sản phẩm": "tv"

**function\_score gói một truy vấn để tạo ra điểm số do người dùng xác định.**

}

}

}

}

}

Truy vấn function\_score mong đợi một vài thuộc tính: truy vấn, hàm, cách áp dụng điểm vào tài liệu, v.v. Thay vì tìm hiểu về chúng trên lý thuyết, chúng ta sẽ thấy những điều này trong các ví dụ thực hành trong phần này.

Các hàm do người dùng định nghĩa cho phép chúng ta sửa đổi và thay thế điểm số bằng điểm số tùy chỉnh của chúng ta. Chúng ta có thể làm như vậy bằng cách cắm vào một hàm điều chỉnh điểm số dựa trên

***11.7 Hàm\_scoretruy vấn* 391**

yêu cầu. Ví dụ, nếu chúng ta muốn một điểm số được tạo ngẫu nhiên, có mộttruy vấn hàm random\_score cho mục đích đó. Hoặc chúng ta có thể muốn tính điểm dựa trên các giá trị trường và tham số, trong trường hợp đó chúng ta có thể sử dụng truy vấn hàm script\_score. Chúng ta cũng xem xét một vài hàm khác trong phần này, nhưng hãy bắt đầu với random\_score.

* + 1. Hàm random\_score

Như tên gọi của nó, hàm random\_score tạo ra một điểm số được tạo ngẫu nhiên cho các tài liệu kết quả. Chúng ta có thể chạy truy vấn từ danh sách 11.25, được gói trong truy vấn function\_score, nhưng lần này chỉ định cụ thể một hàm random\_score cho truy vấn.

**Danh sách 11.26 Tìm kiếm thuật ngữ được gói trong mộtđiểm\_ngẫu\_nhiênchức năng**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "điểm\_hàm": {

"truy vấn": {

"thuật ngữ": {

"sản phẩm": "TV"

}

},

"điểm\_ngẫu\_nhiên": {}

}

}

}

**Một function\_score với một truy vấn thuật ngữ và một hàm**

**Một hàm random\_score tạo và gán một điểm số ngẫu nhiên cho mỗi lần gọi**

Truy vấn function\_score này được tạo thành từ một truy vấn thuật ngữ và một hàm random\_score. Mỗi lần chúng ta thực hiện truy vấn này, chúng ta sẽ nhận được một điểm số khác nhau cho cùng một tài liệu được trả về. Điểm số ngẫu nhiên là, ừm, ngẫu nhiên và không thể tái tạo được. Khi bạn thực hiện lại truy vấn, hãy mong đợi điểm số sẽ thay đổi.

Nếu chúng ta có yêu cầu tái tạo điểm ngẫu nhiên sao cho bất kể chúng ta thực hiện cùng một truy vấn bao nhiêu lần, điểm được tạo ngẫu nhiên luôn giống nhau? Với mục đích này, chúng ta có thể điều chỉnh hàm random\_score bằng cách mồi nó bằng các giá trị hạt giống và trường. Danh sách sau đây hiển thị truy vấn với hàm random\_score được khởi tạo bằng hạt giống.

**Liệt kê 11.27 Điều chỉnh điểm ngẫu nhiên bằng cách thiết lậphạt giống**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "điểm\_hàm": {

"truy vấn": {

"thuật ngữ": {

"sản phẩm": "TV"

}

**392 CPHẦN11*Truy vấn hợp chất***

},

"random\_score": { "hạt giống": 10,

**Khởi tạo random\_score tùy chỉnh bằng một hạt giống**

"trường":"user\_ratings"

}

}

}

}

**Tính điểm ngẫu nhiên**

Như bạn có thể thấy, random\_score được khởi tạo bằng giá trị hạt giống và giá trị trường user\_ratings. Nếu chúng ta thực hiện truy vấn này nhiều lần, chúng ta chắc chắn sẽ nhận được cùng một điểm (mặc dù là ngẫu nhiên). Thuật toán và cơ chế để xác định điểm ngẫu nhiên nằm ngoài phạm vi của cuốn sách này; hãy tham khảo tài liệu Elasticsearch nếu bạn muốn hiểu thêm về cơ chế chấm điểm ngẫu nhiên.

Trong khi hàm random\_score là một cách để tạo điểm ngẫu nhiên, việc tạo điểm tĩnh bằng hàm script cũng rất thú vị. Trong phần tiếp theo, chúng ta hãy xem cách chúng ta có thể sử dụng hàm script\_score.

* + 1. Hàm script\_score

Giả sử chúng ta muốn tăng gấp ba điểm số tài liệu (nhân giá trị của trường với hệ số 3) dựa trên giá trị của trường (ví dụ: user\_rating của sản phẩm). TrongTrong trường hợp này, chúng ta có thể sử dụng hàm script\_score để tính điểm dựa trên giá trị của các trường khác (như user\_ratings) trong tài liệu.

**Liệt kê 11.28 Nhân giá trị của trường với một tham số bên ngoài**

NHẬN sản phẩm/\_tìm kiếm

{

**Kịch bản**

"truy vấn": { "điểm\_hàm": {

"truy vấn": {

"thuật ngữ": {

"sản phẩm": "tv"

}

},

"script\_score": { "script": { #B

**Hàm script\_score giữ vai trò quan trọng trong việc tạo ra điểm số dựa trên tập lệnh được xác định.**

**sự vật**

"nguồn":"\_score \* doc['user\_ratings'].value \* params['factor']",

"tham số": {

"yếu tố":3

}

}

}

}

}

}

**Truyền các tham số bên ngoài vào tập lệnh**

**Nguồn là nơi chúng ta xác định logic của mình.**

Hàm script\_score tạo ra một điểm số và trong ví dụ này, nó tính toán điểm số đó dựa trên phép tính script đơn giản: tìm user\_ratings và nhân giá trị

***11.7 Hàm\_scoretruy vấn* 393**

theo bản gốcđiểm và hệ số (được truyền qua các tham số bên ngoài). Chúng ta có thể xây dựng một truy vấn phức tạp dựa trên một tập lệnh đầy đủ nếu cần.

Các tập lệnh có thể tạo ra một kịch bản phức tạp với các tham số, giá trị trường và các hàm toán học (ví dụ: căn bậc hai của xếp hạng người dùng trung bình nhân với(cho hệ số tăng cường). Tuy nhiên, không phải mọi yêu cầu đều cần một tập lệnh phức tạp như vậy. Nếu yêu cầu của chúng ta chỉ là sử dụng giá trị của một trường, thì một cách đơn giản để có được kết quả là sử dụng một hàm có tên là field\_value\_factor.

* + 1. Các hệ số giá trị trường chức năng

Hàm field\_value\_factor giúp đạt được điểm bằng cách sử dụng các trường mà không cần sự phức tạp của việc viết kịch bản. Danh sách sau đây cho thấy cơ chế.

**Liệt kê 11.29 Lấy điểm từ một trường mà không cần viết mã**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "điểm\_hàm": {

"truy vấn": {

"thuật ngữ": {

"sản phẩm": "tv"

}

},

"field\_value\_factor": { "field": "user\_ratings"

**Đối tượng field\_value\_factor khai báo trường (user\_ratings, trong trường hợp này)**

}

}

}

}

Kịch bản cho thấy rằnghệ số giá trị trườngchức năng hoạt động trên một trường (đánh giá của người dùngtrong danh sách này) để tạo ra điểm số liên quan mới.

Chúng ta có thể thêm thuộc tính vào hàm field\_value\_score. Ví dụ, chúng ta có thể nhân điểm bằng cách sử dụng thuộc tính factor và áp dụng hàm toán học như căn bậc hai hoặc phép tính logarit. Danh sách tiếp theo cho thấy điều này trong hành động.

**Liệt kê 11.30 Các thuộc tính bổ sung trênhệ số giá trị trường**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "điểm\_hàm": {

"truy vấn": {

"thuật ngữ": {

"sản phẩm": "tv"

}

},

"hệ số giá trị trường": {

**394 CPHẦN11*Truy vấn hợp chất***

"trường": "user\_ratings", "hệ số": 2,

"sửa đổi": "vuông"

}

}

}

}

Tập lệnh này lấy giá trị của user\_ratings từ tài liệu. Sau đó, nó nhân giá trị đó với hệ số 2 và bình phương nó.

* + 1. Kết hợpđiểm số chức năng

Chúng ta đã xem xét các hàm riêng lẻ trong một vài phần trước, nhưng chúng ta cũng có thể kết hợp các hàm để tạo ra điểm số tốt hơn nữa. Ví dụ, danh sách sau đây hiển thị truy vấn function\_score tạo ra điểm số sử dụng hai hàm: weight và

hệ số giá trị trường.

**Liệt kê 11.31 Hai hàm tạo ra một điểm số thống nhất**

NHẬN sản phẩm/\_tìm kiếm

{

"truy vấn": { "điểm\_hàm": {

"truy vấn": {

"thuật ngữ": {

"sản phẩm": "TV"

}

},

"chức năng": [

{

"lọc": {

"thuật ngữ": {

"thương hiệu": "LG"

**Đối tượng hàm, mong đợi một mảng các hàm lá**

}

},

"trọng lượng": 3

},

{

"lọc": {

"phạm vi": {

**Cácchức năng trọng lượng**

"user\_ratings": { "gte": 4.5,

"lte": 5

}

}

},

}

"field\_value\_factor": { "field": "user\_ratings", "factor": 5, "modifier": "vuông"

}

**field\_value\_factor dựa trên trường user\_ratings.**

***Bản tóm tắt* 395**

],

"score\_mode": "trung bình", "boost\_mode": "tổng"

}

}

}

Đối tượng hàm trong truy vấn này mong đợi nhiều hàm (như weight và field\_value\_factor), kết hợp để tạo ra một điểm số thống nhất. Trường weight (hàm trọng số) mong đợi một số nguyên dương, được sử dụng trong các phép tính tiếp theo. Điểm số ban đầu của việc lấy TV bằng truy vấn thuật ngữ được bổ sung bằng các nội dung sau:

* Nếu thương hiệu là LG, hãy tăng điểm thêm 3 điểm.
* Nếu xếp hạng của người dùng nằm trong khoảng từ 4,5 đến 5, hãy sử dụngđánh giá của người dùnggiá trị của trường và bình phương nó với hệ số 5.

Khi nhiều chức năng khớp nhau, giá trị cuối cùng của điểm sẽ tăng lên. Do đó, tài liệu có thể xuất hiện ở đầu danh sách.

**Chế độ chấm điểm chức năng**

Bạn có để ý không?chế độ điểm sốVàchế độ tăng cườngcác cánh đồng tạikết thúc của kịch bản? Hai thuộc tính này củađiểm\_hàmtruy vấn cho phép chúng tôi đạt được điểm kết hợptừ truy vấn ban đầu và điểm số do một hoặc nhiều hàm đưa ra.

Theo mặc định, các điểm số do các hàm này tạo ra đều được nhân lên để có được một điểm số cuối cùng duy nhất. Tuy nhiên, chúng ta có thể thay đổi hành vi đó bằng cách đặtchế độ điểm sốthuộc tính trongđiểm\_hàmtruy vấn.chế độ điểm sốthuộc tính xác định cách tính điểm cá nhân. Ví dụ, nếuchế độ điểm sốcủa một truy vấn được thiết lập thànhtổng hợp,điểm số phát ra từ các chức năng riêng lẻ đều được cộng lại. chế độ điểm sốthuộc tính có thể là bất kỳ chế độ nào, chẳng hạn nhưnhân lên(mặc định),tổng hợp,trung bình,tối đa,phút, hoặcĐầu tiên.

Điểm từ các chức năng này sau đó sẽ được thêm vào (hoặc nhân với, hoặc tính trung bình với, v.v.) điểm ban đầu của truy vấn (trong ví dụ,thuật ngữtruy vấn tìm thấyTV) từ tài liệu dựa trênchế độ tăng cườngtham số. Cácchế độ tăng cườngtham số có thể lànhân lên(mặc định),phút,tối đa,thay thế,trung bình, hoặctổng hợp. Để tìm hiểu thêmvề các chế độ và cơ chế liên quan đến việc chấm điểm chức năng, hãy tham khảo tài liệu chính thức của Elasticsearch.

Vậy là xong! Chương này giới thiệu các truy vấn hợp chất nâng cao, hữu ích và thiết thực. Truy vấn bool là con dao đa năng của tất cả các truy vấn và giúp xây dựng các truy vấn tìm kiếm phức tạp. Trong chương tiếp theo, chúng ta sẽ xem xét các tìm kiếm nâng cao khác, chẳng hạn như tìm kiếm không gian địa lý và truy vấn tham gia.

#### Bản tóm tắt

* Truy vấn hợp chất kết hợp các truy vấn lá để tạo ra các truy vấn nâng cao đáp ứng nhiều tiêu chí tìm kiếm.

**396 CPHẦN11*Truy vấn hợp chất***

* Cácbooltruy vấn là truy vấn hợp chất phổ biến nhất, bao gồm bốn mệnh đề:phải,phải\_không,nên, Vàlọc.
* Các truy vấn trongphải\_khôngVàlọccác mệnh đề không đóng góp vào điểm liên quan tổng thể. Mặt khác, các truy vấn trongphảiVànêncác mệnh đề luôn cải thiện điểm số.
* Cácđiểm số hằng sốtruy vấn bao bọc mộtlọctruy vấn và tạo ra điểm số cố định do người dùng thiết lập.
* Cáctăng cườngtruy vấn làm tăng điểm của mộttích cựcmệnh đề trong khi ngăn chặn điểm số trên các truy vấn không khớp (tiêu cựcđiều khoản).
* Cácdis\_maxtruy vấn, được sử dụng bởinhiều\_trận\_hợptruy vấn, gói các truy vấn và thực thi chúng riêng lẻ.
* Cácđiểm\_hàmtruy vấn đặt điểm tùy chỉnh dựa trên hàm do người dùng xác định, chẳng hạn như giá trị hoặc trọng số của trường hoặc giá trị ngẫu nhiên.

*Tìm kiếm nâng cao*

***Chương này bao gồm***

* Kiểu dữ liệu địa lý
* Tìm kiếm vị trí và địa chỉ bằng geoqueries
* Sử dụnghình dạng địa lýđể tìm kiếm hình dạng 2D
* Sử dụngkhoảng cáchtruy vấn để làm việc với vị trí cấp thấpcác mã thông báo
* Các truy vấn chuyên biệt như máy pha cà phê

Các chương trước đã đề cập đến việc tìm kiếm dữ liệu bằng cách sử dụng các truy vấn cấp độ thuật ngữ và toàn văn. Chúng ta cũng đã xem xét các truy vấn nâng cao như bool, increasing và các truy vấn khác. Để tiếp tục xây dựng trên những gì chúng ta đã thảo luận và phát triển bối cảnh truy vấn, chương này giới thiệu một số loại truy vấn chuyên biệt.

Chúng ta bắt đầu bằng cách xem xét các tìm kiếm nhắm vào vị trí địa lý. Các trường hợp sử dụng phổ biến liên quan đến geoqueries bao gồm tìm kiếm nhà hàng gần đó để đặt hàng giao hàng, tìm chỉ đường đến nhà bạn bè, định vị các trường học phổ biến trong phạm vi 10 km, v.v. Elasticsearch có hỗ trợ hạng nhất để đáp ứng các tìm kiếm liên quan đến vị trí như vậy. Nó cũng cung cấp một số truy vấn không gian địa lý: geo\_bounding\_box, geo\_ distance và geo\_shape.

**397**

**398 CPHẦN12*Tìm kiếm nâng cao***

Tiếp theo, chúng ta sẽ xem xét cách tìm kiếm hình dạng hai chiều (2D) bằng cách sử dụng truy vấn hình dạng. Các kỹ sư thiết kế, nhà phát triển trò chơi và những người khác có thể tìm kiếm trong chỉ mục hình dạng 2D.Sau đó, chúng ta xem xét các truy vấn vị trí cấp thấp được gọi là truy vấn span. Mặc dù các truy vấn lá toàn văn và cấp thuật ngữ giúp chúng ta tìm kiếm dữ liệu, nhưng chúng không thể tìm thấy các từ theo thứ tự cụ thể, vị trí của chúng, khoảng cách chính xác (hoặc gần đúng) giữa các từ, v.v. Đây là nơi các truy vấn span phát huy tác dụng.

Cuối cùng, chúng tôi kết thúc chương này bằng cách xem xét các truy vấn chuyên biệt như distance\_feature, percolator, more\_like\_this và pinned. Truy vấn distance\_feature tăng cường các kết quả gần hơn với một vị trí cụ thể: ví dụ, tìm kiếm các trường học trong bán kính 10 km, nhưng ưu tiên các trường học có công viên gần đó hơn. Để thêm kết quả tìm kiếm được tìm thấy tự nhiên vào danh sách các kết quả được tài trợ, chúng tôi sử dụng các truy vấn ghim. Các truy vấn more\_like\_this tìm các tài liệu có giao diện tương tự. Truy vấn chuyên biệt cuối cùng mà chúng tôi thảo luận là truy vấn percolator, giúp thông báo cho người dùng khi có dữ liệu cho các truy vấn không mang lại kết quả trong quá khứ.

Chúng ta hãy bắt đầu bằng cách thảo luận về nhu cầu truy vấn không gian địa lý và các kiểu dữ liệu hỗ trợ chúng. Sau đó, chúng ta xem xét các truy vấn do Elasticsearch cung cấp ngay cho các tiêu chí tìm kiếm như vậy.

**GHI CHÚ**Mã cho chương này có sẵn trên GitHub ([http://mng.bz/](http://mng.bz/2D6w) [2D6w](http://mng.bz/2D6w)) và trên trang web của cuốn sách ([https://www.manning.com/books/elastic](https://www.manning.com/books/elasticsearch-in-action-second-edition) [tìm kiếm-trong-hành-động-phiên-bản-thứ-hai](https://www.manning.com/books/elasticsearch-in-action-second-edition)).

**GHI CHÚ**Không giống như các chương khác, chương này xem xét nhiều loại truy vấn bao gồm truy vấn không gian địa lý, hình dạng, khoảng cách và truy vấn chuyên biệt. Do bản chất của các truy vấn này, tôi đã chuẩn bị nhiều tập dữ liệu để đáp ứng các yêu cầu của chúng. Vì vậy, các ví dụ thay đổi giữa các tập dữ liệu (chỉ mục) trong chương này.

#### Giới thiệu tìm kiếm vị trí

Trong thời đại internet ngày nay, việc bật tìm kiếm dựa trên vị trí trong các ứng dụng và ứng dụng là một yêu cầu phổ biến. Tìm kiếm dựa trên vị trí sẽ tìm kiếm các địa điểm hoặc địa điểm dựa trên khoảng cách, chẳng hạn như nhà hàng gần đó, nhà để bán trong bán kính 1 km, v.v. Chúng tôi cũng sử dụng tìm kiếm dựa trên vị trí để tìm chỉ đường đến một địa điểm quan tâm. Tin tốt là hỗ trợ không gian địa lý là công dân hạng nhất trong Elasticsearch.

Các kiểu dữ liệu chuyên dụng cho phép chúng ta xác định một lược đồ để lập chỉ mục dữ liệu không gian địa lý, do đó cho phép tìm kiếm tập trung. Các kiểu dữ liệu sẵn có hỗ trợ dữ liệu không gian địa lý là geo\_point và geo\_shape.

Elasticsearch cũng cung cấp các truy vấn tìm kiếm không gian địa lý đủ cho hầu hết các trường hợp sử dụng,chẳng hạn như bounding\_box, geo\_distance và geo\_shape. Mỗi truy vấn này đáp ứng một tập hợp các yêu cầu mà chúng tôi thảo luận ngắn gọn trong phần này và chi tiết hơn ở phần sau của chương.

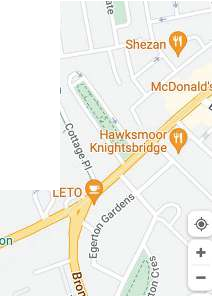
* + 1. Truy vấn bounding\_box

Đôi khi chúng ta muốn tìm danh sách các địa điểm như nhà hàng, trường học hoặc trường đại học trong một khu vực xung quanh—giả sử trong một hình vuông hoặc hình chữ nhật. Chúng ta có thể xây dựng một

* 1. ***Giới thiệu vị trítìm kiếm* 399**

hình chữ nhật, thường được gọi là hình chữ nhật địa lý, bằng cách lấy tọa độ của góc trên bên trái và góc dưới bên phải. Các tọa độ này bao gồm một cặp phép đo kinh độ và vĩ độ biểu diễn các góc.

Elasticsearch cung cấp truy vấn bounding\_box cho phép chúng ta tìm kiếm các địa chỉ cần thiết trong một hình chữ nhật địa lý. Truy vấn này lấy các điểm quan tâm (như tiêu chí truy vấn) bên trong hình chữ nhật địa lý được xây dựng bởi tập hợp tọa độ của chúng ta. Ví dụ, hình 12.1 cho thấy các địa chỉ ở trung tâm London được bao quanh trong một hình chữ nhật địa lý như vậy. Các địa chỉ giao nhau với hình chữ nhật này được trả về dưới dạng kết quả dương. Chúng ta chạy quamột số truy vấn bounding\_box chi tiết trong thời gian ngắn.



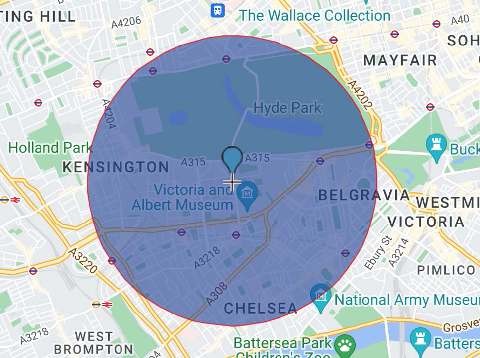
**Hình 12.1 Một hình chữ nhật địa lý được xây dựng bằng một tập hợp các tọa độ kinh độ và vĩ độ**

* + 1. Truy vấn geo\_distance

Bạn có thể đã xem những bộ phim Hollywood, trong đó một điệp viên FBI cố gắng bắt giữ một kẻ chạy trốn trong một khu vực được vẽ thành hình tròn xung quanh một điểm trung tâm. Đó chính là những gì truy vấn geo\_distance thực hiện!

Elasticsearch cung cấp truy vấn geo\_distance để lấy địa chỉ trong một khu vực được bao quanh bởi một vòng tròn. Tâm được xác định bởi kinh độ, vĩ độ và bán kính là khoảng cách. Trong hình 12.2, chúng ta có một vị trí trung tâm (được hiển thị dưới dạng ghim thả trên bản đồ) và một khu vực hình tròn bao phủ các địa chỉ mà chúng ta đang tìm kiếm. Trọng tâm (hoặc vị trí trung tâm) là một điểm trên bản đồ được xác định bởi tọa độ vĩ độ và kinh độ.

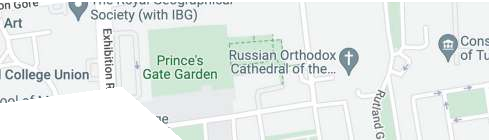
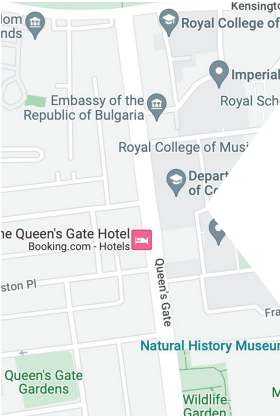
**400 CPHẦN12*Tìm kiếm nâng cao***



**Hình 12.2 Địa chỉ được bao quanh trong một vùng tròn được xây dựng bởikhoảng cách địa lýtruy vấn**

* + 1. Truy vấn geo\_shape

Truy vấn geo\_shape lấy danh sách các điểm địa lý (địa chỉ) trong một geo-envelope được xây dựng theo hình học. Envelope có thể là một hình tam giác ba cạnh hoặc một đa giác nhiều cạnh (nhưng envelope không được mở). Hình 12.3 cho thấy một envelope hình lục giác được xây dựng trên bản đồ với sáu cặp tọa độ (mỗi cặp là một geopoint có vĩ độ và kinh độ). Tìm kiếm geo\_shape tìm các vị trí bên trong đa giác này.



**Hình 12.3 Tìm địa chỉ trong một hình đa giác với mộthình dạng địa lýtruy vấn**

* 1. ***Dữ liệu không gian địa lýcác loại* 401**

[Trước khi chúng ta bắt đầu thử nghiệm với các truy vấn không gian địa lý, chúng ta cần hiểu](https://macwright.com/lonlat) [lược đồ lập bản đồ dữ liệu không gian địa lý: các kiểu dữ liệu hỗ trợ dữ liệu địa lý và](https://macwright.com/lonlat) [cơ họcđể lập chỉ mục dữ liệu đó. Trong phần tiếp theo, chúng tôi sẽ xem xét](https://macwright.com/lonlat)Đầu tiên nhập geo\_point rồi đến geo\_shape.

#### Các loại dữ liệu không gian địa lý

Tương tự như cách kiểu dữ liệu văn bản biểu diễn dữ liệu văn bản, Elasticsearch cung cấp hai kiểu dữ liệu chuyên dụng để làm việc với dữ liệu không gian: geo\_point và geo\_shape. Kiểu dữ liệu geo\_point biểu thị kinh độ và vĩ độ và hoạt động trên các truy vấn dựa trên vị trí. Mặt khác, kiểu geo\_shape cho phép chúng ta lập chỉ mục các hình dạng địa lý như điểm, nhiều đường, đa giác và một số hình dạng khác. Hãy cùng xem các kiểu dữ liệu không gian này.

* + 1. Kiểu dữ liệu geo\_point

Vị trí trên bản đồ được thể hiện phổ biến bằng kinh độ và vĩ độ. Elasticsearchhỗ trợ việc biểu diễn dữ liệu vị trí như vậy bằng cách sử dụng kiểu dữ liệu geo\_point chuyên dụng. Chúng ta đã xem xét kiểu dữ liệu geo\_point một cách ngắn gọn trong chương 4; hãy cùng tóm tắt lại cách chúng ta có thể định nghĩa một trường là geo\_point trong lược đồ lập bản đồ của mình. Khi bản đồ đã sẵn sàng, chúng ta có thể lập chỉ mục cho một tài liệu. Danh sách sau đây tạo một lược đồ dữ liệu cho chỉ mục bus\_stops với một vài trường.

**Liệt kê 12.1 Tạo một ánh xạ vớiđiểm địa lý**

ĐẶT trạm xe buýt

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

},

"vị trí":{

"kiểu": "điểm\_địa\_phương"

}

**Xác định thuộc tính vị trí là kiểu dữ liệu geo\_point**

}

}

}

Chỉ mục bus\_stops được định nghĩa với hai thuộc tính: name và location. Thuộc tính location được biểu diễn bằng kiểu dữ liệu geo\_point, nghĩa là nó phải được thiết lập với các giá trị vĩ độ và kinh độ khi lập chỉ mục tài liệu. Truy vấn sau đây lập chỉ mục trạm xe buýt London Bridge Station.

**Liệt kê 12.2 Lập chỉ mục cho một trạm xe buýt có vị trí được xác định làsợi dây**

POST bus\_stops/\_doc

{

"tên":"Trạm London Bridge", "vị trí":"51,07, 0,08"

}

**Nhập vị trí dưới dạng chuỗi**

**với các giá trị vĩ độ và kinh độ**

**402 CPHẦN12*Tìm kiếm nâng cao***

Như truy vấn cho thấy, trường vị trí có các giá trị vĩ độ và kinh độ dạng chuỗi được phân tách bằng dấu phẩy: "51.07, 0.08". Cung cấp tọa độ theo định dạng chuỗi này không phải là cách duy nhất để thiết lập trường vị trí; chúng ta có thể sử dụng một số định dạng khác để nhập tọa độ địa lý của vị trí, chẳng hạn như một mảng, một điểm văn bản nổi tiếng (WKT) hoặc một geohash.

**Liệt kê 12.3 Lập chỉ mục ở nhiều định dạng khác nhau chođiểm địa lýkiểu dữ liệu**

POST bus\_stops/\_doc

{

"text": "Ga London Victoria", "location" : "ĐIỂM (51,49 0,14)"

}

**Đầu vàogeo\_point là điểm WKT (vĩ độ, kinh độ)**

POST bus\_stops/\_doc

{

"text": "Ga Leicester Square",

}

"vị trí" : { "kinh độ":-0.12, "vĩ độ":51.50

}

**Nhập geo\_pointnhư một đối tượng vị trí**

POST bus\_stops/\_doc

{

"text": "Ga Westminster", "vị trí" : [51.54, 0.23]

}

**Nhập geo\_point dưới dạng một mảng (lon,năm)**

POST bus\_stops/\_doc

{

"text": "Ga Hyde Park", "vị trí": "gcpvh2bg7sff"

}

**Nhập geo\_point**

**như một geohash**

Các truy vấn này lập chỉ mục các vị trí trạm xe buýt khác nhau bằng nhiều định dạng. Chúng ta có thể sử dụng một chuỗi có vĩ độ và kinh độ, như trong danh sách 12.2, hoặc một đối tượng, một mảng, một geo-hash hoặc một hình dạng POINT định dạng WKT, như trong danh sách 12.3.

Bây giờ là lúc tìm hiểu về kiểu dữ liệu geo\_shape. Như tên gọi của nó, kiểu geo\_shape giúp lập chỉ mục và tìm kiếm dữ liệu bằng một hình dạng cụ thể: ví dụ, một đa giác. Hãy cùng xem cách chúng ta có thể lập chỉ mục dữ liệu cho geoshape.

* + 1. Kiểu dữ liệu geo\_shape

Giống như geo\_pointloại, đại diện cho một điểm trên bản đồ, Elasticsearch cung cấpmột kiểu dữ liệu geo\_shape để biểu diễn các hình dạng như điểm, đa điểm, đường thẳng và đa giác. Các hình dạng được biểu diễn bằng một chuẩn mở gọi là GeoJSON ([http://](http://geojson.org/) [geojson.org](http://geojson.org/)) và được viết theo định dạng JSON. Các hình dạng hình học được ánh xạ tới kiểu dữ liệu geo\_shape.

* 1. ***Dữ liệu không gian địa lýcác loại* 403**

Hãy tạo bản đồ cho một chỉ mục các quán cà phê với một vài trường. Một trongđó là trường địa chỉ, trỏ đến vị trí của một quán cà phê, được biểu thị dưới dạng kiểu geo\_shape.

**Liệt kê 12.4 Tạo bản đồ vớihình dạng địa lýloại trường**

Quán cà phê PUT

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

},

"Địa chỉ": {

"kiểu": "hình dạng địa lý"

}

**Đặt loại địa chỉ là geo\_shape**

}

}

}

Mã tạo ra một chỉ mục được gọi là quán cà phê để chứa các nhà hàng địa phương. Trường đáng chú ý làaddress, được định nghĩa là kiểu geo\_shape. Kiểu này hiện mong đợi đầu vào là hình dạng trong Geo-JSON hoặc WKT. Ví dụ, để biểu diễn một điểm trên bản đồ, chúng ta có thể nhập trường bằng Point trong GeoJSON hoặc POINT trong WKT.

**Liệt kê 12.5 Nhập mộthình dạng địa lýsử dụng định dạng WKT và GeoJSON**

PUT quán cà phê/\_doc/1

{

"tên":"Costa Coffee", "địa chỉ" : {

"loại": "Điểm",

**Nhập địa chỉ theo định dạng GeoJSON**

**Đặt loại địa chỉ là geo\_shape**

"tọa độ" : [0,17, 51,57]

}

}

**Nhập địa chỉ**

**Tọa độ (kinh độ và vĩ độ) biểu diễn Điểm**

ĐẶT /cafes/\_doc/2

{

**theo định dạng WKT**

**Đặt loại địa chỉ là**

"địa chỉ" : "ĐIỂM (0,17 51,57)"

}

**ĐIỂM theo định dạng WKT**

Mã này khai báo hai cách để nhậphình dạng địa lýtrường: GeoJSON và WKT. GeoJSONmong đợi mộtkiểuthuộc tính của một hình dạng thích hợp ("type":"Điểm") và các tọa độ tương ứng("tọa độ":[0,17, 51,57]), như trong ví dụ đầu tiên. Ví dụ thứ hai trong danh sách 12.5 cho thấy cơ chế tạo điểm bằng định dạng WKT ("địa chỉ": "ĐIỂM (0,17 51,57)").

**GHI CHÚ** Có một sự khác biệt tinh tế khi biểu diễn tọa độ bằng định dạng chuỗi so với các định dạng khác. Định dạng chuỗi mong đợi các giá trị theo thứ tự vĩ độ theo sau là kinh độ, được phân tách bằng dấu phẩy: ví dụ:

**404 CPHẦN12*Tìm kiếm nâng cao***

"(51,57, 0,17)". Tuy nhiên, các tọa độ được hoán đổi cho các định dạng Geo-JSON và WKT dưới dạng kinh độ theo sau là vĩ độ: ví dụ,"ĐIỂM (0,17 51,57)".

Chúng ta có thể xây dựng nhiều hình dạng khác nhau bằng các định dạng này. Bảng 12.1 cung cấp mô tả ngắn gọn về một số định dạng trong số đó. Xem tài liệu Elasticsearch về cách lập chỉ mục và tìm kiếm tài liệu để hiểu chi tiết các khái niệm và ví dụ.

**Bảng 12.1 Các hình dạng được hỗ trợ bởihình dạng địa lýkiểu dữ liệu**

|  |  |  |  |
| --- | --- | --- | --- |
| **Hình dạng** | **Sự miêu tả** | **Địa lý**  **đại diện** | **Đại diện WKT** |
| Điểm | Một điểm được biểu diễntheo vĩ độ và kinh độ | Điểm | ĐIỂM |
| Đa điểm | Một mảng các điểm | Đa điểm | ĐA ĐIỂM |
| Đa giác | Hình dạng nhiều cạnh | Đa giác | ĐA GIÁC |
| Đa giác | Danh sách nhiều đa giác | Đa đa giác | ĐA ĐA GIÁC |
| Dòng chuỗi | Một đường thẳng giữa hai điểm | DòngChuỗi | CHUỖI DÒNG |
| Chuỗi nhiều dòng | Danh sách các chuỗi nhiều dòng | Chuỗi nhiều dòng | ĐA DÒNG CHUỖI |

Bây giờ chúng ta đã biết cách làm việc với phần lập chỉ mục của geodata bằng cách sử dụng geo\_pointvà các trường geo\_shape, chúng ta đã sẵn sàng tìm kiếm tài liệu của mình. Chúng ta sẽ thảo luận về điều này trong phần tiếp theo.

#### Truy vấn không gian địa lý

Chúng ta cần lập chỉ mục dữ liệu nhà hàng London cho các ví dụ trong phần này. Dữ liệu có sẵn trong thư mục datasets trên GitHub (<http://mng.bz/1q5R>) và trên trang web của cuốn sách.

Để định vị dữ liệu không gian địa lý, nhiệm vụ tiếp theo của chúng ta là tìm kiếm tài liệu cho một tiêu chuẩn địa lý nhất định. Ví dụ, nơi bạn mua cà phê được biểu thị bằng kinh độ và vĩ độ, đến lượt nó được gọi là một điểm trên bản đồ.

Một ví dụ khác, tôi có thể tìm kiếm các nhà hàng gần nhà mình nhất, trả về mỗi quán cà phê phù hợp được biểu diễn dưới dạng một điểm. Hoặc một lô đất trên bản đồ có thể được biểu diễn bằng một hình dạng đại diện cho một quốc gia hoặc sân chơi của trường học địa phương của bạn.

Elasticsearch cung cấp một bộ truy vấn không gian địa lý phù hợp cụ thể để tìm kiếm các trường hợp sử dụng này (chẳng hạn như tìm địa chỉ gần đó hoặc tìm kiếm tất cả các điểm thú vị trong một khu vực nhất định). Chúng tôi xem xét các truy vấn sau trong các phần sau:

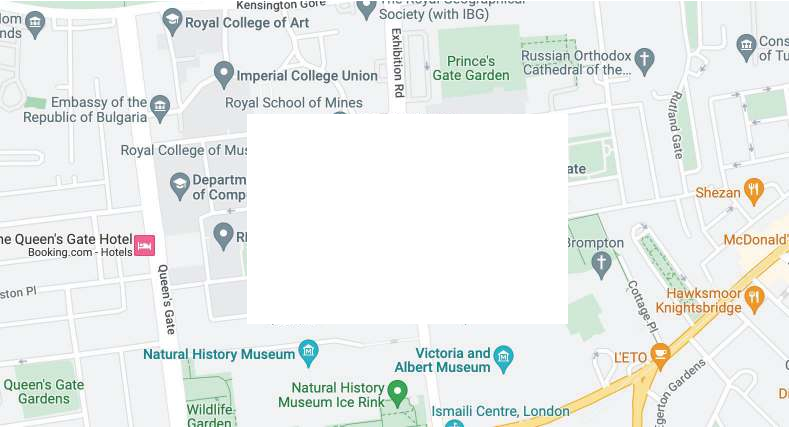
* hộp giới hạn địa lý*truy vấn*—Tìm các tài liệu được bao bọc trong một hình chữ nhật được xây dựng bởi các điểm địa lý; ví dụ, tất cả các nhà hàng nằm bên trong một hình chữ nhật địa lý
* khoảng cách địa lý*truy vấn*—Tìm địa chỉ trong một khoảng cách nhất định từ một điểm; ví dụ, tất cả các máy ATM trong phạm vi 1 km tính từ Cầu London
  1. ***Hộp giới hạn địa lýtruy vấn* 405**
  + hình dạng địa lý*truy vấn*—Tìm các địa chỉ được biểu diễn dưới dạng các hình dạng trong một hình dạng được xây dựng bởi một tập hợp các tọa độ; ví dụ, các trang trại nông nghiệp nằm trong vành đai xanh, trong đó các trang trại và vành đai xanh được biểu diễn bằng các hình dạng riêng lẻ cũng như vành đai xanh

Chúng tôi thảo luận chi tiết về các truy vấn cho cả điểm địa lý và hình dạng địa lý trong phần sau.tions. Hãy bắt đầu với các truy vấn được thực hiện trên các trường geo\_point.

#### 12.4 Truy vấn geo\_bounding\_box

Khi chúng ta tìm kiếm danh sách địa chỉ, chúng ta có thể sử dụng một khu vực quan tâm. Khu vực này có thể được biểu diễn bằng một hình tròn có bán kính nhất định, một khu vực được bao quanh bởi một hình dạng như hình chữ nhật, hoặc một hình đa giác từ một điểm trung tâm (một điểm mốc).

Elasticsearch cung cấp truy vấn geo\_bounding\_box cho phép chúng ta tìm kiếm các vị trí bên trong các khu vực này. Ví dụ, như hình 12.4 cho thấy, chúng ta có thể xây dựng một hình chữ nhật sử dụng tọa độ vĩ độ và kinh độ và tìm kiếm để xem địa chỉ của chúng ta có tồn tại trong khu vực đó không.



**Hộp giới hạn địa lý**

(x1, y1)

**Tọa độ góc dưới bên phải (kinh độ, vĩ độ)**

**Tọa độ góc trên bên trái (kinh độ, vĩ độ)**

(x2, y2)

**Hình 12.4 Hình chữ nhật địa lý từ tọa độ vĩ độ và kinh độ**

Các trường top\_left và bottom\_right là tọa độ vĩ độ và kinh độ tạo nên hình chữ nhật địa lý của chúng ta. Khi chúng ta xác định một hình chữ nhật địa lý, chúng ta có thể kiểm tra xem các điểm quan tâm (ví dụ như Imperial College London) có nằm trong hình chữ nhật này không.

Trước khi thảo luận chi tiết về truy vấn geo\_bounding\_box, hãy viết truy vấn; sau đó chúng ta có thể phân tích nó. Truy vấn sau đây tìm kiếm tất cả các tài liệu (vị trí) phù hợp với hình chữ nhật được xây dựng với tọa độ top\_left và bottom\_right.

**406 CPHẦN12*Tìm kiếm nâng cao***

**Liệt kê 12.6 Vị trí nhà hàng phù hợp trong mộthình chữ nhật địa lý**

NHẬN nhà hàng/\_tìm kiếm

{

"truy vấn": { "hộp giới hạn địa lý": {

"vị trí": {

"trên cùng bên trái": {

"vĩ độ": 52,

"lớn": 0,2

},

"bottom\_right": {["vĩ độ": 49,](https://www.elastic.co/guide/en/elasticsearch/reference/current/span-queries.html)

"lớn": 0,1

}

}

}

}

}

**Xây dựng một**

**hình chữ nhật địa lý Thiết lập tài liệu**

**trường geo\_point**

**Xác định điểm top\_left, được hình thành bởi cặp vĩ độ/kinh độ**

**Xác định điểm bottom\_right, được hình thành bởi cặp vĩ độ/kinh độ**

Truy vấn tìm kiếm tất cả các tài liệu giao nhau (phù hợp với) một hình chữ nhật địa lý được tạo bởihai tọa độ top\_left và bottom\_right, như trong hình 12.4. Người dùng có thể cung cấp hai tọa độ này để chúng ta có thể xây dựng một hình chữ nhật. Các nhà hàng bên trong hình chữ nhật này được trả về dưới dạng kết quả tìm kiếm và phần còn lại sẽ bị loại bỏ.

**GHI CHÚ**Chúng ta cũng có thể biểu diễn các đỉnh của hình chữ nhật bị chặn nhưtrên cùng bên phảiVàdưới cùng bên trái(trái ngược vớitrên cùng bên tráiVàdưới cùng bên phải). Hoặc chúng ta có thể chia nhỏ nó hơn nữa bằng cách sử dụng tọa độ được đặt tênđứng đầu,bên trái,đáy, VàPhải.

Làm việc với dữ liệu hình dạng địa lý



Chúng tôi đã thực hiệnhộp giới hạn địa lýtruy vấn trong danh sách 12.6 trên lược đồ với các trường được khai báo làđiểm địa lýkiểu dữ liệu. Nhưng chúng ta có thể sử dụng cùng một truy vấn nếu tài liệu của chúng tabao gồm một trường được định nghĩa làhình dạng địa lý, thay vào đó? Chúng ta đã học trong phần 12.3.3 rằng dữ liệu không gian địa lý cũng có thể được biểu diễn bằng cách sử dụnghình dạng địa lýkiểu dữ liệu, nhớ không?

Chúng ta có thể sử dụng truy vấn từ danh sách 12.6 cho dữ liệu hình dạng địa lý, ngoại trừ việc chúng ta phải hoán đổi URL trỏ đến chỉ mục thích hợp. Ví dụ, vớiquán cà phêchỉ số và địa lýdữ liệu hình dạng trong tay, tất cả những gì chúng ta cần làm là xây dựng cùng mộthộp giới hạn địa lýtruy vấn nhưng thay đổi URL để phản ánh chỉ mục hình dạng địa lý (quán cà phê):

GET quán cà phê/\_tìm kiếm

{

"truy vấn": { "hộp giới hạn địa lý": {

"Địa chỉ": {

"trên cùng bên trái": {

"vĩ độ": 52,

"lớn": 0,04

**Chỉ mục quán cà phê với các trường dữ liệu kiểu geo\_shape**

**Trường địa chỉ được định nghĩa là kiểu dữ liệu geo\_shape.**

},

"bottom\_right": {

* 1. ***Hộp giới hạn địa lýtruy vấn* 407**

"vĩ độ": 49,

"lớn": 0,2

}

}

}

}

}

Truy vấn này tìm kiếm các quán cà phê trong một hình chữ nhật địa lý nhất định được xây dựng bằngtrên cùng bên tráiVàdưới cùng bên phảitham số. Ở đây, chúng tôi sử dụngquán cà phêchỉ mục trong URL (LẤY) khi gọihộp giới hạn địa lýtruy vấn. Ngoài việc hoán đổi chỉ số geopoints(nhà hàng)với một chỉ mục bao gồm các hình dạng địa lý (quán cà phê), không có sự khác biệt nào trong truy vấn!

Chúng tôi cung cấp các giá trị kinh độ và vĩ độ dưới dạng các đối tượng trong danh sách 12.6. Tuy nhiên, latude và longitude có thể được thiết lập với nhiều định dạng: dưới dạng một mảng hoặc các giá trị WKT. Ví dụ, các truy vấn geo\_bounding\_box trước đó cung cấp các thuộc tính top\_left và bottom\_right dưới dạng các đối tượng "lat" và "lon":

"trên cùng bên trái": {

"vĩ độ": 52.00,

"lớn": 0,20

}

Thay vào đó, chúng ta có thể đặt kinh độ và vĩ độ thành một mảng. Nhưng đây là một điều cần lưu ýsider: các giá trị trong mảng phải được đảo ngược. Chúng phải là lon rồi đến lat (trái ngược với lat rồi đến lon trong các ví dụ trước).

**GHI CHÚ**Để biết thêm thông tin về sự không nhất quán lon-lat này trong các mô-đun phần mềm không gian địa lý, hãy xem bài viết của Tom MacWright tại[https://macwright.com/lonlat.](https://macwright.com/lonlat)

Danh sách sau đây hiển thị cùng một truy vấn geo\_bounding\_box, nhưng lần này kinh độ và vĩ độ (được tô đậm) được cung cấp dưới dạng một mảng.

**Liệt kê 12.7 Geoquery với một geopoint được chỉ định là một mảng**

NHẬN nhà hàng/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [

{

"match\_all": {}

}

],

"lọc": [

{

"geo\_bounding\_box": { "vị trí": {

**"trên cùng bên trái": [0, 52.00]**,

**Thuộc tính top\_left với các giá trị kinh độ và vĩ độ**

**408 CPHẦN12*Tìm kiếm nâng cao***

**"bottom\_right": [0.10, 49]**

}

}

}

]

}

}

}

**Thuộc tính bottom\_right với các giá trị kinh độ và vĩ độ**

Chúng tôi định nghĩatrên cùng bên tráiVàdưới cùng bên phảithuộc tính dưới dạng một mảng gồm hai điểm địa lý: kinh độ và vĩ độ.

Chúng ta cũng có thể cung cấp kinh độ và vĩ độ dưới dạng đối tượng vectơ. WKT là ngôn ngữ đánh dấu văn bản chuẩn để biểu diễn các đối tượng vectơ trên bản đồ. Ví dụ, đểbiểu diễn một điểm trong WKT, chúng ta viết POINT(10, 20), biểu diễn một điểm trên bản đồ có tọa độ x và y lần lượt là 10 và 20. Elasticsearch cung cấp đánh dấu WKT cho các truy vấn hộp giới hạn dưới dạng BBOX với các giá trị tương ứng. Truy vấn sau đây cho thấy điều này.

**Liệt kê 12.8 Geoquery với vị trí được biểu diễn là WKT**

NHẬN nhà hàng/\_tìm kiếm

{

"truy vấn": {

"bool": {

"phải": [

{

"match\_all": {}

}

],

"lọc": [

{

"geo\_bounding\_box": { "vị trí": {

"wkt":"BBOX(0,08, 0,04, 52,00, 49,00)"

}

**Đặt tọa độ theo định dạng WKT**

}

}

]

}

}

}

Trường vị trí của bộ lọc geo\_bounding\_box chấp nhận tọa độ là BBOX với các giá trị kinh độ và vĩ độ tương ứng. BBOX tạo một hình chữ nhật địa lý từ cặp này để tạo ra các điểm trên cùng bên trái và dưới cùng bên phải.

Ngoài sở thích của riêng bạn, không có sự khác biệt nào giữa định dạng WKT và mảng. Nếu bạn đang xây dựng một ứng dụng sử dụng các tiêu chuẩn WKT cho geodata, thì việc sử dụng lập chỉ mục và tìm kiếm dựa trên WKT trong Elasticsearch là hợp lý.

Trong phần này, chúng ta đã học cách tìm vị trí bên trong một hình chữ nhật địa lý bằng cách sử dụnggeo\_bounding\_box query. Đôi khi chúng ta có thể muốn tìm nhà hàng gần một vị trí trung tâm: ví dụ, tất cả các nhà hàng cách trung tâm thành phố 10 km. Đây là nơi chúng ta có thể sử dụng

* 1. ***Khoảng cách địa lýtruy vấn* 409**

truy vấn geo\_distance. Truy vấn này lấy tất cả các vị trí khả dụng trong một vòng tròn có điểm hội tụ ở giữa. Chúng ta sẽ thảo luận chi tiết về truy vấn geo\_distance sau.

#### Truy vấn geo\_distance

Khi chúng ta muốn tìm danh sách các địa chỉ xung quanh một điểm trung tâm, truy vấn geo\_distance sẽ hữu ích. Truy vấn này hoạt động bằng cách khoanh tròn một khu vực có bán kính bằng một khoảng cách nhất định từ một điểm tiêu cự. Ví dụ, như hình 12.5 cho thấy, chúng ta có thể muốn tìm các trường học gần đó trong bán kính 10 km.



R = 10 km

R = 10 km

**Hình 12.5 Trở lại trường học với mộtkhoảng cách địa lýtruy vấn**

Hãy xem truy vấn geo\_distance đang hoạt động. Danh sách sau đây định nghĩa mộtTruy vấn geo\_distance lấy tất cả các nhà hàng trong phạm vi 175 km tính từ tọa độ trung tâm đã cho.

**Danh sách 12.9 Tìm kiếm nhà hàng trong bán kính nhất định**

NHẬN nhà hàng/\_tìm kiếm

{

"truy vấn": { "khoảng cách địa lý": {

"khoảng cách": "175 km",

**Tuyên bố**

**truy vấn geo\_distance**

**Thiết lập vùng lân cận của một khu vực để tìm kiếm (khoảng cách từ điểm trung tâm)**

**410 CPHẦN12*Tìm kiếm nâng cao***

"vị trí": {

"vĩ độ": 50.00,

"lớn": 0,10

**Đặt vị trí trung tâm, được xác định là một điểm trên bản đồ**

}

}

}

}

Như danh sách cho thấy, truy vấn geo\_distance mong đợi hai thuộc tính: distance, cung cấp bán kính của geocircle và location, xác định điểm trung tâm của geocircle. Truy vấn trả về tất cả các nhà hàng trong phạm vi 175 km tính từ điểm đã xác định.

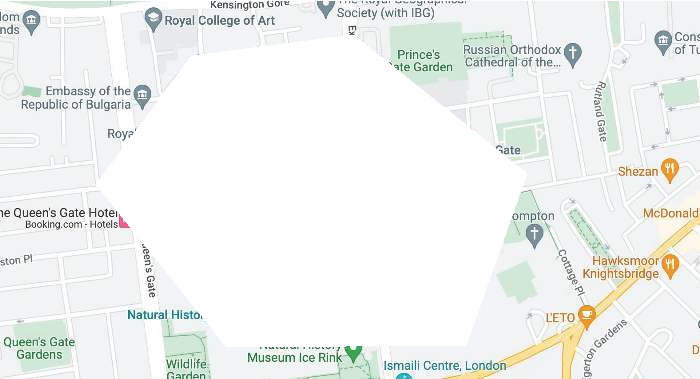
**GHI CHÚ**Trường khoảng cách chấp nhận khoảng cách được đo bằng kilômét hoặcdặm theo km hoặc mi. Elasticsearch cũng chấp nhận giá trị được đưa ra là "350 mi" cũng như "350mi" (đã xóa khoảng trắng).

Bạn sẽ không ngạc nhiên khi biết rằng trường để xác định điểm trung tâm có thể được nhập bằng vĩ độ và kinh độ dưới dạng chuỗi, mảng, WKT và các định dạng khác. Truy vấn cũng có thể chạy trên geoshapes, mặc dù tôi để bạn tự thử nghiệm với chúng.Các truy vấn geo\_bounding\_box và geo\_distance cho phép chúng ta tìm kiếm các địa chỉ được biểu diễn dưới dạng các vị trí dựa trên điểm trong các hình chữ nhật và hình tròn. Tuy nhiên, chúng ta thường cần tìm kiếm một địa chỉ được xác định là một hình dạng bên trong một hình dạng khác, tốt nhất là một

đa giác. Đây là nơi chúng ta sử dụnghình dạng địa lýtruy vấn.

#### Truy vấn geo\_shape

Không phải tất cả các vị trí đều dựa trên điểm (tọa độ với các giá trị vĩ độ và kinh độ). Đôi khi chúng ta muốn xác định xem các hình dạng có nằm bên trong (hoặc bên ngoài) ranh giới của một hình dạng khác hay chúng có giao nhau với ranh giới hay không. Ví dụ, hình 12.6 cho thấy một số lô đất trên bản đồ London của chúng tôi.



Cốt truyện C

Cốt truyện D

Cốt truyện B

Cốt truyện A

**Hình 12.6 Các lô đất nông nghiệp ở một khu vực của London**

* 1. ***Hình dạng địa lýtruy vấn* 411**

Trong hình, geoshape của chúng ta được biểu diễn bằng hình lục giác. Các ô A và B nằm trong ranh giới của hình này, ô C giao với hình lục giác và ô D nằm ngoài ranh giới của geoshape. Elasticsearch cung cấp truy vấn geo\_shape để tìm kiếm các ô có nhiều hình dạng khác nhau này trong một đường bao được xây dựng theo tọa độ.

Truy vấn geo\_shape lấy các vị trí hoặc địa chỉ được biểu diễn dưới dạng hình dạng trong một hình dạng khác, chẳng hạn như hình lục giác trong hình 12.6. Chúng tôi xây dựng hình lục giác này bằng cách cung cấp tọa độ dưới dạng giá trị cho trường bao thư.

Của chúng tôicafes index chứa một vài tài liệu có hình dạng dựa trên điểm, vì vậy hãy viết truy vấn để xem cách lấy dữ liệu này. (Nếu bạn tò mò, hãy lập chỉ mục thêm một vài tài liệu có hình dạng khác nữa.) Truy vấn trong danh sách sau sẽ tìm kiếm tất cả các quán cà phê trong hình dạng được xác định bởi trường envelope với các cặp kinh độ và vĩ độ.

**Danh sách 12.10 Tìm kiếm các quán cà phê trong một hình dạng nhất định**

GET quán cà phê/\_tìm kiếm

{

"truy vấn": {

"hình dạng địa lý": {

"Địa chỉ": {

"hình dạng": {

"loại": "phong bì", "tọa độ": [

[0,1,55],

[1,45]

]

},

"quan hệ": "trong"

}

}

}

}

**Xác định truy vấn geo\_shape**

**mong đợi một lĩnh vực và một hình dạng**

**Đặt loại thuộc tính hình dạng mong đợi một đường bao được xây dựng bằng cách sử dụng tọa độ**

**Xác định mối quan hệ giữa đường bao và hình dạng địa lý kết quả**

Truy vấn sẽ lấy các tài liệu (quán cà phê) nằm trong phong bì được xây dựng bởi cặp giá trị kinh độ và vĩ độ đã cho. Trong trường hợp này, tìm kiếm sẽ trả về quán cà phê Costa Coffee được tìm thấy trong phong bì.

Một điều cuối cùng chúng ta cần hiểu là thuộc tính quan hệ. Thuộc tính quan hệ xác định mối quan hệ của các tài liệu cần tìm cho hình dạng đã cho. Giá trị mặc định của quan hệ là giao nhau, nghĩa là truy vấn trả về các tài liệu giao nhau với hình dạng đã cho. Bảng 12.2 mô tả các giá trị có thể có cho thuộc tính quan hệ.

**Bảng 12.2 Mối quan hệ giữa tài liệu và hình dạng phong bì**

|  |  |
| --- | --- |
| **Giá trị củamối quan hệ** | **Sự miêu tả** |
| giao nhau (mặc định)  ở trong | Trả về các tài liệu giao với hình dạng hình học đã cho  Phù hợp với các tài liệu nếu chúng tồn tại trong ranh giới của hình dạng hình học đã cho |

**412 CPHẦN12*Tìm kiếm nâng cao***

**Bảng 12.2 Mối quan hệ giữa các tài liệu và hình dạng phong bì (tiếp theo)**

|  |  |
| --- | --- |
| **Giá trị củamối quan hệ** | **Sự miêu tả** |
| chứa  phân chia | Trả về các tài liệu nếu chúng chứa hình dạng hình học  Trả về các tài liệu không tồn tại trong hình dạng hình học đã cho |

Trong hình 12.6, nếu chúng ta đặt cụ thể relation=intersects, các đồ thị mong đợi là A, B và C (vì C giao với đường bao chính). Nếu chúng ta đặt relation=within, các đồ thị A và B được trả về vì chúng nằm trong đường bao bị giới hạn. Đặt relation= contains trả về các đồ thị A và B vì chúng nằm trong đường bao; và không có gì ngạc nhiên khi đồ thị D là kết quả của relation=disjoint.

Cho đến nay, chúng ta đã xem xét các truy vấn không gian địa lý: các truy vấn trên geodata được biểu diễn bằng các trường geo\_point và geo\_shape. Các truy vấn này cho phép chúng ta tìm kiếm dữ liệu trên bản đồ cho nhiều trường hợp sử dụng khác nhau. Bây giờ, hãy chuyển hướng và thảo luận về các truy vấn để tìm kiếm các hình dạng hai chiều (2D). Trong phần tiếp theo, chúng ta sẽ khám phá chi tiết các truy vấn hình dạng.

#### Truy vấn hình dạng

Chúng tôi xây dựng các hình dạng 2D như đường thẳng, điểm và đa giác bằng cách sử dụng tọa độ Descartes x và y. Elasticsearch cung cấp chức năng lập chỉ mục và tìm kiếm các đối tượng 2D bằng cách sử dụng truy vấn hình dạng. Ví dụ, dữ liệu bản thiết kế của kỹ sư xây dựng, thiết kế CAD (thiết kế hỗ trợ máy tính) của người vận hành máy và các dữ liệu khác phù hợp với tiêu chí này. Trong phần này, chúng tôi sẽ lập chỉ mục và tìm kiếm các hình dạng địa lý một cách ngắn gọn.

Khi làm việc với dữ liệu 2D, chúng tôi sử dụng kiểu dữ liệu hình dạng chuyên dụng. Chúng tôi tạo các trường có kiểu hình dạng này để lập chỉ mục và tìm kiếm dữ liệu 2D. Điều này dễ hiểu khi xem qua các ví dụ sau.

Truy vấn sau đây hiển thị bản đồ cho myshapes index có hai thuộc tính, name và myshape. Như bạn có thể thấy, thuộc tính myshape được định nghĩa là kiểu dữ liệu shape.

**Liệt kê 12.11 Ánh xạ chỉ mục vớihình dạngkiểu**

ĐẶT myshapes

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

},

"hình dạng của tôi":{

"kiểu": "hình dạng"

}

}

}

}

Bây giờ chúng ta đã có bản đồ, bước tiếp theo là lập chỉ mục các tài liệu có hình dạng khác nhau. Truy vấn sau đây lập chỉ mục hai tài liệu có hình dạng điểm và đường.

* 1. ***Hình dạngtruy vấn* 413**

**Liệt kê 12.12 Lập chỉ mục hình dạng điểm và đa điểm**

ĐẶT myshapes/\_doc/1

{

"name":"Một điểm hình dạng", "myshape":{

"kiểu":"điểm", "tọa độ":[12,14]

**Đánh dấu một điểm**

}

}

ĐẶT myshapes/\_doc/2

{

"name":"Một hình dạng đa điểm", "myshape":{

"loại":"đa điểm",

**Chỉ số**

**một đa điểm**

"tọa độ":[[10,13],[13,16]]

}

}

Mã này đặt một điểm và một đa điểm, cả hai đều là hình dạng 2D, vào chỉ mục myshapes. Hình 12.7 hiển thị hai hình dạng: đường thẳng và điểm.



**Một phong bì**

Một hình dạng đường thẳng

**Hình 12.7 Tìm kiếm hình dạng 2D trong một đường bao giới hạn**

**414 CPHẦN12*Tìm kiếm nâng cao***

Chúng ta có thể tìm kiếm tất cả các tài liệu để tìm các hình dạng nằm trong hình dạng địa lý được bao gồm trong phong bì. Truy vấn sau đây hiển thị mã cho tìm kiếm này.

**Liệt kê 12.13 Tìm kiếm tất cả các hình dạng trong một phong bì nhất định**

NHẬN myshapes/\_search

{

**Hình dạng chúng ta**

"truy vấn": {

"hình dạng":{

"hình dạng của tôi": {

"hình dạng": {

**Chỉ định một**

**hình dạngtruy vấn Trường mà**

**truy vấn được chạy**

**muốn xây dựng**

}

}

"loại":"phong bì",

"tọa độ":[[10,16],[14,10]]

}

}

}

**Đường bao được xây dựng bởi các tọa độ đã cho**

Truy vấn hình dạng được xác định trong danh sách tìm kiếm các tài liệu có trong phong bì được tạo từ các tọa độ đã cho [10,16],[14,10]. (Tham khảo hình 12.7 để xem phong bì có giới hạn được tạo bởi truy vấn này và các hình dạng có trong đó.)

Như truy vấn trong danh sách 12.13 cho thấy, chúng ta có thể tạo một đường bao đa giác bằng cách sử dụng các tọa độ cần thiết (hãy đảm bảo các tọa độ cuối trùng nhau, vì Elasticsearch không hỗ trợ đa giác mở). Hãy nhớ rằng các truy vấn hình dạng hữu ích khi làm việc với tọa độ Descartes 2D cho bản vẽ và thiết kế.

Bây giờ chúng ta chuyển sang một tập hợp các truy vấn hoàn toàn khác trong phạm vi các truy vấn chuyên biệt: truy vấn span. Các truy vấn này hỗ trợ tìm kiếm các thuật ngữ tại một vị trí cụ thể trong tài liệu, không giống như các truy vấn tìm kiếm thông thường, trong đó vị trí của mã thông báo bị bỏ qua. Điều này được hiểu rõ nhất bằng cách chạy một số truy vấn; chúng ta thực hiện điều này tiếp theo.

#### Truy vấn khoảng

Truy vấn cấp độ thuật ngữ và toàn văn bản hoạt động theo phép thuật tìm kiếm ở cấp độ mã thông báo (từ). Chúng không tập trung vào vị trí của các mã thông báo (từ) hoặc thứ tự của chúng. Hãy xem xét văn bản sau, một trích dẫn của Isaac Newton:

*Plato là bạn tôi. Aristotle là bạn tôi. Nhưng người bạn vĩ đại nhất của tôi là sự thật.*

Giả sử chúng ta muốn tìm một tài liệu (trích dẫn) trong đó Plato và Aristotle đều được đề cập theo cùng một thứ tự (không phải Aristotle và Plato), và từ Aristotle phải cách Plato ít nhất bốn vị trí. Hình 12.8 cho thấy mối quan hệ này: Plato ở vị trí 1, Aristotle ở vị trí 5 và khoảng cách của họ là 5. Yêu cầu của chúng ta là lấy tất cả các trích dẫn trong đó Plato và Aristotle đáp ứng thông số kỹ thuật này. Chúng ta không thể đáp ứng yêu cầu này bằng cách sử dụng các truy vấn toàn văn (hoặc cấp thuật ngữ). Mặc dù truy vấn tiền tố có thể thực hiện công lý một chút cho yêu cầu này, nhưng nó không thể đáp ứng các tiêu chí phức tạp khác mà chúng ta sẽ xem xét trong các phần tiếp theo.

***12.8 Khoảng cáchtruy vấn* 415**

*Plato là bạn tôi. Aristotle là bạn tôi. Nhưng người bạn lớn nhất của tôi là sự thật*

***Isaac Newton***

*~~là bạn của tôi~~*

Plato

Aristoteles

**Vị trí 1**

Khoảng cách = 5

**Vị trí 5**

**Hình 12.8**

**Tìm một trích dẫn bằng cách sử dụngmột truy vấn vị trí**

Ví dụ này cho thấy lý do tại sao truy vấn span lại hữu ích. Chúng là các truy vấn cấp thấp giúp chúng ta tìm các tài liệu có mã thông báo được chỉ định theo vị trí. Khi làm việc với các tài liệu pháp lý, bài báo nghiên cứu hoặc sách kỹ thuật, trong đó các câu có vị trí chính xác của các từ là bắt buộc, chúng ta có thể sử dụng truy vấn span. Có một số loại truy vấn span—span\_first, span\_within, span\_near, v.v.—và chúng ta sẽ xem qua một vài loại trong số chúng ở đây; như thường lệ, hãy tham khảo tài liệu để tìm hiểu thêm.

* + 1. Dữ liệu mẫu

Trước khi làm việc với truy vấn span, hãy chuẩn bị Elasticsearch bằng chỉ mục trích dẫn và một vài tài liệu.

**Liệt kê 12.14 Chuẩn bị Elasticsearch cho mộtkhoảng cáchtruy vấn**

PUT trích dẫn

{

"ánh xạ": { "thuộc tính": {

"tác giả":{

**Tạo một chỉ mục trích dẫn với một vài thuộc tính**

"loại": "văn bản"

},

"trích dẫn":{

"loại": "văn bản"

}

}

}

}

**Chỉ số**

PUT trích dẫn/\_doc/1

{

**Trích dẫn của Newton**

"tác giả":"Isaac Newton",

"trích dẫn":"Plato là bạn tôi. Aristotle là bạn tôi.

➥Nhưng người bạn lớn nhất của tôi chính là sự thật."

}

Chúng tôi tạo chỉ mục trích dẫn với một vài thuộc tính: tác giả và trích dẫn, cả hai đều là trường văn bản ("type": "text"). Chúng tôi cũng lập chỉ mục trích dẫn của Isaac Newton như được định nghĩa trong danh sách. Bây giờ chúng tôi đã chuẩn bị chỉ mục trích dẫn, hãy xem một số truy vấn span đang hoạt động, bắt đầu với truy vấn span\_first.

**416 CPHẦN12*Tìm kiếm nâng cao***

* + 1. Khoảng cách đầu tiêntruy vấn

Giả sử chúng ta muốn tìm một từ cụ thể trong n số token đầu tiên. Ví dụ, chúng ta muốn biết Aristotle có tồn tại ở năm vị trí đầu tiên trong tài liệu của chúng ta không (xem hình 12.9).

**Chức vụ5**

**Chức vụ1 Chức vụ6**

*là*

*Plato*

Khoảng cách = 5

Aristoteles

*bạn bè*

*là*

*Của tôi*

**Hình 12.9 Tìm kiếm một tài liệu có thuật ngữ trong số n mã thông báo đầu tiên**

Như bạn có thể suy ra từ hình, Aristotle ở vị trí thứ năm và do đó tồn tại ở năm vị trí đầu tiên. Trường hợp sử dụng này có thể được đáp ứng bằng truy vấn span\_first. Chúng ta thấy truy vấn đang hoạt động trong danh sách sau.

**Liệt kê 12.15 Tìm kiếm thuật ngữ đã cho ở năm vị trí đầu tiên**

NHẬN trích dẫn/\_tìm kiếm

{

"truy vấn": { "span\_first": {

"cuộc thi đấu": {

"khoảng thời gian": {

**Lấy tài liệu trong**

**n số nhịp đầu tiên**

**Thuật ngữ chúng tôi muốn**

"trích dẫn": "aristotle"

}

},

**để tìm kiếm**

"kết thúc": 5

}

}

}

**Vị trí đầu tiên trong n vị trí để tìm một sự khớp**

Truy vấn span\_first mong đợi một truy vấn khớp trong đó chúng tôi cung cấp các truy vấn span khác. Ở đây, truy vấn span\_term được gói trong truy vấn span\_first. Mặc dù truy vấn span\_term tương đương với truy vấn term, nhưng nó thường được gói trong các khối truy vấn span khác. Thuộc tính end chỉ ra số lượng vị trí tối đa được phép khi tìm kiếm thuật ngữ khớp từ đầu trường (trong trường hợp này, end là 5).

**GHI CHÚ**Tên thuộc tính end hơi khó hiểu. Đây là vị trí cuối của mã thông báo được phép khi tìm kiếm các thuật ngữ khớp. Tôi nghĩ tên n\_position\_from\_beginning sẽ phù hợp hơn.

Trong danh sách 12.15, chúng tôi tìm kiếm Aristotle ở năm vị trí đầu tiên từ đầu trích dẫn (xem hình 12.9). Vì vị trí của Aristotle là thứ năm, nên tài liệu bao gồm

***12.8 Khoảng cáchtruy vấn* 417**

trích dẫn được trả về thành công. Nếu chúng ta thay đổi thuộc tính kết thúc thành bất kỳ giá trị nào nhỏ hơn 5, truy vấn sẽ không trả về thuật ngữ tìm kiếm. Truy vấn trả về thành công nếu giá trị cho thuộc tính kết thúc lớn hơn 5 (6, 7, 10, v.v.).

* + 1. Truy vấn span\_near

Trong truy vấn span\_first, từ truy vấn luôn được đếm từ vị trí bắt đầu (vị trí 1). Thay vì tìm từ nếu nó tồn tại ở n vị trí đầu tiên, đôi khi chúng ta muốn tìm những từ gần nhau hơn. Ví dụ, tiếp tục với trích dẫn của Newton, giả sử chúng ta muốn xác định xem các từ Plato và Aristotle có nằm cạnh nhau hay không hoặc cách nhau không quá ba hoặc bốn vị trí. Như hình 12.10 minh họa, chúng ta thu được kết quả dương nếu chúng ta đang tìm kiếm một trích dẫn mà Plato và Aristotle khác nhau ba vị trí.

Chênh lệch vị trí = 3

*là*

*Của tôi*

Plato

Aristoteles

*bạn bè*

**Hình 12.10 Từ dự kiếnở một khoảng cách xác định**

**Chức vụ1 Chức vụ5**

Ngoài việc tìm ra sự khác biệt về khoảng cách giữa các từ này, chúng ta có thể muốn chúng tồn tại theo cùng một thứ tự. Danh sách tiếp theo hiển thị truy vấn để xác định xem Plato và Aristotle có gần nhau không (cách nhau khoảng ba vị trí).

**Liệt kê 12.16 Tìm kiếm tài liệu có các thuật ngữ gần nhau**

NHẬN trích dẫn/\_tìm kiếm

{

"truy vấn": {

"khoảng\_gần": {

"các điều khoản": [

{

"khoảng thời gian": {

**Định nghĩa truy vấn span\_near**

**với một vài mệnh đề**

**Các mệnh đề bao gồm hai span\_term độc lập để tìm kiếm các từ riêng lẻ**

"trích dẫn": "plato"

}

},

{

"khoảng thời gian": {

"trích dẫn": "aristotle"

}

} **Thuộc tính slop cung cấp**

], **số lượng chấp nhận được**

"slop": 3, "in\_order": đúng

}

}

}

**vị trí giữa các từ.**

**Thuộc tính in\_order thiết lập thứ tự của các thuộc tính.**

**418 CPHẦN12*Tìm kiếm nâng cao***

Truy vấn span\_near chấp nhận nhiều mệnh đề; chúng ta cũng có các truy vấn span\_term cố gắng khớp các thuật ngữ của chúng ta. Ngoài ra, vì chúng ta biết hai từ này cách nhau ba vị trí, chúng ta có thể cung cấp sự khác biệt này dưới dạng thuộc tính slop. Thuộc tính slop cho phép sự khác biệt vị trí tối đa có thể chấp nhận được giữa các từ. Ví dụ, truy vấn chỉ định rằng hai từ có thể được tách biệt tối đa ba vị trí. Chúng ta có thể tăng slop (ví dụ, "slop": 10) để giảm các ràng buộc nghiêm ngặt để truy vấn có nhiều khả năng thành công hơn. Tuy nhiên, các truy vấn span rất tốt trong việc tìm các từ chính xác ở các vị trí chính xác, vì vậy hãy tăng thuộc tính slop một cách thận trọng.

Ngoài việc sử dụng thuộc tính slop, chúng ta có thể định nghĩa thứ tự của các từ. Tiếp tục với cùng ví dụ, nếu thứ tự không quan trọng, chúng ta có thể trả về kết quả dươngngay cả khi chúng ta yêu cầu span\_near giữa Aristotle và Plato (thay vì Plato và Aristotle). Nếu thứ tự quan trọng, chúng ta có thể đặt Boolean cho cờ in\_order. Thuộc tính in\_order có thể được đặt thành true hoặc false; khi được đặt thành true, như trong danh sách 12.16, thứ tự mà các từ được lập chỉ mục sẽ được tính đến.

* + 1. Truy vấn span\_within

Trường hợp sử dụng tiếp theo cho truy vấn span là khi chúng ta muốn tìm một từ giữa hai từ. Ví dụ, giả sử chúng ta muốn tìm các tài liệu mà Aristotle nằm giữa hai từ: friend và friend, như thể hiện trong hình 12.11.

Plato là của tôi**bạn bè**. Aristotle là của tôi**bạn bè**Nhưng người bạn lớn nhất của tôi là sự thật

**Hình 12.11 Tìm một từ nếu nó tồn tại giữa các từ khác**

Chúng ta có thể sử dụng kiểu truy vấn span\_within cho mục đích này. Hãy xem xét truy vấn trong danh sách sau và sau đó phân tích nó.

**Liệt kê 12.17 Tìm kiếm một từ giữa các từ khác**

NHẬN trích dẫn/\_tìm kiếm

{

"truy vấn": { "khoảng\_trong": {

"nhỏ bé": {

"khoảng thời gian": {

**Một truy vấn span\_within bao gồm**

**của hai khối: nhỏ và lớn**

**Xác định từ tìm kiếm**

"trích dẫn": "aristotle"

}

},

"to lớn": {

"khoảng\_gần": {

"các điều khoản": [

{

**Bao quanh khối nhỏ**

* 1. ***Truy vấn khoảng***

"khoảng thời gian": {

"trích dẫn": "bạn bè"

**419**

}

},

{

"khoảng thời gian": {

"trích dẫn": "bạn bè"

}

}

],

"slop": 4, "in\_order": đúng

}

}

}

}

}

Cái nàykhoảng\_trongtruy vấn bao gồm hai khối,nhỏ béVàto lớn. Cácnhỏ békhối trong tìm kiếm mong đợi được bao gồm trongto lớnkhối.Trong truy vấn này, chúng tôi muốn tìm các tài liệu trong đó Aristotle được bao gồm giữa hai từ friend và friend, được định nghĩa trongto lớnkhối.

Hãy nhớ rằng, khối lớn không gì khác ngoài truy vấn span\_near (chúng ta đã thảo luận về truy vấn này ở phần trước). Không có giới hạn nào về số lượng mệnh đề chúng ta có thể có trong khối lớn. Ví dụ, chúng ta có thể mở rộng truy vấn trong danh sách 12.17 với danh sách tiếp theo, có ba mệnh đề, mỗi mệnh đề tìm kiếm từ friend.

**Liệt kê 12.18 Kiểm tra xem một từ có tồn tại trong một tập hợp hay khôngtừ**

NHẬN trích dẫn/\_tìm kiếm

{

"truy vấn": { "khoảng\_trong": {

"nhỏ bé": {

"span\_term": { "trích dẫn": "aristotle"

}

},

"to lớn": {

"khoảng\_gần": {

"các điều khoản": [

{

"khoảng thời gian": {

"trích dẫn": "bạn bè"

}

},

{

"khoảng thời gian": {

"trích dẫn": "bạn bè"

}

},

{

**420 CPHẦN12*Tìm kiếm nâng cao***

"khoảng thời gian": {

"trích dẫn": "bạn bè"

}

}

],

"slop": 10, "in\_order": đúng

}

}

}

}

}

**Tăng giá trị của thuộc tính slop**

Truy vấn này hiện đang cố gắng xác định xem Aristotle có nằm giữa một tập hợp các từ (tất cả đều là bạn) không được định nghĩa trong khối lớn. Thay đổi đáng chú ý là tăng giá trị slop. Do đó, truy vấn span\_within giúp chúng ta xác định một truy vấn trong một truy vấn khác.

* + 1. Truy vấn span\_or

Truy vấn span cuối cùng mà chúng ta xem xét thỏa mãn điều kiện OR, trả về kết quả khớp với một hoặc nhiều tiêu chí đầu vào. Elasticsearch cung cấp truy vấn span\_or cho mục đích này. Nó tìm các tài liệu khớp với một hoặc nhiều truy vấn span từ một tập hợp các mệnh đề nhất định. Ví dụ, truy vấn sau đây tìm các tài liệu khớp với Plato hoặc Aristotle nhưng bỏ qua từ friends (lưu ý số nhiều; tài liệu của chúng ta chứa từ friend trong trường trích dẫn, không phải friends).

**Liệt kê 12.19 Tìm kiếm bất kỳ từ nào phù hợp**

NHẬN trích dẫn/\_tìm kiếm

{

"truy vấn": {

"khoảng\_hoặc": {

"các điều khoản": [

{

"khoảng thời gian": {

**Xác định**

**truy vấn span\_or**

**Liệt kê nhiều mệnh đề**

"trích dẫn": "plato"

}

},

{

"span\_term": { "quote": "bạn bè"

}

},

{

"span\_term": { "trích dẫn": "aristotle"

}

}

]

}

}

}

* 1. ***Chuyêntruy vấn* 421**

Truy vấn span\_or này lấy tài liệu có trích dẫn của Newton vì nó khớp với cả Plato và Aristotle. Lưu ý rằng truy vấn friends không khớp, nhưng vì toán tử là OR, truy vấn sẽ vui vẻ tiếp tục vì ít nhất một trong các từ được truy vấn khớp. Truy vấn không thất bại, mặc dù từ friends không khớp.

Elastic search có các truy vấn span khác như span\_not, span\_taining, span\_ multi\_term và các truy vấn khác, nhưng thật không may, chúng tôi không thể thảo luận tất cả các loại này ở đây. Tôi khuyên bạn nên tham khảo tài liệu để hiểu rõ hơn về các truy vấn này. Tài liệu có sẵn tại đây:<http://mng.bz/QPa4>.

Phần tiếp theo sẽ đề cập đến các truy vấn chuyên biệt nhưkhoảng cách\_tính năng,máy lọc nướcvà những thứ khác. Bây giờ chúng ta hãy chú ý đến những thứ đó.

#### Các truy vấn chuyên biệt

Ngoài các loại truy vấn mà chúng ta đã thấy cho đến nay, Elasticsearch có một số truy vấn nâng cao dành riêng cho việc phục vụ các chức năng chuyên biệt: ví dụ, tăng điểm cho các quán cà phê phục vụ đồ uống lạnh tại một địa điểm cụ thể (truy vấn distance\_feature), cảnh báo người dùng khi một mặt hàng có hàng trở lại (truy vấn percolate), tìm các tài liệu có giao diện tương tự (truy vấn more\_like\_this), tăng tầm quan trọng của tài liệu (truy vấn ghim), v.v. Chúng tôi dành phần cuối của chương này để xem xét chi tiết các truy vấn chuyên biệt này.

* + 1. Truy vấn distance\_feature

Khi tìm kiếm tác phẩm văn học kinh điển, giả sử chúng ta muốn thêm một mệnh đề để tìm những cuốn sách được xuất bản vào năm 1813. Cùng với việc trả về tất cả các cuốn sách là tác phẩm văn học kinh điển, chúng ta mong đợi tìm thấy Kiêu hãnh và Định kiến ​​(tác phẩm kinh điển của Jane Austen), nhưng chúng ta muốn đưa Kiêu hãnh và Định kiến ​​lên đầu danh sách vì nó được in vào năm 1813. Đứng đầu danh sách có nghĩa là tăng điểm liên quan của kết quả truy vấn dựa trên một mệnh đề cụ thể; trong trường hợp này, chúng ta đặc biệt muốn những cuốn sách được xuất bản vào năm 1813 được coi trọng hơn. Điều nàychức năng có sẵn trong Elasticsearch bằng cách sử dụng truy vấn distance\_feature. Truy vấn sẽ lấy kết quả và đánh dấu chúng bằng điểm liên quan cao hơn nếu chúng gần với ngày gốc hơn (1813, trong ví dụ này).

Truy vấn distance\_featurecung cấp hỗ trợ tương tự cho các vị trí. Chúng ta có thể làm nổi bật các vị trí gần một địa chỉ cụ thể và đưa chúng lên đầu danh sách nếu chúng ta muốn. Giả sử chúng ta muốn tìm tất cả các nhà hàng phục vụ cá và khoai tây chiên, nhưng những nhà hàng đứng đầu danh sách phải ở gần Chợ Borough gần Cầu London. (Chợ Borough là một chợ thực phẩm thủ công nổi tiếng thế giới vào thế kỷ thứ mười ba; xem[https://boroughmarket](https://boroughmarket.org.uk/)

[.org.uk](https://boroughmarket.org.uk/).)

Chúng ta có thể sử dụng truy vấn distance\_feature cho các trường hợp sử dụng như vậy, để tìm kết quả gần vị trí hoặc ngày gốc. Ngày và vị trí là các trường được khai báo là kiểu dữ liệu date (hoặc date\_nanos) và geo\_point. Kết quả gần với ngày hoặc vị trí nhất định được đánh giá cao hơn về điểm liên quan. Hãy xem một số ví dụ để hiểu khái niệm này một cách chi tiết.

**422 CPHẦN12*Tìm kiếm nâng cao***

**BĐIỂM SỐ OOSTING CHO CÁC TRƯỜNG ĐẠI HỌC GẦN ĐÂY SỬ DỤNG ĐỊA LÝ**

Giả sử chúng ta đang tìm kiếm các trường đại học ở Vương quốc Anh. Trong khi tìm kiếm, chúng ta muốn ưu tiên các trường đại học trong bán kính 10 km tính từ Cầu London. Chúng ta tăng điểm cho các trường này.

Để thử kịch bản này, hãy tạo một bản đồ cho chỉ mục trường đại học với vị trí được khai báo là trường geo\_point. Danh sách sau đây tạo bản đồ và chỉ mục cho bốn trường đại học: hai trường ở London và hai trường khác trong cả nước.

**Liệt kê 12.20 Tạo mộtcác trường đại họcchỉ số**

Các trường đại học PUT

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

},

"vị trí":{

"kiểu": "điểm\_địa\_phương"

}

}

}

}

PUT các trường đại học/\_doc/1

{

"tên":"Trường Kinh tế London (LSE)", "vị trí":[0,1165, 51,5144]

}

PUT các trường đại học/\_doc/2

{

"tên":"Cao đẳng Hoàng gia London", "vị trí":[0,1749, 51,4988]

}

PUT các trường đại học/\_doc/3

{

"tên":"Đại học Oxford", "vị trí":[1.2544, 51.7548]

}

PUT các trường đại học/\_doc/4

{

"tên":"Đại học Cambridge", "vị trí":[0,1132, 52,2054]

}

Bây giờ chỉ mục và dữ liệu đã được chuẩn bị, hãy lấy các trường đại học, tăng điểm liên quan để những trường gần Cầu London hơn sẽ ở đầu danh sách. Bản đồ London trong hình 12.12 cho thấy khoảng cách gần đúng của các trường đại học này từ Cầu London và Knightsbridge. Chúng tôi sử dụng truy vấn distance\_feature cho mục đích này

***12.9 Chuyên ngànhtruy vấn* 423**

mục đích, phù hợp với tiêu chí truy vấn nhưng tăng điểm liên quan dựa trên các tham số bổ sung được cung cấp.

**Cầu Knightsbridge**



**Cao đẳng Hoàng gia**

**Trường London**

**Kinh tế (LSE)**

**Cầu London**

**Hình 12.12 Bản đồ London hiển thị các trường đại học gần Cầu London**

Hãy viết truy vấn và sau đó đào sâu vào để tìm hiểu chi tiết. Danh sách sau đây sử dụng truy vấn distance\_feature trong truy vấn bool để tìm các trường đại học.

**Danh sách 12.21 Tăng điểm số của các trường đại học gần London hơnCầu**

GET các trường đại học/\_tìm kiếm

{

**Vị trí**

"truy vấn": { "tính năng khoảng cách": {

"trường": "vị trí",

**Tính năng khoảng cách**

**khai báo truy vấn**

**để tìm kiếm**

"nguồn gốc": [-0,0860, 51,5048],

**Điểm hội tụ nơi khoảng cách**

"trục": "10 km"

}

}

}

**Khoảng cách từ tiêu điểm**

**được đo từ một nguồn gốc**

Truy vấn tìm kiếm tất cả các trường đại học và trả về hai trường: Trường Kinh tế London và Cao đẳng Hoàng gia London. Ngoài ra, nếu bất kỳ trường đại học nào nằm trong phạm vi 10 km xung quanh gốc tọa độ (-0,0860, 51,5048 biểu thị Cầu London ở Vương quốc Anh), thì các trường này được chấm điểm cao hơn các trường khác.

Cáckhoảng cách\_tính năngtruy vấn, như được định nghĩa trong danh sách, mong đợi các thuộc tính sau:

* cánh đồng—Cácđiểm địa lýtrường trong tài liệu

**424 CPHẦN12*Tìm kiếm nâng cao***

* nguồn gốc—Điểm hội tụ (theo kinh độ và vĩ độ) để đo khoảng cách
* trục xoay—Khoảng cách từ tiêu điểm

Trong danh sách 12.21, Trường Kinh tế London gần Cầu London hơn là Cao đẳng Imperial, do đó Trường Kinh tế London được trả về ở đầu kết quả vớiđiểm cao hơn. Bây giờ chúng ta hãy xem xét việc sử dụng truy vấn distance\_feature với ngày tháng.

**B****TÍNH ĐIỂM BẰNG CÁCH SỬ DỤNG NGÀY**

Trong phần trước, truy vấn distance\_feature đã giúp chúng tôi tìm kiếm các trường đại học, tăng điểm cho những trường gần một vị trí địa lý nhất định. Yêu cầu tương tự cũng có thể được đáp ứng với distance\_feature: tăng điểm của kết quả nếu chúng gần với một ngày.

Giả sử chúng ta muốn tìm kiếm tất cả các ngày phát hành iPhone, đứng đầu danh sách với các iPhone được phát hành trong vòng 30 ngày vào khoảng ngày 1 tháng 12 năm 2020 (không vì lý do cụ thể nào khác ngoài việc thử nghiệm khái niệm). Chúng ta có thể viết một truy vấn tương tự như danh sách 12.21,ngoại trừ thuộc tính trường dựa trên ngày. Trước tiên, hãy tạo một bản đồ iPhone và lập chỉ mục cho một vài iPhone.

**Liệt kê 12.22 Tạo mộtiphonelập chỉ mục và chuẩn bị nó với các tài liệu**

ĐẶT iPhone

{

"ánh xạ": { "thuộc tính": {

"tên":{

"loại": "văn bản"

},

"ngày phát hành":{ "loại": "ngày",

"định dạng": "dd-MM-yyyy"

}

}

}

}

**Chỉ số một vài**

ĐẶT iphones/\_doc/1

{

**tài liệu**

"tên":"iPhone", "ngày phát hành":"29-06-2007"

}

ĐẶT iphones/\_doc/2

{

"tên":"iPhone 12", "ngày phát hành":"23-10-2020"

}

ĐẶT iphones/\_doc/3

{

"tên":"iPhone 13", "ngày phát hành":"24-09-2021"

}

***12.9 Chuyên ngànhtruy vấn* 425**

ĐẶT iphones/\_doc/4

{

"tên":"iPhone 12 Mini", "ngày phát hành":"13-11-2020"

}

Bây giờ chúng ta đã có chỉ mục chứa một số iPhone, hãy phát triển truy vấn để đáp ứng yêu cầu của mình: chúng ta muốn lấy tất cả iPhone nhưng ưu tiên những chiếc được phát hành trong 30 ngày kể từ ngày 1 tháng 12 năm 2020. Truy vấn trong danh sách tiếp theo sẽ thực hiện điều này.

**Liệt kê 12.23 Lấy iPhone và tăng cườngxếp hạng**

NHẬN iphones/\_search

{

"truy vấn": {

"bool": {

"phải": [

{

"cuộc thi đấu": {

"tên": "12"

}

}

],

"nên": [

{

**Định nghĩacái**

"distance\_feature": { "field": "ngày phát hành", "origin": "1-12-2020",

**Lĩnh vực mà chống lại**

**truy vấn của chúng tôi thực hiện**

**ngày trọng tâm**

}

]

}

"trục": "30 ngày"

}

**Xoay số ngày để tăng điểm**

}

}

Trong truy vấn này, chúng tôi gói distance\_feature trong truy vấn bool với mệnh đề must và mệnh đề should (chúng tôi đã thảo luận về truy vấn bool trong chương 11). Mệnh đề must tìm kiếm tất cả các tài liệu có 12 trong trường name và trả về các tài liệu iPhone 12 và iPhone 12 mini từ chỉ mục của chúng tôi. Yêu cầu của chúng tôi là ưu tiên các điện thoại được phát hành trong 30 ngày vào khoảng ngày đầu tiên của tháng 12 (có khả năng là tất cả các điện thoại được phát hành vào tháng 11 và tháng 12 năm 2020).

Để đáp ứng yêu cầu này, mệnh đề should sử dụng truy vấn distance\_feature để tăng điểm của các tài liệu khớp gần nhất với ngày trục. Truy vấn sẽ lấy tất cả các tài liệu từ chỉ mục iphones. Bất kỳ iPhone nào được phát hành 30 ngày trước hoặc sau ngày 1 tháng 12 năm 2020 (ngày gốc) đều được trả về với điểm liên quan cao hơn.

Hãy nhớ rằng các trận đấunênmệnh đề trả về thêm vào tổng điểm. Vì vậy, iPhone 12 Mini nên đứng đầu danh sách vì ngày phát hành ("ngày phát hành": "13-11-2020") của iPhone này gần với ngày trục ("nguồn gốc":"01-12-2020"

± 30 ngày). Kết quả của truy vấn được hiển thị ở đây để đầy đủ:

**426 CPHẦN12*Tìm kiếm nâng cao***

"lượt truy cập" : [

{

"\_index" : "iPhone",

"\_id" : "4",

"\_score" : 1.1876879, "\_source" : {

"tên": "iPhone 12 Mini", "ngày phát hành": "13-11-2020"

}

},

{

"\_index" : "iPhone",

"\_id" : "2",

"\_score" : 1.1217185, "\_source" : {

"tên": "iPhone 12", "ngày phát hành": "23-10-2020"

}

}

]

iPhone 12 Mini đạt điểm cao hơn iPhone 12 vì nó được phát hành trước ngày ra mắt 17 ngày, trong khi iPhone 12 được phát hành sớm hơn gần năm tuần.

* + 1. Truy vấn được ghim

Bạn có thể đã thấy kết quả tìm kiếm được tài trợ ở đầu tập kết quả khi truy vấn trang web thương mại điện tử yêu thích của mình, chẳng hạn như Amazon. Giả sử chúng ta muốn triển khai chức năng như vậy trong ứng dụng của mình bằng Elasticsearch. Đừng lo lắng: truy vấn được ghim đã có trong tầm tay.

Truy vấn ghim sẽ thêm các tài liệu đã chọn vào tập kết quả để chúng xuất hiện ở đầu danh sách. Truy vấn này thực hiện bằng cách làm cho điểm liên quan của chúng cao hơn các điểm khác. Truy vấn ví dụ trong danh sách sau đây cho thấy chức năng này.

**Liệt kê 12.24 Sửa đổi kết quả tìm kiếm bằng cách thêm tài trợkết quả**

NHẬN iphones/\_search

{

"truy vấn": {

"ghim":{

**Chỉ định**

**ghimtruy vấn Danh sách ID tài liệu được chấm điểm cao hơn**

**Thực hiện tìm kiếm truy vấn**

"id":["1","3"],

"hữu cơ":{

"trận đấu":{ "tên":"iPhone 12"

**hơn phần còn lại của kết quả**

**Truy vấn khớp tìm kiếm iPhone 12**

}

}

}

}

}

Truy vấn được ghim này có một số phần chuyển động. Trước tiên, hãy xem khối hữu cơ.

khối truy vấn chứa truy vấn tìm kiếm; trong trường hợp này, chúng tôi đang tìm kiếm iPhone 12 trong

***12.9 Chuyên ngànhtruy vấn* 427**

chỉ mục iphones của chúng tôi. Về mặt lý tưởng, truy vấn này sẽ trả về các tài liệu iPhone 12 và iPhone 12 Mini. Tuy nhiên, đầu ra bao gồm hai tài liệu (iPhone và iPhone 13) ngoài iPhone 12 và iPhone 12 Mini. Lý do là trường ids. Trường này bao gồm các tài liệu bổ sung phải được thêm vào kết quả và hiển thị ở đầu danh sách (kết quả được tài trợ), do đó tạo ra điểm liên quan cao hơn một cách tổng hợp.

Đã ghimtruy vấn thêm các tài liệu có mức độ ưu tiên cao vào các tập kết quả. Các tài liệu này sẽ đánh bại các tài liệu khác trong vị trí danh sách để tạo ra các kết quả được tài trợ.

Bạn có thể tự hỏi liệu các kết quả ghim có điểm không: một hoặc nhiều kết quả ghim có thể được ưu tiên hơn các kết quả khác không? Thật không may, câu trả lời là không. Các tài liệu này được trình bày theo thứ tự ID làm đầu vào trong truy vấn: "ids":["1","3"], chẳng hạn.

* + 1. Truy vấn more\_like\_this

Bạn có thể đã thấy Netflix hoặc Amazon Prime Video (hoặc ứng dụng phát trực tuyến yêu thích của bạn) hiển thị cho bạn các bộ phim More Like This khi bạn duyệt. Ví dụ, hình 12.13 hiển thị các bộ phim More Like This khi tôi truy cập Paddington 2.

Một trong những yêu cầu đối với người dùng là tìm kiếm "tương tự" hoặc "giống" trong tài liệu: ví dụ, tìm các trích dẫn tương tự như "Friends and Truth" của Newton, nghiên cứu các bài báo về COVID và SARS hoặc truy vấn các bộ phim như The Godfather. Hãy cùng xem một ví dụ để hiểu rõ hơn về trường hợp sử dụng.

Giả sử chúng ta đang thu thập một danh sách các hồ sơ về nhiều người khác nhau. Để tạo hồ sơ, chúng ta lập chỉ mục các tài liệu mẫu vàochỉ mục hồ sơ.



**Hình 12.13 Xem Thêm Những Thứ Như Thế Nàyphim**

**Liệt kê 12.25 Hồ sơ mẫu lập chỉ mục**

ĐẶT hồ sơ/\_doc/1

{

"tên":"John Smith",

"profile":"John Smith là một thợ mộc có năng lực"

}

ĐẶT hồ sơ/\_doc/2

{

"tên":"John Smith Patterson",

"profile":"John Smith Patterson là một thợ sửa ống nước xinh đẹp"

}

**428 CPHẦN12*Tìm kiếm nâng cao***

ĐẶT hồ sơ/\_doc/3

{

"tên":"Smith Sotherby",

"profile":"Smith Sotherby là một họa sĩ nhẹ nhàng"

}

ĐẶT hồ sơ/\_doc/4

{

"name":"Frances Sotherby", "profile":"Frances Sotherby là một quý ông"

}

Không có gì đáng ngạc nhiên về những tài liệu này; chúng chỉ là hồ sơ của những người bình thường. Bây giờ các tài liệu này đã được lập chỉ mục, hãy cùng tìm hiểu cách chúng ta có thể yêu cầu Elastic-search tìm kiếm các tài liệu tương tự như văn bản “gentle painter” hoặc “capable carpenter”, hoặc thậm chí tìm kiếm các tài liệu có tên tương tự như Sotherby. Đó là những gì truy vấn more\_ like\_this giúp chúng ta thực hiện. Danh sách tiếp theo tạo ra một truy vấn để tìm kiếm các hồ sơ giống Sotherby hơn.

**Danh sách 12.26 Tìm kiếm thêm nội dung tương tựtài liệu**

NHẬN hồ sơ/\_tìm kiếm

{

"truy vấn": { "more\_like\_this": {

**Xác định**

**nhiều hơn như thế nàytruy vấn Tìm kiếm đã cho**

**Xác định tiêu chí truy vấn**

"fields": ["name", "profile"], "like": "Sotherby", "min\_term\_freq": 1,

"max\_query\_terms": 12, "min\_doc\_freq": 1



**nhập vào các trường**

**Đặt tần suất thuật ngữ (mặc định là 2)**

} **Thiết lập số lượng**

} **các điều khoản được lựa chọn**

}

Truy vấn more\_like\_this chấp nhận văn bản trong tham số like và văn bản đầu vào này được khớp với các trường có trong tham số fields. Truy vấn chấp nhận một số tham số điều chỉnh, chẳng hạn như tần suất thuật ngữ và tài liệu tối thiểu (min\_term) và số lượng thuật ngữ tối đa (max\_query\_terms) mà truy vấn nên chọn. Nếu chúng ta muốn cung cấp cho người dùng trải nghiệm tốt hơn khi hiển thị các tài liệu tương tự, truy vấn more\_like\_this là lựa chọn phù hợp.

* + 1. Truy vấn thấm lọc

Việc tìm kiếm tài liệu khi nhập dữ liệu rất đơn giản. Tất cả những gì chúng ta cần làm là trả về kết quả tìm kiếm từ chỉ mục nếu có bất kỳ kết quả nào khớp với tiêu chí đã cho. Điều này đáp ứng yêu cầu tìm kiếm theo tiêu chí của người dùng và đây là những gì chúng ta đã làm cho đến nay khi truy vấn kết quả.

Elasticsearch còn đáp ứng một yêu cầu khác: thông báo cho người dùng khi tìm kiếm hiện tại của họ cho kết quả âm nhưng kết quả sẽ có vào một ngày trong tương lai. Ví dụ, giả sử người dùng tìm kiếm sách Python in Action trên trang web bán sách thương mại điện tử của chúng tôi, nhưng thật không may, chúng tôi không có sách trong kho.

***12.9 Chuyên ngànhtruy vấn* 429**

khách hàng không hài lòng rời khỏi trang web. Tuy nhiên, chúng tôi nhận được hàng mới sau một hoặc hai ngày và cuốn sách được thêm vào kho. Bây giờ, vì cuốn sách xuất hiện trở lại trong kho của chúng tôi, chúng tôi muốn thông báo cho người dùng để họ có thể mua nó.

Elasticsearch hỗ trợ trường hợp sử dụng này bằng cách cung cấp một truy vấn percolate đặc biệt, sử dụng loại trường percolator. Truy vấn percolate ngược lại với cơ chế truy vấn tìm kiếm thông thường của chúng tôi ở chỗ thay vì chạy truy vấn đối với các tài liệu, chúng tôi tìm kiếm một truy vấn được cung cấp một tài liệu. Thoạt nhìn đây là một khái niệm lạ, nhưng chúng tôi sẽ giải thích rõ hơn trong phần này. Hình 12.14 cho thấy sự khác biệt giữa truy vấn thông thường và truy vấn percolate.

**Lập chỉ mục một tài liệu Chỉ mục bình thường**

PUT nhân viên/\_doc/1

{

"tên":"Jack Johnson", "tuổi": 34

}

**Tìm kiếm tài liệu bằng cách gửi truy vấn**

NHẬN nhân viên/\_tìm kiếm

{

"truy vấn":{..}

}

**Tìm kiếm một truy vấn**

**bằng cách gửi mộttài liệu**

người lao động

**Lập chỉ mục truy vấn**

nhân viên\_máy pha cà phê

**Chỉ số của máy pha cà phê**

NHẬN nhân viên/\_tìm kiếm

{

"truy vấn":{..}

}

NHẬN nhân viên/\_tìm kiếm

{

"truy vấn": {

...

"tài liệu": {

"tên":"Jack"

}

}

}

**Hình 12.14 Bình thường so vớithấm quatruy vấn**

Chúng ta hãy xem các truy vấn percolate hoạt động bằng cách lập chỉ mục một số tài liệu trước. Danh sách sau đây lập chỉ mục ba cuốn sách kỹ thuật vào chỉ mục tech\_books. Lưu ý rằng chúng tôi chưa bao gồm một cuốn sách Python nào.

**Liệt kê 12.27 Kỹ thuật lập chỉ mụcsách**

ĐẶT tech\_books/\_doc/1

{

"name":"Java hiệu quả",

"thẻ":["Java","Kỹ thuật phần mềm", "Lập trình"]

}

ĐẶT tech\_books/\_doc/2

{

**430 CPHẦN12*Tìm kiếm nâng cao***

"name":"Khóa học cấp tốc về Elasticsearch", "tags":["Elasticsearch","Kỹ thuật phần mềm", "Lập trình"]

}

ĐẶT tech\_books/\_doc/3

{

"name":"Cơ bản về Java Core", "tags":["Java","Phần mềm"]

}

Bây giờ chúng ta đã gieo hạt giống cho chỉ mục kiểm kê sách của mình, người dùng có thể tìm kiếm sách bằng các truy vấn khớp/thuật ngữ đơn giản. (Tôi bỏ qua các truy vấn này ở đây vì chúng ta đã thành thạo chúng.) Tuy nhiên, không phải tất cả các truy vấn của người dùng đều mang lại kết quả; ví dụ, một người tìm kiếm Python in Action sẽ không tìm thấy nó. Danh sách sau đây cho thấy điều này.

**Liệt kê 12.28 Tìm kiếm một cái không tồn tạisách**

NHẬN tech\_books/\_search

{

"truy vấn": {

"cuộc thi đấu": {

"tên": "Python"

}

}

}

Theo quan điểm của người dùng, tìm kiếm kết thúc với kết quả đáng thất vọng: không trả về cuốn sách họ đã tìm kiếm. Chúng ta có thể đưa truy vấn của mình lên cấp độ tiếp theo bằng cách thông báo cho người dùng khi sách Python hết hàng có sẵn. Đây là nơi chúng ta có thể đưa bộ lọc vào hoạt động.

Cũng giống như chúng ta lập chỉ mục các tài liệu vào một chỉ mục, một percolator có thể có một chỉ mục cho một tập hợp các truy vấn và mong đợi các truy vấn được lập chỉ mục. Chúng ta cần xác định một lược đồ cho một chỉ mục percolator. Hãy gọi nó là tech\_books\_percolator.

**Liệt kê 12.29 Tạo mộtmáy lọc nướcchỉ số**

ĐẶT tech\_books\_percolator

{

"ánh xạ": { "thuộc tính": {

"truy vấn": {

**Thiết lập trường**

**tên đểtruy vấn Đặt trường truy vấn**

"loại": "máy pha cà phê"

},

**kiểu dữ liệu cho máy phay**

"tên": {

"loại": "văn bản"

},

"thẻ": {

"loại": "văn bản"

}

}

}

}

**Đặt trường tên như nó xuất hiện trong chỉ mục tech\_books**

**Đặt trường thẻ như nó xuất hiện trong chỉ mục tech\_books**

***12.9 Chuyên ngànhtruy vấn* 431**

Danh sách này xác định chỉ mục để lưu trữ các truy vấn của bộ lọc. Lưu ý những điều sau:

* Nó chứa mộttruy vấntrường để lưu trữ các truy vấn (không thành công) của người dùng.
* Kiểu dữ liệu củatruy vấntrường phải làmáy lọc nước.

Phần còn lại của lược đồ bao gồm định nghĩa được mượn từ chỉ mục sách công nghệ gốc.

Cũng giống như chúng ta định nghĩa các trường với các kiểu dữ liệu như văn bản, dài, gấp đôi, từ khóa, v.v.trên, Elasticsearch cung cấp một loại percolator. Trường truy vấn được định nghĩa là percolator trong danh sách 12.29 và nó mong đợi một truy vấn làm giá trị trường, mà chúng ta sẽ thấy ngay sau đây.

Bây giờ, khi chỉ mục percolator (tech\_books\_percolator) của chúng ta đã sẵn sàng, bước tiếp theo là lưu trữ các truy vấn. Các truy vấn này thường không trả về kết quả cho người dùng (như ví dụ Python).

Trong thế giới thực, một truy vấn của người dùng không mang lại kết quả sẽ được lập chỉ mục vào đâypercolator index. Quá trình đối chiếu các truy vấn không thành công của người dùng vào một percolator index có thể được thực hiện bên trong ứng dụng tìm kiếm, nhưng thật không may, việc thảo luận về nó nằm ngoài phạm vi ở đây. Hãy tưởng tượng rằng truy vấn trong danh sách 12.29 không tạo ra kết quả và bây giờ được gửi đến percolator index để lưu trữ. Danh sách tiếp theo cung cấp mã để lưu trữ truy vấn.

**Liệt kê 12.30 Lưu trữ truy vấn để tìm kiếm một cuốn sách**

ĐẶT tech\_books\_percolator/\_doc/1

{

"truy vấn" : {

"cuộc thi đấu":{

"tên":"Python"

}

}

}

**Cùng một truy vấn mà người dùng đã thử tìm kiếm nhưng không nhận được kết quả khả quan**

Danh sách này cho thấy việc lập chỉ mục của một truy vấn, không giống như việc lập chỉ mục một tài liệu thông thường. Nếu bạn nhớ các hoạt động lập chỉ mục/tài liệu từ các chương trước, bất kỳ khi nào chúng ta lập chỉ mục một tài liệu, chúng ta sử dụng một tài liệu được định dạng JSON với tên-giá trịcặp. Tuy nhiên, mã này có truy vấn khớp.

Truy vấn trong danh sách 12.30 được lưu trữ trong chỉ mục tech\_books\_percolator với ID tài liệu là 1. Như bạn có thể hình dung, chỉ mục này tiếp tục tăng lên với các tìm kiếm không thành công. Tài liệu JSON bao gồm các truy vấn do người dùng đưa ra không trả về kết quả dương tính. Mảnh ghép cuối cùng của câu đố là tìm kiếm chỉ mục tech\_books\_percolator khi kho của chúng tôi được cập nhật. Là chủ hiệu sách, chúng tôi được kỳ vọng sẽ tích trữ; lần tiếp theo chúng tôi nhận được kho mới, hãy giả sử sách Python được bao gồm. Bây giờ chúng tôi có thể lập chỉ mục cho nó

vào mục tech\_books của chúng tôi để người dùng tìm kiếm và mua.

**Liệt kê 12.31 Đang tích trữ (lập chỉ mục) một Pythonsách**

ĐẶT tech\_books/\_doc/4

{

"name":"Python trong hành động",

**432 CPHẦN12*Tìm kiếm nâng cao***

"tags":["Python","Lập trình phần mềm"]

}

Bây giờ sách Python đã được lập chỉ mục, chúng ta cần chạy lại truy vấn không thành công của người dùng. Nhưng lần này, thay vì chạy truy vấn trên chỉ mục tech\_books, hãy chạy nó trên chỉ mục tech\_books\_percolator. Truy vấn trên chỉ mục percolator có cú pháp đặc biệt sau.

**Liệt kê 12.32 Tìm kiếm các truy vấn trongmáy lọc nướcchỉ số**

NHẬN tech\_books\_percolator/\_search

{

"truy vấn": {

"percolate": { "field": "query", "document": {

**Chỉ định**

**thấm quatruy vấn Thiết lập trường**

**tên để truy vấn**

**Chỉ định tài liệu**

"name":"Python trong hành động",

"tags":["Python","Lập trình phần mềm"]

}

}

}

}

**bao gồm cuốn sách gốc được lập chỉ mục trong tech\_books**

Truy vấn percolate mong đợi hai bit đầu vào: một trường có giá trị là query (điều này trùng với thuộc tính được xác định trong ánh xạ percolator trong danh sách 12.28) và một document, là cùng một document mà chúng tôi đã lập chỉ mục trong chỉ mục tech\_books của mình. Tất cả những gì chúng tôi cần làm là kiểm tra bất kỳ truy vấn nào khớp với document đã cho. May mắn thay, Python in Action khớp (chúng tôi đã lập chỉ mục một truy vấn vào chỉ mục percolator của mình trước đó).

Với tài liệu Python được định nghĩa trong danh sách 12.31, chúng ta có thể trả về một truy vấn từchỉ mục tech\_books\_percolator. Điều này cho phép chúng tôi thông báo cho người dùng rằng cuốn sách họ đang tìm kiếm đã có hàng trở lại! Lưu ý rằng chúng tôi có thể mở rộng truy vấn được lưu trữ trong chỉ mục tech\_books\_percolator với ID người dùng cụ thể.

Percolators hơi khó hiểu, nhưng khi bạn hiểu được trường hợp sử dụng, chúng không khó để triển khai. Hãy nhớ rằng phải luôn có một quy trình tự động, bán tự động hoặc thủ công để đồng bộ hóa các thao tác mà người dùng thực hiện với dữ liệu được lưu trữ trong chỉ mục percolator.

Vậy là xong! Bằng cách thảo luận về các truy vấn nâng cao trong chương này, chúng ta đã đề cập đến phần tìm kiếm của Elasticsearch. Phần cuối cùng là tổng hợp dữ liệu, đây là chủ đề của chương tiếp theo. Hãy theo dõi để tìm hiểu thêm về cách chúng ta có thể tìm thấy thông tin tình báo trong dữ liệu của mình bằng cách phân tích dữ liệu đó bằng nhiều hàm toán học và thống kê khác nhau.

#### Bản tóm tắt

* Elasticsearch hỗ trợđiểm địa lýVàhình dạng địa lýcác kiểu dữ liệu để làm việc với dữ liệu địa lý.
* Truy vấn không gian địa lý lấy vị trí và địa chỉ bằng cách sử dụng tọa độ được hình thành từ các giá trị kinh độ và vĩ độ.

***Bản tóm tắt* 433**

* + MỘThộp giới hạn địa lýtruy vấn lấy các địa chỉ theo dạng hình chữ nhật địa lý, được xây dựng bằng cách sử dụng một cặp giá trị kinh độ và vĩ độ làm tọa độ trên cùng bên trái và dưới cùng bên phải.
  + MỘTkhoảng cách địa lýtruy vấn tìm vị trí bên trong một vùng hình tròn có tâm được cung cấp làm trục và bán kính là khoảng cách từ trục.
  + MỘThình dạng địa lýtruy vấn tìm nạp tất cả các vị trí phù hợp bên trong một phạm vi nhất định được tạo thành bởi tọa độ.
  + MỘThình dạng địa lýtruy vấn tìm kiếm các hình dạng 2D trong hệ tọa độ hình chữ nhật (mặt phẳng Descartes).
  + Truy vấn Span là truy vấn nâng cao hoạt động với các vị trí cấp thấp hơn của từng mã thông báo hoặc từ. Elasticsearch hỗ trợ một sốkhoảng cáchtruy vấn:span\_đầu tiên,khoảng\_trong,khoảng cách gầnvà nhiều hơn nữa.
  + MỘTkhoảng cách\_tính năngTruy vấn là truy vấn chuyên biệt trong đó độ gần của tài liệu với một điểm trọng tâm nhất định sẽ làm tăng điểm liên quan và do đó, ưu tiên tài liệu cao hơn.
  + MỘTghimtruy vấn cho phép chúng ta gom các tài liệu bổ sung (thậm chí là không khớp) với tập kết quả gốc, có khả năng tạo ra các kết quả tìm kiếm được tài trợ.
  + MỘTnhiều hơn như thế nàytruy vấn tìm kiếm các kết quả có liên quan hoặc tương tự.
  + MỘTthấm quatruy vấn cho phép chúng tôi thông báo cho người dùng về các tìm kiếm tiêu cực của họ vào một ngày trong tương lai.

*Tổng hợp*

***Chương này bao gồm***

* Cơ bản về tổng hợp
* Làm việc với tổng hợp số liệu
* Phân loại dữ liệu bằng cách sử dụng tổng hợp bucket
* Kết nối các số liệu và tổng hợp nhóm trong tổng hợp đường ống

Tìm kiếm và phân tích là hai mặt của một đồng xu, và Elasticsearch cung cấp thông tin chi tiết tuyệt đối và vô số tính năng. Elasticsearch là công ty dẫn đầu thị trường về phân tích bằng cách cung cấp các chức năng giàu tính năng để truy vấn và phân tích dữ liệu, do đó cho phép các tổ chức tìm ra thông tin chi tiết và trí tuệ sâu sắc từ dữ liệu của họ. Trong khi tìm kiếm tìm thấy kết quả cho các tiêu chí nhất định, thì phân tích, mặt khác, giúp các tổ chức lấy được số liệu thống kê và số liệu từ đó. Cho đến nay, chúng ta đã xem xét việc tìm kiếm tài liệu từ một tập hợp tài liệu nhất định. Với phân tích, chúng ta lùi lại một bước và kiểm tra dữ liệu ở cấp độ cao để rút ra kết luận về nó.

Trong chương này, chúng ta sẽ xem xét chi tiết về các tổng hợp của Elasticsearch. Elasticsearch tự hào có nhiều tổng hợp, chủ yếu được phân loại thành một trong các loại sau: metric, bucket và pipeline. Các tổng hợp metric cho phép chúng ta sử dụng các hàm phân tích như

**434**

***13.1 Tổng quan* 435**

sum, min, max và avg để thực hiện các phép tínhtrên dữ liệu. Tổng hợp bucket giúp chúng ta phân loại dữ liệu thành bucket hoặc phạm vi. Cuối cùng, tổng hợp pipeline cho phép chúng ta liên kết các tổng hợp: tức là lấy tổng hợp số liệu hoặc bucket và tạo ra các tổng hợp mới.

Elasticsearch cung cấp rất nhiều tổng hợp ngay khi cài đặt. Chúng ta sẽ làm quen với từng loại trước khi bắt tay vào thực hành. Lưu ý rằng không thực tế khi xem xét từng loại do hạn chế về không gian, nhưng việc tìm hiểu các khái niệm và áp dụng chúng vào một số tổng hợp phổ biến quan trọng hơn là ghi lại từng loại. Tuy nhiên, mã nguồn của cuốn sách chứa càng nhiều tổng hợp càng tốt, đặc biệt là những tổng hợp không được thảo luận trong chương này.

**GHI CHÚ**Mã cho chương này có sẵn trên GitHub ([http://mng.bz/](http://mng.bz/wvZg) [wvzg](http://mng.bz/wvZg)) và trên trang web của cuốn sách ([https://www.manning.com/books/elastic](https://www.manning.com/books/elasticsearch-in-action-second-edition) [tìm kiếm-trong-hành-động-phiên-bản-thứ-hai](https://www.manning.com/books/elasticsearch-in-action-second-edition)).

#### Tổng quan

Tổng hợp cho phép các công ty hiểu dữ liệu đã tích lũy của họ. Chúng giúp hiểu khách hàng và mối quan hệ của họ, đánh giá hiệu suất sản phẩm, dự báo doanh số và trả lời nhiều câu hỏi về hiệu suất ứng dụng theo thời gian, các mối đe dọa bảo mật, v.v. Tổng hợp trong Elasticsearch chủ yếu thuộc ba loại:

* + - *Tổng hợp số liệu*—Các tổng hợp này tạo ra các số liệu như tổng, trung bình, tối thiểu, tối đa, lượt truy cập hàng đầu, chế độ và nhiều số liệu khác. Chúng bao gồm các số liệu phổ biến, giá trị đơn giúp hiểu dữ liệu trên một tập hợp lớn các tài liệu bằng cách trả lời các câu hỏi như: Tổng doanh số của tất cả các sản phẩm trong tháng qua là bao nhiêu? Số lỗi API tối thiểu là bao nhiêu? và Lượt truy cập tìm kiếm hàng đầu là bao nhiêu?
    - *Tổng hợp thùng*—Phân nhóm là quá trình thu thập dữ liệu vào các nhóm khoảng thời gian. Các tổng hợp này chia dữ liệu thành các tập hợp nhất định: ví dụ, chia ô tô thành năm đăng ký, chia học sinh thành các lớp khác nhau, v.v. Biểu đồ tần suất, phạm vi, thuật ngữ, bộ lọc và các thành phần khác thuộc loại này.
    - *Tổng hợp đường ống*—Các tổng hợp này hoạt động dựa trên kết quả của các tổng hợp khác để cung cấp một tập hợp phân tích thống kê phức tạp như trung bình động, đạo hàm và các phân tích khác.

Chúng tôi thảo luận về các tổng hợp này với các ví dụ trong các phần sau. Trước tiên, hãy xem cú pháp và điểm cuối được sử dụng để thực hiện các tổng hợp.

* + 1. Điểm cuối và cú pháp

Sau khi làm việc với các truy vấn tìm kiếm, bạn hẳn đã khá quen thuộc với điểm cuối \_search. Tin tốt là chúng ta có thể sử dụng cùng điểm cuối \_search để chạy các phép tổng hợp. Tuy nhiên, phần thân của yêu cầu sử dụng một đối tượng mới: các phép tổng hợp (viết tắt là aggs) thay vì đối tượng truy vấn thông thường. Đoạn mã sau đây cho thấy cú pháp truy vấn tổng hợp:

**436 CPHẦN13*Tổng hợp***

NHẬN <index\_name>/\_search

{

"tổng hợp|aggs": { "TÊN": {

"AGG\_TYPE": {}

}

}

}

Đối tượng aggs cho Elasticsearch biết rằng lệnh gọi truy vấn là một loại tổng hợp. Thuộc tính NAME do người dùng cung cấp sẽ đưa ra tên phù hợp cho tổng hợp.Cuối cùng, AGG\_TYPE là kiểu tổng hợp: tổng, min, max, phạm vi, thuật ngữ, biểu đồ, v.v.

* + 1. Kết hợp tìm kiếmvà tổng hợp

Chúng ta cũng có thể kết hợp các tổng hợp với các truy vấn. Ví dụ, chúng ta có thể chạy một truy vấn đểlấy kết quả và sau đó chạy tổng hợp trên tập kết quả đó. Chúng được gọi là tổng hợp có phạm vi vì đầu vào là kết quả của truy vấn. Cú pháp được mở rộng một chút từ truy vấn tổng hợp trước đó:

NHẬN <index\_name>/\_search

{

"query": { "QUERY\_TYPE": {

"FIELD": "VĂN BẢN"

}

},

"aggs": {

"TÊN": {

"AGG\_TYPE": {}

}

}

}

Phạm vi tổng hợp được xác định bởi đầu ra của truy vấn. Nếu chúng ta không liên kết truy vấn với yêu cầu tổng hợp, thì chúng ta cho rằng nó sẽ hoạt động trên tất cả các tài liệu của chỉ mục (hoặc các chỉ mục) được xác định trong URL yêu cầu.

* + 1. Tổng hợp nhiều và lồng nhau

Ngoài việc chạy tổng hợp solo, chúng ta có thể thực hiện nhiều tổng hợp trên tập dữ liệu đã cho. Tính năng này cực kỳ tiện dụng nếu chúng ta cần trích xuất phân tích trên nhiều trường với nhiều điều kiện. Ví dụ, chúng ta có thể muốn tạo một histo-gram doanh số bán iPhone 14 mỗi ngày cũng như tổng hợp doanh số bán hàng trong tháng. Để thực hiện việc này, chúng ta có thể sử dụng tổng hợp phân nhóm biểu đồ và tổng hợp số liệu.

Ngoài ra, đôi khi chúng ta cần lồng các tập hợp; ví dụ, dữ liệu được phân nhóm trong biểu đồ có thể cần phân loại thêm (phân nhóm) dữ liệu theo ngày hoặc bằng cách tìm giá trị tối thiểu và tối đa cho mỗi nhóm. Trong trường hợp này, dữ liệu được tổng hợp

* 1. ***Hệ méttổng hợp* 437**

trong mỗi thùng cấp cao nhất được đưa vào thùng cấp tiếp theo để tổng hợp thêm. Tổng hợp lồng nhau có thể được phân loại thêm bằng cách sử dụng nhiều thùng hơn hoặc số liệu giá trị đơn (tổng, trung bình, v.v.). Chúng ta sẽ xem xét tổng hợp lồng nhau với các ví dụ trong các phần tiếp theo.

* + 1. Bỏ qua kết quả

Truy vấn tìm kiếm (hoặc tổng hợp) có xu hướng trả về các tài liệu nguồn trong phản hồi nếu không được yêu cầu xóa chúng trong truy vấn. Chúng tôi đã xem xét cách thao tác các phản hồi trong chương 8, nơi chúng tôi định cấu hình \_source, \_source\_includes và \_source\_loại trừ các tham số có cài đặt phù hợp.

Khi làm việc với các tổng hợp, các tài liệu trong phản hồi thường không quan trọng vì chúng ta quan tâm nhiều hơn đến các tổng hợp hơn là nguồn. Trong những trường hợp bình thường, thật không may, các tài liệu nguồn được gắn thẻ cùng vớitổng hợp ngay cả khi chúng ta chạy truy vấn tổng hợp. Nếu đây không phải là ý định của chúng ta (và thường là không phải khi chạy tổng hợp), chúng ta có thể điều chỉnh truy vấn bằng cách đặt tham số kích thước thành không. Đoạn mã sau đây cho thấy cách tiếp cận này:

NHẬN tv\_sales/\_search

{

**kích thước = 0**

"aggs": {

[**Đặt tham số kích thước thành 0**](https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-pipeline.html)

<<câu hỏi của bạn ở đây>>

}

}

}

}

Chúng tôi sử dụng tham số tiện dụng này (kích thước = 0) trong suốt chương này để ẩn các tài liệu nguồn khi thực hiện truy vấn tổng hợp.

Như đã chỉ ra trước đó, các tổng hợp được phân loại rộng rãi thành metric, bucket hoặc pipeline. Trong phần tiếp theo, chúng ta sẽ xem xét các tổng hợp metric với dữ liệu mẫu được thiết kế riêng cho các phép tính metric. Các phần tiếp theo sẽ thảo luận về các tổng hợp còn lại.

#### Tổng hợp số liệu

Tổng hợp số liệu là tổng hợp đơn giản mà chúng ta thường sử dụng trong cuộc sống hàng ngày. Ví dụ,

* + - Chiều cao và cân nặng trung bình của học sinh trong lớp là bao nhiêu?
    - Giao dịch phòng ngừa rủi ro tối thiểu là gì?
    - Thu nhập gộp của một cuốn sách bán chạy nhất là bao nhiêu?

Elasticsearch cung cấp các hàm số liệu để tính toán hầu hết các số liệu đơn và đa giá trị. Nếu bạn đang thắc mắc chúng tôi muốn nói gì về tổng hợp đơn so với đa giá trị, thì đây là một khái niệm đơn giản dựa trên số lượng đầu ra.

Tổng hợp số liệu giá trị đơn là tổng hợp trên một tập dữ liệu đưa ra một giá trị đơn, chẳng hạn như min, max hoặc average. Các tổng hợp này hoạt động trên các tài liệu đầu vào để tạo ra dữ liệu đầu ra giá trị đơn. Mặt khác, số liệu thống kê và

**438 CPHẦN13*Tổng hợp***

Tổng hợp extended\_stats tạo ra nhiều giá trị dưới dạng đầu ra. Ví dụ, đầu ra tổng hợp stats bao gồm min, max, sum, avg và một vài giá trị khác cho cùng một tập tài liệu.

Hầu hết các tổng hợp số liệu đều tự giải thích. Ví dụ, tổng hợp tổngtổng hợp tất cả các giá trị đã cho và avg tính trung bình các giá trị. Nếu bạn cần làm việc với các số liệu không được thảo luận trong chương này, hãy xem tài liệu trên trang Elasticsearch để biết cách sử dụng:<http://mng.bz/qrRz>.

* + 1. Dữ liệu mẫu

Trong một vài phần tiếp theo, chúng ta sẽ thảo luận về các tổng hợp phổ biến. Trước khi thực hiện, hãy chuẩn bị Elasticsearch bằng một vài tài liệu. Danh sách sau chuẩn bị kho dữ liệu của chúng ta bằng danh sách doanh số bán TV bằng cách tạo chỉ mục mới, tv\_sales. Bạn cũng có thể tải xuống cùng một bộ mẫu từ GitHub (<http://mng.bz/7DQ4>) hoặc trang web của cuốn sách.

**Liệt kê 13.1 Lập chỉ mục doanh số bán TVdữ liệu**

ĐẶT tv\_sales/\_bulk

{"chỉ mục":{"\_id":"1"}}

{"thương hiệu": "Samsung","tên":"TV UHD","kích thước\_inches":65,"giá\_gbp":1400,

➥"bán hàng":17}

{"chỉ mục":{"\_id":"2"}}

{"thương hiệu":"Samsung","tên":"TV UHD","kích thước\_inches":45,"giá\_gbp":1000,

➥"bán hàng":11}

{"chỉ mục":{"\_id":"3"}}

{"thương hiệu":"Samsung","tên":"TV UHD","kích thước\_inches":23,"giá\_gbp":999,

➥"bán hàng":14}

{"chỉ mục":{"\_id":"4"}}

{"thương hiệu":"LG","tên":"TV 8K","kích thước\_inches":65,"giá\_gbp":1499,"doanh số":13}

{"chỉ mục":{"\_id":"5"}}

{ "thương hiệu":"LG","tên":"TV 4K","kích thước\_inches":55,"giá\_gbp":1100,"doanh số":31}

{"chỉ mục":{"\_id":"6"}}

{"thương hiệu":"Philips","tên":"TV 8K","kích thước\_inches":65,"giá\_gbp":1800,

➥"bán hàng":23}

{"chỉ mục":{"\_id":"7"}}

{"name":"TV 8K","size\_inches":65,"price\_gbp":2000,"sales":23}

{"chỉ mục":{"\_id":"9"}}

{"name":"TV 8K","size\_inches":65,"price\_gbp":2000,"sales":23,

➥"best\_seller":đúng}

{"chỉ mục":{"\_id":"10"}}

{"name":"TV 4K","size\_inches":75,"price\_gbp":2200,"sales":14,

➥"best\_seller":false}

Danh sách này lập chỉ mục các tài liệu có nhiều thuộc tính khác nhau vào chỉ mục tv\_sales. Tham gialưu ý đặc biệt của trường best\_seller: nó chỉ được đặt trên hai bản ghi cuối cùng. Bây giờ chúng ta đã có một tập dữ liệu mẫu, hãy chạy một vài tổng hợp số liệu phổ biến.

* + 1. Chỉ số value\_count

Số liệu value\_count đếm số lượng giá trị hiện có cho một trường trong một tập hợp các tài liệu. Nếu yêu cầu của chúng tôi là lấy số lượng giá trị trong Elasticsearch cho một

***13.2 Hệ méttổng hợp* 439**

trường, tổng hợp value\_count thực hiện những gì chúng ta cần. Ví dụ, chạy truy vấn sau sẽ trả về số lượng giá trị trong stash của chúng ta cho trường best\_seller.

**Liệt kê 13.2 Tìm số lượng giá trị cho một trường**

NHẬN tv\_sales/\_search

{

kích thước = 0 "aggs": {

"tổng số-giá-trị": { "giá\_trị\_đếm": {

"field": "best\_seller"

**Đặt tên cho kết quả tổng hợp**

**Têntổng hợp (value\_count)**



}

} **Trường mà trên đó**

} **value\_count được thực hiện**

}

Tổng hợp value\_count được thực hiện trên trường best\_seller. Lưu ý rằng theo mặc định, tổng hợp không được thực hiện trên các trường văn bản. Trong dữ liệu mẫu của chúng tôi, trường best\_seller là kiểu dữ liệu boolean và là ứng cử viên tốt cho tổng hợp số liệu value\_count. Chạy truy vấn trong danh sách 13.2 sẽ cho kết quả sau:

"tổng hợp" : { "tổng giá trị" : {

**"giá trị" : 2**

}

}

Có hai giá trị (hai tài liệu) cho trường best\_seller. Lưu ý rằng value\_count không chọn các giá trị duy nhất; nó không xóa các giá trị trùng lặp cho trường được chỉ định trên toàn bộ tập tài liệu.

**Tổng hợp trên các trường văn bản không được tối ưu hóa**

Các trường văn bản không hỗ trợ sắp xếp,viết kịch bản và tổng hợp. Tổng hợp lý tưởng nhất là được thực hiện trên các trường không phải văn bản nhưcon số,từ khóa,Boolean, và vân vân. Bởi vìchữcác trường không được tối ưu hóađối với các tổng hợp, theo mặc định, Elasticsearch ngăn chúng ta tạo các truy vấn tổng hợp trên chúng. Nếu bạn tò mò, hãy thử tổng hợp trênchữtrường như trường tên và xem Elasticsearch đưa ra ngoại lệ nào:

"nguyên nhân gốc rễ" : [

{

"loại": "ngoại lệ\_đối\_số\_bất\_pháp",

"lý do": "Các trường văn bản không được tối ưu hóa cho các hoạt động yêu cầu

➥dữ liệu trường theo tài liệu như tổng hợp và sắp xếp, vì vậy những

➥các hoạt động bị vô hiệu hóa theo mặc định. Vui lòng sử dụng một trường từ khóa

➥thay vào đó. Hoặc, đặt fielddata=true trên [tên] để

➥tải dữ liệu trường bằng cách đảo ngược chỉ số đảo ngược. Lưu ý rằng điều này

➥có thể sử dụng bộ nhớ đáng kể."

}

**440 CPHẦN13*Tổng hợp***

***(tiếp theo)***

Như lỗi đã nói, chạy tổng hợp trên mộtchữtrường bị cấm theo mặc định, vì vậy nếu chúng ta muốn thực hiện tổng hợp trênchữcác trường, chúng ta cần phải kích hoạtdữ liệu trườngtrên các trường tương ứng. Chúng ta có thể thiết lập"fielddata": đúngkhi xác định ánh xạ:

ĐẶT tv\_sales\_with\_field\_data

{

"ánh xạ": { "thuộc tính": {

"tên":{

"type": "text", "fielddata": true

}

}

}

}

Lưu ý rằng việc kích hoạtdữ liệu trườngcó thể dẫn đến hiệu suấtvấn đề vì dữ liệu được lưu trữ trong bộ nhớ trên các nút. Thay vì phải chịu ảnh hưởng về hiệu suất bằng cách bậtdữ liệu trường, thay vào đó chúng ta có thể tạo một kiểu dữ liệu đa trường vớitừ khóanhư loại thứ hai. Điều này có hiệu quả vìtừ khóacác kiểu dữ liệu được phép sử dụng để tổng hợp.

Tìm giá trị trung bình của một tập hợp số là một hoạt động thường xuyên trong phân tích. Như mong đợi, Elasticsearch cung cấp một hàm tiện dụng có tên là avg để tìm giá trị trung bình, đây là chủ đề của phần tiếp theo.

* + 1. Số liệu trung bình

Tìm giá trị trung bình của một tập hợp số là một hàm thống kê cơ bản mà chúng ta thường yêu cầu. Elasticsearch cung cấp phép tổng hợp số liệu avg ngay khi chạy phép tính trung bình. Ví dụ, truy vấn tiếp theo sẽ lấy giá TV trung bình bằng avg.

**Danh sách 13.3 Giá trung bình của tất cảTivi**

NHẬN tv\_sales/\_search

{

"kích thước": 0,

"aggs": { "tv\_average\_price": {

"trung bình": {

"trường": "giá\_gbp"

**Đặt tên cho tập hợp**

**Tính giá trung bình**

}

} **Trường mà trên đó**

} **trung bình được tính toán**

}

tv\_giá\_trung\_bìnhlà tên do người dùng định nghĩa được đặt cho tổng hợp trung bình này.trung bình

khai báo trong mã biểu diễn hàm trung bình. Trường dữ liệu mà chúng ta

***13.2 Hệ méttổng hợp* 441**

muốn chạy số liệu avg trường đơn của chúng tôi được gọi là trường. Khi truy vấn được thực hiện, chúng tôi nhận được kết quả sau:

"tổng hợp" : { "tv\_average\_price" : {

"giá trị" : 1555.3333333333333

}

}

Công cụ tính toán giá trung bình của tất cả các TV và trả về cho người dùng. Giá TV trung bình của cả sáu tài liệu là khoảng 1555 bảng Anh.

* + 1. Tổng số liệu

Phép đo tổng giá trị đơn cộng các giá trị của trường đang xét và tạo ra kết quả cuối cùng. Ví dụ, để tìm tổng giá trị của tất cả các TV đã bán, chúng ta có thể đưa ra truy vấn sau.

**Danh sách 13.4 Tổng số tất cả các TVđã bán**

NHẬN tv\_sales/\_search

{

"kích thước": 0,

"aggs": { "tv\_total\_price": {

"tổng": {

"trường": "giá\_gbp"

}

}

}

}

Phép đo tổng cộng tất cả các mức giá và tạo ra một con số duy nhất là 13.998 bảng Anh khi chúng ta chạy truy vấn. Tương tự, chúng ta hãy xem xét các hàm số liệu tối thiểu và tối đa tiếp theo.

* + 1. Số liệu min và max

Đôi khi chúng ta cần tìm số lượng tối thiểu và tối đa từ một tập hợp các giá trị, chẳng hạn như số lượng diễn giả tối thiểu có sẵn cho một hội nghị hoặc phiên họp có nhiều người tham dự nhất. Elasticsearch hiển thị các số liệu tương ứng dưới dạngmin và max để tạo ra các giá trị cực đại này của một tập dữ liệu. Các số liệu này tự giải thích, nhưng để có sự đầy đủ, chúng ta hãy xem xét chúng một cách ngắn gọn.

**TôiSỐ LIỆU TỐI THIỂU**

Giả sử chúng ta muốn tìm chiếc TV rẻ nhất trong kho của mình. Rõ ràng đây là ứng cử viên chosử dụng số liệu min trên các giá trị dữ liệu.

**442 CPHẦN13*Tổng hợp***

**Niêm yết 13.5 Giá rẻ nhất choTivi**

NHẬN tv\_sales/\_search

{

"kích thước": 0,

"aggs": { "giá tivi rẻ nhất": {

"phút": {

"trường": "giá\_gbp"

}

**Tính giá trị nhỏ nhất**

**Áp dụng hàm min cho trường này**

}

}

}

Từ khóa min lấy số liệu, hoạt động trên trường price\_gbp để tạo ra kết quả mong đợi: giá trị tối thiểu của trường được lấy từ tất cả các tài liệu. Chúng tôi lấy TV có giá thấp nhất (£999) trong kho của mình bằng cách thực hiện truy vấn.

**Tôi****ĐỊNH LƯỢNG CỰC ĐỈNH**

Chúng ta có thể sử dụng logic tương tự để tìm ra chiếc TV bán chạy nhất: chiếc TV có số lượng bán ra nhiều nhất.

**Danh sách 13.6 Bán chạy nhấtTV**

NHẬN tv\_sales/\_search

{

"kích thước": 0,

"aggs": { "bán chạy nhất trên truyền hình": {

"tối đa": {

"lĩnh vực": "bán hàng"

}

}

}

}

Khi chúng tôi thực hiện truy vấn, nó trả về một chiếc TV bán rất nhanh (doanh số tối đa). Từ kết quả, đó là TV 8K của LG với 48 lần bán.

* + 1. Số liệu thống kê

Trong khi các số liệu trước đó là giá trị đơn (có nghĩa là chúng chỉ hoạt động trên một trường duy nhất), số liệu thống kê sẽ lấy tất cả các hàm thống kê phổ biến. Đây là tổng hợp nhiều giá trị lấy một số số liệu (trung bình, tối thiểu, tối đa, đếmvà tổng) cùng một lúc. Truy vấntrong danh sách tiếp theo áp dụng tổng hợp số liệu thống kê vào trường price\_gbp.

**Liệt kê 13.7 Tất cả các số liệu thống kê chung tạimột lần**

NHẬN tv\_sales/\_search

{

"kích thước": 0,

"aggs": {

* 1. ***Hệ méttổng hợp* 443**

"thống kê chung":{ "thống kê": {

**Chức năng thống kê**

"trường": "giá\_gbp"

}

}

}

}

**Áp dụng số liệu thống kê vào trường này**

Sau khi thực hiện, truy vấn này trả về kết quả sau:

"tổng hợp" : { "common\_stats" : {

"đếm" : 6,

"tối thiểu" : 999.0,

"tối đa": 1800.0,

"trung bình": 1299.666666666667,

"tổng" : 7798.0

}

}

Số liệu thống kê trả về tất cả năm số liệu cùng một lúc. Điều này rất tiện lợi nếu chúng ta muốn xem tất cả các tổng hợp cơ bản ở một nơi.

* + 1. Chỉ số extended\_stats

Mặc dù stats là một số liệu hữu ích, phổ biến, nhưng nó không cung cấp các phân tích nâng cao như phương sai, độ lệch chuẩn và các hàm thống kê khác. Elasticsearch bao gồm một số liệu khác có tên là extended\_stats ngay khi xuất xưởng, đây là họ hàng củathống kê và xử lý các số liệu thống kê nâng cao.

Cácthống kê mở rộngsố liệu cung cấp ba số liệu thống kê ngoài các số liệu thống kê tiêu chuẩn:tổng\_bình\_bình\_thượng,sự khác biệt, Vàđộ lệch chuẩn. Danh sách sau đây minh họa.

**Danh sách13.8 Thống kê nâng cao (mở rộng) vềgiá\_GBPcánh đồng**

NHẬN tv\_sales/\_search

{

"kích thước": 0,

"aggs": { "thống kê bổ sung":{

"extended\_stats": { "field": "price\_gbp"

}

**Áp dụng hàm extended\_stats sẽ lấy các phép đo thống kê nâng cao.**

**Sử dụng extended\_stats trên trường này**

}

}

}

Chúng tôi gọi hàm extended\_stats trên trường price\_gbp. Thực hiện như vậy sẽ lấy tất cả dữ liệu thống kê được hiển thị trong hình 13.1. Truy vấn tính toán rất nhiều thông tin thống kê nâng cao về price\_gbp cũng như các số liệu chung (trung bình, tối thiểu, tối đa, v.v.) và nhiều phương sai và độ lệch chuẩn khác nhau.

**444 CPHẦN13*Tổng hợp***



**Hình 13.1 Thống kê mở rộng vềgiá\_GBPcánh đồng**

* + 1. Tính số lượngmét

Số liệu về số lượng trả về các giá trị duy nhất cho tập hợp các tài liệu đã cho. Số liệu về một giá trị sẽ lấy các lần xuất hiện của các giá trị riêng biệt từ dữ liệu của chúng tôi. Ví dụ: truy vấn trong danh sách tiếp theo sẽ lấy số lượng các thương hiệu TV duy nhất trong chỉ mục tv\_sales của chúng tôi.

**Liệt kê 13.9 Tìm kiếm các thương hiệu TV độc đáo**

NHẬN tv\_sales/\_search

{

"kích thước": 0,

"aggs": { "unique\_tvs": {

"số lượng": {

**Chỉ số về số lượng sẽ lấy các giá trị duy nhất.**

"field": "thương hiệu.từ khóa"

}

}

}

}

**Áp dụng số lượng cho trường brand.keyword**

Bởi vì chúng tôi có bốn thương hiệu độc đáo (Samsung, LG, Philips và Panasonic),kết quả truy vấn có 4 trong tổng hợp unique\_tvs:

"tổng hợp" : { "unique\_tvs" : {

"giá trị" : 4

}

}

Dữ liệu được phân phối trong Elasticsearch, do đó, việc cố gắng lấy số lượng chính xác của số lượng có thể dẫn đến các vấn đề về hiệu suất. Để lấy một số chính xác, dữ liệu phải được truy xuất và

* 1. ***Xôtổng hợp* 445**

được tải vào một hashset trong bộ nhớ đệm trong bộ nhớ. Và vì đây là một hoạt động tốn kém, nên số lượng chạy như một phép tính gần đúng. Vì vậy, số lượng cho các giá trị duy nhất có thể không chính xác, nhưng chúng gần đúng.

Ngoài các phép tổng hợp số liệu mà chúng ta đã thảo luận, Elasticsearch còn tiết lộ một số phép tổng hợp khác. Việc xem xét tất cả chúng trong chương này là không thực tế, vì vậy tôi thực sự khuyên bạn nên xem tài liệu Elasticsearch để tìm hiểu về những phép tổng hợp không được đề cập ở đây.

Loại số liệu tiếp theo tạo ra một nhóm tài liệu thay vì tạo ra mộtsố liệu trên tất cả các tài liệu. Chúng được gọi là tổng hợp bucket, một chủ đề được thảo luận trong phần tiếp theo.

#### Xôtổng hợp

Một yêu cầu đối với dữ liệu là chạy các hoạt động nhóm. Elasticsearch gọi các hoạt động nhóm này là tập hợp nhóm. Mục đích duy nhất của chúng là phân loại dữ liệu thành các nhóm, thường được gọi là nhóm.

Bucketing là một quá trình thu thập dữ liệu vào các bucket khoảng thời gian. Ví dụ,

* + - Nhóm những người chạy marathon theo nhóm tuổi (21–30, 31–40, 41–50, v.v.)
    - Phân loại trường học dựa trên xếp hạng kiểm tra của họ (tốt, xuất sắc, đặc biệt)
    - Nhận số lượng nhà mới được xây dựng mỗi tháng hoặc mỗi năm

Để chơi với các tổng hợp nhóm, chúng tôi sử dụng lại một tập dữ liệu mà chúng tôi đã làm việc trong quá khứ: dữ liệu sách. Lấy tập dữ liệu từ GitHub (<http://mng.bz/mVZ0>) hoặc trang web của cuốn sách và lập chỉ mục cho nó. Đoạn trích mẫu này cung cấp lời nhắc nhở nhanh (đây không phải là tập dữ liệu đầy đủ):

POST \_số lượng lớn

{"index":{"\_index":"sách","\_id":"1"}}

{"title": "Java Core Tập I – Cơ bản","author": "Cay S. Horstmann",

➥"phiên bản": 11, "tóm tắt": "Sách tham khảo Java cung cấp thông tin chi tiết

➥giải thích về các tính năng khác nhau của Core Java, bao gồm cả ngoại lệ

➥xử lý, giao diện và biểu thức lambda. Điểm nổi bật đáng chú ý

➥của cuốn sách bao gồm ngôn ngữ đơn giản, súc tích và chi tiết

➥ví dụ.","amazon\_rating": 4.6,"release\_date": "2018-08-27",

➥"thẻ": ["Ngôn ngữ lập trình, Lập trình Java"]}

{"index":{"\_index":"sách","\_id":"2"}}

{"title": "Java hiệu quả","author": "Joshua Bloch", "edition": 3,"synopsis":

➥"Một cuốn sách phải có cho mọi lập trình viên Java và những người khao khát Java,

➥Java hiệu quả tạo nên một bài đọc bổ sung tuyệt vời với các bài đọc khác

➥Sách Java hoặc tài liệu học tập. Cuốn sách cung cấp 78 phương pháp hay nhất để

➥theo dõi để làm cho mã tốt hơn.", "amazon\_rating": 4.7,

➥"release\_date": "2017-12-27", "tags": ["Thiết kế phần mềm hướng đối tượng"]}

Bây giờ chúng ta đã chuẩn bị máy chủ của mình bằng dữ liệu sách, hãy chạy một số tổng hợp nhóm phổ biến. Có ít nhất hai chục tổng hợp sẵn có, mỗi tổng hợp có chiến lược nhóm riêng. Như tôi đã đề cập trước đó, sẽ rất nhàm chán và lặp đi lặp lại khi ghi lại tất cả các tổng hợp trong cuốn sách này. Nhưng một khi bạn hiểu được khái niệm và

**446 CPHẦN13*Tổng hợp***

để có được ý tưởng làm việc với bucketing từ các ví dụ của chúng tôi, bạn sẽ có thể làm việc tốt với các tập hợp khác bằng cách làm theo tài liệu. Hãy bắt đầu với một tập hợp bucket phổ biến: histogram.

* + 1. Biểu đồ Histogram

Histogram là biểu đồ thanh gọn gàng biểu diễn dữ liệu được nhóm lại. Hầu hết các công cụ phần mềm phân tích đều cung cấp biểu diễn trực quan cũng như dữ liệu của histogram. Elasticsearch cho thấymột tập hợp biểu đồ dạng thùng ngay khi xuất xưởng.

Bạn có thể đã làm việc với biểu đồ histogram, trong đó dữ liệu được chia thành nhiều loại dựa trên khoảng thời gian thích hợp. Biểu đồ histogram trong Elasticsearch cũng không khác: chúng tạo ra một tập hợp các bucket trên tất cả các tài liệu theo một khoảng thời gian được xác định trước.

Hãy lấy ví dụ về việc phân loại sách theo xếp hạng. Chúng ta muốn tìm số lượng sách trong mỗi danh mục xếp hạng: 2 đến 3, 3 đến 4, 4 đến 5, v.v. Chúng ta có thể tạotổng hợp biểu đồ sử dụng giá trị khoảng thời gian cố định là 1 để sách được xếp vào nhóm xếp hạng một bước.

**Liệt kê 13.10 Abiểu đồtổng hợp cho sách**

NHẬN sách/\_tìm kiếm

{

**Phân loại dữ liệu thành**

"kích thước": 0,

"aggs": { "biểu đồ xếp hạng": {

"biểu đồ": {

"trường": "amazon\_rating",

**Đặt tên cho tập hợp của chúng tôi**



**Áp dụng tổng hợp trên trường này**

**xô**

}

"khoảng cách": 1

}

}

}

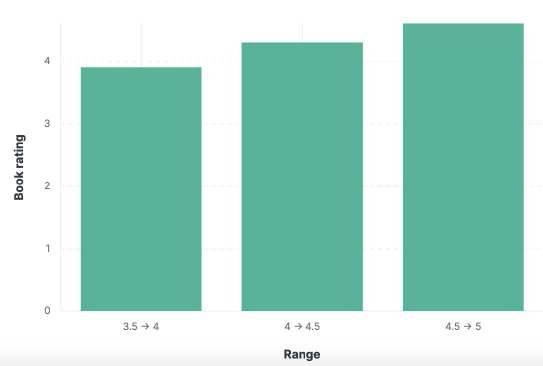
**Chỉ định khoảng thời gian thùng (một đơn vị)**

Tổng hợp biểu đồ histogram mong đợi trường mà chúng ta muốn tổng hợp các nhóm cũng như khoảng thời gian cho các nhóm. Trong danh sách này, chúng tôi chiacác cuốn sách dựa trên trường amazon\_rating với khoảng cách là 1. Điều này sẽ lấy tất cả các cuốn sách nằm trong khoảng từ 3 đến 4, 4 đến 5, v.v. Hình 13.2 hiển thị phản hồi.

Như bạn có thể thấy, chạy truy vấn sẽ lấy 2 cuốn sách nằm trong nhóm xếp hạng từ 2 đến 3, 35 cuốn sách có xếp hạng từ 3 đến 4, v.v. Phản hồi trong hình 13.2 cho thấy các nhóm có hai trường: khóa và doc\_count. Trường khóa biểu thịphân loại thùng và trường doc\_count chỉ ra số lượng tài liệu phù hợp với thùng.

**Hình 13.2 Tổng hợp xếp hạng sách dưới dạng biểu đồ**

***13.3 Xôtổng hợp* 447**



**Tổng hợp biểu đồ với Kibana**

Trong danh sách 13.10, chúng tôi đã phát triển một truy vấn tổng hợp và thực hiện truy vấn đó trong Kibana Console. Kết quả ở định dạng JSON không hấp dẫn về mặt trực quan, như thể hiện trong hình 13.2. Tùy thuộc vào khách hàng nhận được dữ liệu đó để biểu diễn dữ liệu đó dưới dạng biểu đồ trực quan. Tuy nhiên, Kibana có một bộ hình ảnh trực quan phong phú để tổng hợp dữ liệu. Làm việc với hình ảnh trực quan của Kibana nằm ngoài phạm vi của chương này, nhưng hình này hiển thị cùng dữ liệu được biểu diễn dưới dạng biểu đồ tần suất trong Bảng điều khiển của Kibana, lần này với khoảng cách là 0,5.

**Biểu đồ phân loại sách theo xếp hạng được xem dưới dạng biểu đồ thanh trong Bảng điều khiển của Kibana**

Như bạn có thể thấy, dữ liệu được phân loại thành các nhóm dựa trên khoảng 0,5 và các nhóm được điền đầy đủ các tài liệu phù hợp với chúng. Để tìm hiểu về hình ảnh hóa Kibana, hãy tham khảo tài liệu tại[https://www.elastic.co/guide/en/kibana/](https://www.elastic.co/guide/en/kibana/current/dashboard.html) [hiện tại/bảng điều khiển.html](https://www.elastic.co/guide/en/kibana/current/dashboard.html).

**TBIỂU ĐỒ NGÀY CỦA ANH**

Đôi khi chúng ta muốn nhóm dữ liệu dựa trên ngày tháng thay vì số. Ví dụ, chúng ta có thể muốn tìm tất cả các cuốn sách được phát hành mỗi năm, lấy doanh số bán hàng hàng tuần của một sản phẩm iPhone, xác định số lần máy chủ bị đe dọa mỗi giờ, v.v. Đây là nơi tổng hợp date\_histogram trở nên hữu ích.

Chiến lược phân nhóm histogram mà chúng ta đã xem xét ở phần trước dựa trên các khoảng số, nhưng Elasticsearch cũng cung cấp một histogram dựa trên ngày, được gọi là date\_histogram. Giả sử chúng ta muốn phân loại sách dựa trên ngày phát hành của chúng. Sau đây là truy vấn áp dụng bucketing để thực hiện việc này.

**Liệt kê 13.11 Abiểu đồ ngày thángtruy vấn**

NHẬN sách/\_tìm kiếm

{

"kích thước":0,

**448 CPHẦN13*Tổng hợp***

"aggs": { "biểu đồ năm phát hành":



{

"date\_histogram": { "field": "release\_date",

**Khai báo biểu đồ histogram**

**kiểu(biểu đồ ngày tháng) Áp dụng tổng hợp**

**đến lĩnh vực này**

"calendar\_interval": "năm"

}

}

}

}

**Xác định khoảng thời gian của thùng**

Tổng hợp date\_histogram yêu cầu trường mà tổng hợp sẽchạy và khoảng thời gian bucket. Trong ví dụ, chúng tôi sử dụng release\_date làm trường ngày với khoảng thời gian một năm.

**GHI CHÚ**Chúng ta có thể thiết lập giá trị khoảng thời gian của thùng thànhnăm,một phần tư,tháng,tuần,ngày,giờ,phút,thứ hai, hoặcmili giây, dựa trên yêu cầu của chúng tôi.

Chạy truy vấn trong danh sách 13.11 sẽ tạo ra các nhóm riêng lẻ cho mỗi năm và số lượng tài liệu trong nhóm đó. Đoạn mã sau đây hiển thị kết quả trong nháy mắt:

...

{

"key\_as\_string" : "2020-01-01T00:00:00.000Z", "key" : 1577836800000,

"doc\_count" : 5

},

{

"key\_as\_string" : "2021-01-01T00:00:00.000Z", "key" : 1609459200000,

"doc\_count" : 6

},

{

"key\_as\_string" : "2022-01-01T00:00:00.000Z", "key" : 1640995200000,

"doc\_count" : 3

}

...

Mỗi khóa (được biểu thị dưới dạng key\_as\_string) biểu thị một năm: 2020, 2021 và 2022. Theo kết quả cho thấy, năm cuốn sách được phát hành vào năm 2020, sáu cuốn vào năm 2021 và ba cuốn vào năm 2022.

**TÔITHIẾT LẬP KHOẢNG THỜI GIAN CHO BIỂU ĐỒ NGÀY**

Trong danh sách 13.11, chúng tôi đặt khoảng thời gian thànhnămtrongkhoảng thời gian lịch thuộc tính. Ngoài rakhoảng thời gian lịch, có một loại khoảng khác:khoảng thời gian cố định. Chúng ta có thể thiết lập theo khoảng thời gian lịch hoặc khoảng thời gian cố định.

Các*khoảng thời gian lịch*(được tuyên bố làkhoảng thời gian lịch) có thể nhận biết lịch, nghĩa là giờ và ngày trong tháng được điều chỉnh theo cài đặt ánh sáng ban ngày của lịch. Các đơn vị sau đây là giá trị chấp nhận được:năm,một phần tư,tháng,tuần,ngày,giờ,phút,thứ hai, Vàmili giây. Chúng cũng có thể được biểu diễn như các đơn vị riêng lẻ như1 năm,1q,1 triệu,1 tuần,1 ngày,1 giờ, Và1 phút, tương ứng. Ví dụ, chúng ta có thể viết truy vấn trongliệt kê 13.11 là"calendar\_interval": "1 năm"thay vì sử dụng"năm".

***13.3 Xôtổng hợp* 449**

Lưu ý rằng chúng ta không thể sử dụng bội số như5 năm(năm năm) hoặc4q(bốn phần tư) khi thiết lập khoảng thời gian bằng cách sử dụngkhoảng thời gian lịch. Ví dụ, thiết lập khoảng thời gian là"calendar\_interval": "4q"dẫn đến một ngoại lệ của trình phân tích cú pháp: "Khoảng thời gian được cung cấp [4q] không thể được phân tích cú pháp như một khoảng thời gian lịch".

Khoảng thời gian cố định (khoảng thời gian cố định) cho phép chúng ta thiết lập các khoảng thời gian dưới dạng một số đơn vị cố định, chẳng hạn như365d(365 ngày) hoặc12 giờ(12 giờ). Chúng ta có thể sử dụng các khoảng thời gian cố định này khi chúng ta không cần phải lo lắng về cài đặt lịch. Các giá trị được chấp nhận làngày(ngày),giờ(h),phút(tôi),giây(S), Vàmili giây(bệnh đa xơ cứng).

Vì fixed\_interval không biết về lịch, không giống như calendar\_ interval, không có đơn vị nào hỗ trợ tháng, năm, quý, v.v. Các thuộc tính này phụ thuộc vào lịch (mỗi tháng có một số ngày nhất định). Ví dụ, danh sách sau đây sẽ lấy tất cả các tài liệu trong 730 ngày (2 năm).

**Liệt kê 13.12 Biểu đồ tần suất với khoảng thời gian cố định là 2 năm (730ngày)**

NHẬN sách/\_tìm kiếm

{

"kích thước":0,

"aggs": { "biểu đồ ngày phát hành": {

"date\_histogram": { "field": "ngày phát hành", "fixed\_interval": "730d"

}

**Đặt khoảng thời gian cố định là 2 năm (730 ngày)**

}

}

}

Truy vấn sử dụng fixed\_interval là 730d (2 năm). Kết quả hiển thị tất cả sách trong các nhóm có chính xác 730 ngày:

{

"key\_as\_string" : "2017-12-20T00:00:00.000Z", "key" : 1513728000000,

"doc\_count" : 11

},

{

"key\_as\_string" : "2019-12-20T00:00:00.000Z", "key" : 1576800000000,

"doc\_count" : 11

},

{

"key\_as\_string" : "2021-12-19T00:00:00.000Z", "key" : 1639872000000,

"doc\_count" : 3

}

Nếu bạn tò mò, hãy chạy cùng truy vấn với hai thiết lập khác nhau:"lịch\_khoảng thời gian": "1y"Và"khoảng\_gian\_cố\_định": "365ngày". (Mã thực thi có sẵn trong tệp của sách nếu bạn muốn thử nghiệm các thiết lập này.) Sau khi các truy vấn chạy thành công, hãy kiểm tra các khóa. Trong tập kết quả đầu tiên (vớikhoảng thời gian cố định:730 ngày),

**450 CPHẦN13*Tổng hợp***

các phím bắt đầu vào ngày 1 tháng 1 năm 2005("key\_as\_string" : "2005-01-01");trong phần sau (khoảng thời gian cố định:365d), chúng bắt đầu vào ngày phát hành đầu tiên, ngày 23 tháng 12 năm 2004 ("key\_as\_string" : "2004-12-23"). Thùng thứ hai thêm 365 ngày kể từ ngày phát hành đầu tiên, tạo ra ngày 23 tháng 12 năm 2005 ("key\_as\_string" : "2005-12-23").

**GHI CHÚ**Khi chúng ta sử dụng fixed\_interval, phạm vi bắt đầu từ ngày có sẵn của tài liệu đầu tiên. Về sau, fixed\_interval được thêm vào đó. Ví dụ, nếu publish\_date của một tài liệu là 25-12-2020 và chúng ta đặt khoảng thời gian là "tháng", phạm vi bắt đầu từ 25-12-2020 và đi đến 25-01-2021, 25-02-2021, v.v.

* + 1. Tổng hợp cấp độ con

Chúng tôi đã xem xét việc phân loại dữ liệu thành các nhóm ngày trong phần trước. Ngoài việc tạo các nhóm với các phạm vi tương ứng, chúng tôi có thể muốn tổng hợp dữ liệu bên trong các nhóm đó. Ví dụ, chúng tôi có thể muốn tìm xếp hạng trung bình của một cuốn sách cho mỗi nhóm.

Để đáp ứng các yêu cầu như vậy, chúng ta có thể sử dụng một tổng hợp phụ: một tổng hợp hoạt động trên dữ liệu của nhóm. Với tổng hợp nhóm, hỗ trợ cho cả tổng hợp số liệu và tổng hợp nhóm được áp dụng ở cấp độ con. Danh sách sau đây lấy các cuốn sách được phát hành hàng năm cộng với xếp hạng trung bình cho mỗi nhóm.

**Danh sách 13.13 Số liệu trung bình về sách được phân loại theo năm**

NHẬN sách/\_tìm kiếm

{

"kích thước":0,

"aggs": { "biểu đồ ngày phát hành": {

"date\_histogram": { "field": "ngày phát hành", "calendar\_interval": "1 năm"

},

"aggs": { "xếp hạng trung bình trên mỗi nhóm": {



"trung bình": {

"trường": "amazon\_rating"

}

}

}

}

}

}

**Biểu đồ phân loại sách theo năm**

**Đặt tên chotổng hợp phụ**

**Áp dụng một số liệu giá trị đơn lẻ trên các nhóm riêng lẻ**

Có hai khối tổng hợp, một khối được đan xen bên trong khối kia. Tổng hợp bên ngoài (release\_date\_histogram) tạo ra dữ liệu dưới dạng biểu đồ dựa trên khoảng thời gian lịch của một năm. Sau đó, kết quả của tổng hợp này được đưa vào cấp tổng hợp tiếp theo: tổng hợp bên trong (avg\_rating\_per\_bucket). Tổng hợp bên trong coi mỗi bucket là phạm vi của nó và chạy tổng hợp trung bình (avg) trên đó

***13.3 Xôtổng hợp* 451**

dữ liệu. Điều này tạo ra xếp hạng trung bình của một cuốn sách trên mỗi nhóm. Hình 13.3 cho thấy kết quả mong đợi từ quá trình thực hiện tổng hợp.

**Hình 13.3 Tìm xếp hạng trung bình cho mỗi nhóm(tổng hợp phụ)**

Các khóa là những ngày tương ứng với năm dương lịch cho các tài liệu trong nhóm đó. Điều đáng chú ý về truy vấn này là một đối tượng bổ sung trong nhóm, avg\_rating\_per\_bucket, bao gồm xếp hạng sách trung bình.

* + 1. Tổng hợp phạm vi tùy chỉnh

Biểu đồ histogram cung cấp một tập hợp tự động các phạm vi cho các khoảng thời gian nhất định. Đôi khi chúng ta muốn phân tách dữ liệu thành một phạm vi không bị chi phối bởi một khoảng thời gian nghiêm ngặt (ví dụ, phân loại mọi người thành ba nhóm theo độ tuổi: 18 đến 21, 22 đến 49 và 50+). Khoảng thời gian chuẩn hóa không đáp ứng được yêu cầu này. Tất cả những gì chúng ta cần là một cơ chế đểtùy chỉnh các phạm vi—và đó là mục đích chúng tôi sử dụng tổng hợp phạm vi.

Tổng hợp phạm vi tổng hợp các tài liệu trong phạm vi do người dùng xác định. Hãy xem xét cách thức hoạt động của nó bằng cách viết truy vấn để tìm các cuốn sách thuộc hai loại xếp hạng: trên và dưới giá trị 4 (1 đến 4 và 4 đến 5).

**Liệt kê 13.14 Phân loại sách thành haigiỏ hàng**

NHẬN sách/\_tìm kiếm

{

"kích thước": 0,

"aggs": { "phạm vi xếp hạng sách":



{

"phạm vi": {

**Khai báo tổng hợp phạm vi**

**Thiết lập tùy chỉnh**

**phạm vi**

"trường": "amazon\_rating", "phạm vi": [

{

"từ": 1,

"đến": 4

**Áp dụng tổng hợp cho trường này**

},

{

**452 CPHẦN13*Tổng hợp***

"từ": 4,

"đến": 5

}

]

}

}

}

}

Truy vấn xây dựng một tổng hợp với phạm vi tùy chỉnh được xác định bởi một mảng (phạm vi) chỉ có hai nhóm: từ 1 đến 4 và từ 4 đến 5. Phản hồi sau đây chỉ ra rằng có hai cuốn sách được xếp hạng từ 1 đến 4 và phần còn lại được xếp hạng từ 4 đến 5:

"tổng hợp" : { "phạm vi xếp hạng sách" : {

"xô" : [

{

"key" : "1.0-4.0",

"từ" : 1.0,

"đến" : 4.0,

"doc\_count" : 2

},

{

"key" : "4.0-5.0",

"từ" : 4.0,

"đến" : 5.0,

"doc\_count" : 35

}

]

}

}

Tổng hợp phạm vi là một biến thể nhỏ của tổng hợp histogram và phù hợp với các phạm vi đặc biệt hoặc tùy chỉnh mà người dùng có thể cần. Tất nhiên, nếu bạn muốn sử dụng các danh mục do hệ thống cung cấp và không cần tùy chỉnh, histogram là phù hợp.

**GHI CHÚ**Tổng hợp phạm vi được tạo thành từ các thuộc tính from và to. Giá trị from được bao gồm và giá trị to bị loại trừ khi tính toán các mục bucket phù hợp với phạm vi này.

Theo cùng nguyên tắc, chúng ta có thể phân loại địa chỉ IP trong một phạm vi tùy chỉnh bằng cách sử dụng tổng hợp ip\_range chuyên dụng. Danh sách tiếp theo cho thấy điều đó. (Lưu ý rằng mã này dành cho quỷchỉ nhằm mục đích chiến lược—chúng tôi không có chỉ mục mạng được chuẩn bị bằng dữ liệu bao gồm trường localhost\_ip\_address.)

**Liệt kê 13.15 Phân loại địa chỉ IP thành haigiỏ hàng**

NHẬN mạng lưới/\_tìm kiếm

{

"aggs": {

* 1. ***Xôtổng hợp* 453**

**Phân vùng ip\_range**

"my\_ip\_addresses\_custom\_range": { "ip\_range": {

"trường": "localhost\_ip\_address",

**Chạy phạm vi**

**phân tách một số IP nhất địnhvào các thùng nhất định.**

{

"phạm vi": [

{

"đến": "192.168.0.10",

"từ": "192.168.0.20"

},

**Xác định phạm vi tùy chỉnh của các địa chỉ IP dự kiến ​​sẽ được phân loại**

**tổng hợp trên trường này (phải là loại ip)**

"đến": "192.168.0.20",

"từ": "192.168.0.100"

}

]

}

}

}

}

Như bạn có thể thấy từ tổng hợp mẫu này, chúng ta có thể phân tách các địa chỉ IP dựa trên các phạm vi tùy chỉnh của mình. Truy vấn tạo ra hai phạm vi: một phạm vi bao gồm192.168.0.10ĐẾN192.168.0.20và thứ hai với192.168.0.20ĐẾN192.168.0.100.

* + 1. Các thuật ngữ tổng hợp

Khi chúng ta muốn lấy số lượng tổng hợp của một trường nhất định, chẳng hạn như tác giả và số lượng sách của họ, thì tổng hợp thuật ngữ sẽ hữu ích. Tổng hợp thuật ngữ thu thập dữ liệu trong các nhóm cho mỗi lần xuất hiện của thuật ngữ. Ví dụ, trong truy vấn sau, tổng hợp thuật ngữ tạo một nhóm cho mỗi tác giả và số lượng sách họ đã viết.

**Liệt kê 13.16 Tổng số sách theotác giả**

NHẬN sách/\_search?size=0

{



{

"aggs": { "tác giả\_số\_sách":

"điều khoản": {

**Khai báo các loại tổng hợp thuật ngữ**

"field": "tác giả.từ khóa"

}

}

}

}

**Áp dụng tổng hợp cho trường này**

Truy vấn sử dụng thuật ngữ tổng hợp để lấy danh sách các tác giả trong chỉ mục sách và số lượng sách của họ. Phản hồi cho biết khóa là tác giả và doc\_count hiển thị số lượng sách của mỗi tác giả:

"xô" : [

{

"khóa": "Herbert Schildt", "doc\_count": 2

},

**454 CPHẦN13*Tổng hợp***

{

"key" : "Mike McGrath", "doc\_count" : 2

},

{

"key" : "Terry Norton", "doc\_count" : 2

},

{

"key" : "Adam Scott", "doc\_count" : 1

}

..

]}

Mỗi thùng đại diện cho một tác giả với số lượng sách mà tác giả đã viết. Bởitheo mặc định, thuật ngữ tổng hợp chỉ trả về 10 tổng hợp hàng đầu, nhưng chúng ta có thể điều chỉnh kích thước trả về này bằng cách đặt tham số kích thước, như danh sách sau đây hiển thị.

**Liệt kê 13.17 Ađiều khoảntruy vấn với kích thước tùy chỉnh**

NHẬN sách/\_search?size=0

{

"aggs": { "tác giả\_số\_sách": {

"điều khoản": {

"trường": "tác giả.từ khóa",

"kích thước": 25

}

}

}

}

**Đặt kích thước tổng hợp**

Tại đây, thiết lập kích thước thành 25 sẽ lấy 25 tập hợp (25 tác giả và số lượng sách của họ).

* + 1. Tổng hợp nhiều thuật ngữ

Tổng hợp multi\_terms giống với tổng hợp terms với một tính năng bổ sung: tổng hợp dữ liệu dựa trên nhiều khóa. Ví dụ, thay vì chỉ tìm số lượng sách do một tác giả viết, chúng ta có thể tìm số lượng sách có tiêu đề và tác giả cụ thể. Danh sách sau đây lấy tác giả và tiêu đề sách của họ dưới dạng bản đồ.

**Liệt kê 13.18 Tổng hợp cho các tác giả và tiêu đề sách của họ như mộtbản đồ**

NHẬN sách/\_search?size=0

{

"aggs": { "bản đồ tiêu đề tác giả": {

"nhiều\_thuật\_thuật": { "thuật\_thuật": [

{

**Tuyên bố**

**tổng hợpkiểu Tập hợp các thuật ngữ mà**

**để hình thành bản đồ tác giả/tiêu đề**

* 1. ***Cha mẹ và anh chị em ruộttổng hợp* 455**

"field": "tác giả.từ khóa"

},

{

"field": "title.keyword"

}

]

}

}

}

}

Như bạn có thể thấy, multi\_terms chấp nhận một tập hợp các thuật ngữ. Trong ví dụ, chúng tôi mong đợi Elastic-search trả về số lượng sách bằng cách sử dụng khóa tác giả/tiêu đề. Phản hồi cho biết chúng tôi có thể truy xuất thông tin này:

{

"chìa khóa" : [

"Adam Scott", "JavaScript ở khắp mọi nơi"

],

"key\_as\_string": "Adam Scott|JavaScript ở khắp mọi nơi", "doc\_count": 1

},

{

"chìa khóa" : [

"Al Sweigart",

"Tự động hóa những thứ nhàm chán với Python"

],

"key\_as\_string" : "Al Sweigart|Tự động hóa những thứ nhàm chán bằng Python", "doc\_count" : 1

},

...

Phản hồi này cho thấy hai biến thể của biểu diễn khóa: dưới dạng một tập hợp các trường (cả tác giả và tiêu đề) và dưới dạng chuỗi (key\_as\_string), chỉ đơn giản là ghép các trường lại với nhau bằng dấu phân cách (|). doc\_count chỉ ra số lượng tài liệu (sách) trong chỉ mục cho khóa đó.

Nếu bạn tò mò, hãy chạy lại truy vấn trong danh sách 13.18, lần này sử dụng các thẻ và tiêu đề làm thuật ngữ. Bạn sẽ nhận được nhiều sách trong cùng một thẻ (mã có sẵn trong các tệp của sách).

Trước khi thảo luận về loại tổng hợp thứ ba—tổng hợp pipeline—chúng ta cần hiểu khái niệm tổng hợp parent và sister. Chúng tạo thành cơ sở của tổng hợp pipeline. Chúng ta thảo luận về tổng hợp parent và sister trong phần tiếp theo và sau đó chuyển sang tổng hợp pipeline.

#### Tổng hợp cha mẹ và anh chị em

Nói chung, chúng ta có thể nhóm các tập hợp thành hai loại: tập hợp cha mẹ và tập hợp anh chị em. Bạn có thể thấy chúng hơi khó hiểu, vì vậy hãy cùng xem chúng là gì và cách sử dụng chúng.

**456 CPHẦN13*Tổng hợp***

13.4.1 Tổng hợp phụ huynh

Tổng hợp cha mẹ hoạt động trên dữ liệu đầu vào từ tổng hợp con để tạo ra các nhóm mới, sau đó được thêm vào các nhóm hiện có. Hãy xem danh sách mã sau.

**Danh sách 13.19 Cha mẹtổng hợp**

NHẬN coffee\_sales/\_search

{

"kích thước": 0,

"aggs": { "doanh số bán cà phê theo ngày": {

"date\_histogram": { "field": "ngày", "calendar\_interval": "1 ngày"

},

"aggs": { "bán cà phê cappuccino": {

"tổng": {

"field": "sales.cappuccino"

}

}

}

}

}

}

Như hình 13.4 cho thấy,cappuccino\_bántổng hợp được tạo ra như là một phần tử con của phần tử chadoanh số bán cà phê theo ngàytổng hợp. Nó ở cùng cấp độ vớibiểu đồ ngày tháng.

**Tổng hợp cha mẹ**



**Tổng hợp con**

**Hình 13.4 Tổng hợp cha mẹ được trực quan hóa**

Một tổng hợp như vậy tạo ra các thùng bên trong thùng hiện có. Hình 13.5 cho thấy kết quả này: tổng hợp cappuccino\_sales tạo ra các thùng mới được giấu bên dưới thùng date\_histogram chính.

***13.4 Tổng hợp cha mẹ và anh chị em***



**Các thùng mới sẽ được thêm vào các thùng hiện có.**

**Hình 13.5 Các thùng mới được tạo bên trongthùng hiện có**

**457**

13.4.2Tổng hợp anh chị em

Các tập hợp anh chị em tạo ra một tập hợp mới ở cùng một cấp độ. Danh sách sau đây tạo ra một tập hợp với hai truy vấn ở cùng một cấp độ (do đó, chúng tôi gọi chúng là anh chị em).

**Liệt kê 13.20 Tổng hợp anh chị em tronghoạt động**

NHẬN coffee\_sales/\_search

{

"kích thước": 0,

"aggs": { "coffee\_date\_histogram": {

"date\_histogram": { "field": "ngày", "calendar\_interval": "1 ngày"

}

},

"tổng\_số\_bán\_của\_người\_Mỹ":{"tổng": {

"trường": "sales.americano"

}

}

}

}

**458 CPHẦN13*Tổng hợp***

CácBiểu đồ ngày tháng cà phêVàtổng\_bán\_của\_người Mỹtổng hợp được định nghĩa ở cùng một cấp độ. Nếu chúng ta chụp nhanh truy vấn với tổng hợp được thu gọn, chúng trông giống như hình 13.6.

**Anh em ruộttổng hợp được định nghĩa ở cùng một cấp độ.**

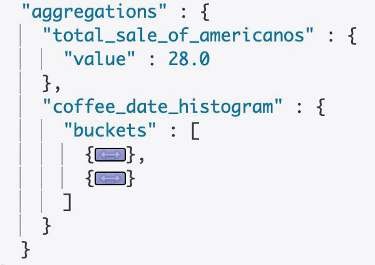


**Hình 13.6 Tổng hợp anh chị em ở phía truy vấn**

Khi chúng ta thực hiện các truy vấn anh chị em, các thùng mới được tạo ra; tuy nhiên, không giống như với các tổng hợp cha mẹ, trong đó các thùng được tạo và thêm vào các thùng hiện có, với các tổng hợp anh chị em, các tổng hợp mới hoặc các thùng mới được tạo ra ở cấp tổng hợp gốc. Truy vấn trong danh sách 13.20 tạo ra các kết quả tổng hợp trong hình

13.7 với các thùng mới được tạo cho mỗi bộ tổng hợp liên quan.

**Cả hai tổng hợp đều được xuất ra ở cùng một cấp độ.**



**Hình 13.7 Đầu ra của truy vấn anh chị emtổng hợp ở cùng một cấp độ**

#### Tổng hợp đường ống

Trong một vài phần trước, chúng ta đã xem xét việc tạo tổng hợp bằng cách tạo số liệu trên dữ liệu, phân nhóm dữ liệu hoặc cả hai. Nhưng đôi khi chúng ta muốn nối nhiều tổng hợp để tạo ra một cấp số liệu khác hoặc một nhóm. Ví dụ, giả sử chúng ta muốn tìm giá trị lớn nhất và nhỏ nhất của tất cả các nhóm được tạo ra trong quá trình tổng hợp hoặc tìm

***13.5 Đường ốngtổng hợp* 459**

trung bình động của một cửa sổ dữ liệu trượt, chẳng hạn như doanh số trung bình theo giờ trong đợt bán hàng Cyber-Monday. Tổng hợp số liệu và nhóm sẽ không cho phép chúng tôi kết nối các tổng hợp.

Elasticsearch cung cấp một tập hợp thứ ba các tổng hợp được gọi là tổng hợp đường ống cho phép kết nối tổng hợp. Các tổng hợp này hoạt động trên đầu ra của các tổng hợp khác thay vì các tài liệu hoặc trường tài liệu riêng lẻ. Nghĩa là, chúng ta tạo một tổng hợp đường ống bằng cách truyền đầu ra của một tổng hợp bucket hoặc metric. Trước khi bắt tay vào thực hiện, hãy cùng xem các loại đường ống, cú pháp của chúng và các chi tiết khác.

* + 1. Các loại tổng hợp đường ống

Nói chung, chúng ta có thể nhóm các tập hợp pipeline thành hai loại: parent và si-ling. Như đã giải thích trong phần 13.4, các tập hợp pipeline parent hoạt động trên đầu vào từ tập hợp child để tạo ra các bucket mới hoặc các tập hợp mới, sau đó được thêm vào các bucket hiện có. Các tập hợp pipeline si-ling tạo ra một tập hợp mới ở cùng cấp độ.

* + 1. Dữ liệu mẫu

Chúng tôi xem xét các kiểu tổng hợp parent và sister một cách chi tiết khi chúng tôi thực hiện một số ví dụ trong phần này. Chúng tôi sử dụng bộ dữ liệu coffee\_sales để chạy tổng hợp pipeline. Thực hiện theo quy trình lập chỉ mục dữ liệu thông thường bằng API \_bulk, như trong danh sách sau. Bạn có thể lấy dữ liệu mẫu từ kho lưu trữ của cuốn sách trên GitHub (<http://mng.bz/6D45>) hoặc trang web của cuốn sách.

**Liệt kê 13.21 Lập chỉ mục dữ liệu bằng cách sử dụng\_số lượng lớnGiao diện lập trình ứng dụng (API)**

PUT coffee\_sales/\_số lượng lớn

{"chỉ mục":{"\_id":"1"}}

{"date://2022-09-01","bán hàng":{"cappuccino":23,"latte":12,"americano":9,

➥"tea":7},"price":{"cappuccino":2.50,"latte":2.40,"americano":2.10,

➥"trà":1.50}}

{"chỉ mục":{"\_id":"2"}}

{ "date://2022-09-02","sale":{"cappuccino":40,"latte":16,"americano":19,

➥"tea":15},"price":{"cappuccino":2.50,"latte":2.40,"americano":2.10,

➥"trà":1.50}}

Thực hiện truy vấn này sẽ lập chỉ mục hai tài liệu bán hàng vào chỉ mục coffee\_sales của chúng tôi. Bước tiếp theo là tạo các tổng hợp đường ống để giúp chúng tôi hiểu chúng chi tiết.

* + 1. Cú pháp cho tổng hợp đường ống

Như đã đề cập trước đó, tổng hợp pipeline hoạt động trên dữ liệu đầu vào từ các tổng hợp khác. Vì vậy, khi khai báo pipeline, chúng ta phải cung cấp tham chiếu đến các tổng hợp số liệu hoặc bucket đó. Đối với ví dụ của chúng tôi, chúng ta có thể đặt tham chiếu là buckets\_path, bao gồm các tên tổng hợp với dấu phân cách thích hợp trong truy vấn.Biến buckets\_path là cơ chế để xác định đầu vào cho truy vấn đường ống.

**460 CPHẦN13*Tổng hợp***

Hình 13.8 cho thấy sự tổng hợp của cha mẹcappuccino\_bán. Tổng hợp đường ốngtổng\_tích\_tích, như được định nghĩa bởitổng số\_cappuccino, đề cập đến tập hợp cha mẹ thông quađường dẫn buckets, được thiết lập với giá trị tham chiếu đến tên của tập hợp cha.

**Hình 13.8 Tổng hợp đường ống cha mẹđường dẫn bucketscài đặt**



**Cáctổng\_tích\_tíchđề cập đến tập hợp cha mẹ (được xác định bởicappuccino\_bán) bằng cách thiết lậpđường dẫn bucketsĐẾNcappuccino\_bán**

Cácđường dẫn bucketsthiết lập trở nên phức tạp hơn một chút nếu tổng hợp trong trò chơi là tổng hợp anh chị em.thùng\_tối\_đatrong tổng hợp trong hình 13.9 là tổng hợp đường ống anh chị em (được định nghĩa theo thùng bán cappuccino cao nhất



**Tổng hợp anh chị em**

**Cácthùng\_tối\_đa(một tập hợp anh chị em) đề cập đến các thành phần của tập hợp anh chị em (được xác định bởibán\_bởi\_cà\_phêVàcappuccino\_bán) bằng cách thiết lậpđường dẫn bucketsĐẾNsales\_by\_coffee>sales\_cappuccino.**

**Toán tử “>” là dấu phân cách tổng hợp.**

**Cácđường dẫn bucketscho một tổng hợp đường ống anh chị em**

**Hình 13.9 Tổng hợp đường ống anh chị emđường dẫn bucketscài đặt**

***13.5 Đường ốngtổng hợp* 461**

tổng hợp), tính toán kết quả bằng cách lấy đầu vào từ các tổng hợp khác được đặt bởi biến buckets\_path. Trong trường hợp này, nó được cung cấp bởi tổng hợp được gọi làcappuccino\_sales, nằm trong tập hợp anh chị em sales\_by\_coffee.

Nếu bạn đang bối rối về buckets\_path hoặc pipeline aggregations, hãy kiên nhẫn. Chúng ta sẽ xem xét chúng trong thực tế ở một vài phần tiếp theo.

* + 1. Tổng hợp đường ống có sẵn

Biết được tổng hợp đường ống là cha hay anh chị em giúp chúng ta dễ dàng phát triển các tổng hợp này. Bảng 13.1 và 13.2 liệt kê các tổng hợp đường ống và định nghĩa của chúng.

**Bảng 13.1 Tổng hợp đường ống cha mẹ**

|  |  |
| --- | --- |
| **Tên** | **Sự miêu tả** |
| Tập lệnh thùng (buckets\_script) Bộ chọn thùng (bucket\_selector)  Sắp xếp theo thùng (xô\_sắp xếp)  Số lượng tích lũy (cumulative\_cardinality)  Tổng tích lũy (tổng\_tích\_tích)Đạo hàm (derivative)  Suy luận (suy luận)  Chức năng di chuyển (moving\_function) Di chuyển phần trăm (moving\_percentiles)  Chuẩn hóa (chuẩn hóa)  Sự khác biệt về số sê-ri (serial\_khác biệt) | Thực thi một tập lệnh trên các tập hợp nhiều thùng  Thực thi một tập lệnh để chọn thùng hiện tại cho vị trí của nó trong tổng hợp nhiều thùng  Sắp xếp các thùng  Kiểm tra gần đây đã thêm duy nhất (số lượng tích lũy)giá trị  Tìm tổng tích lũy của một số liệu  Tìm đạo hàm của một số liệu trong biểu đồ histogram hoặc biểu đồ histogram ngày  Tìm suy luận trên một mô hình được đào tạo trước Thực hiện một tập lệnh tùy chỉnh trên một cửa sổ trượt  Tương tự như moving\_function, ngoại trừ việc nó tính theo phần trăm  Tính toán giá trị chuẩn hóa của một thùng nhất định Tính toán sự khác biệt nối tiếp trên một số liệu |

**Bảng 13.2 Tổng hợp đường ống anh chị em**

|  |  |
| --- | --- |
| **Tên** | **Sự miêu tả** |
| Trung bình (avg\_bucket)  Số lượng xô(kiểm tra số lượng xô)  Tương quan xô (tương quan xô)Thay đổi điểm (thay đổi\_điểm)  Thống kê mở rộng (thống kê mở rộng) | Tính giá trị trung bình của số liệu anh chị em  Tính toán thống kê Kolmogorov–Smirnov trên một phân phối  Thực hiện một hàm tương quan  Phát hiện sự sụt giảm và thay đổi trong một số liệu Tính toán nhiều hàm thống kê |

**462 CPHẦN13*Tổng hợp***

**Bảng 13.2 Tổng hợp đường ống anh chị em (tiếp theo)**

|  |  |
| --- | --- |
| **Tên** | **Sự miêu tả** |
| Thùng tối đa (max\_bucket)Xô của tôi (my\_bucket)  Phần trămbucket (percentiles\_bucket) Thống kê bucket (thống kê\_thùng)  Thùng tổng (sum\_bucket) | Tìm nhóm có giá trị lớn nhất Tìm nhóm có giá trị nhỏ nhất Tính toán phần trăm của một số liệu Tính toán số liệu thống kê chung cho một số liệu  Tính tổng của một số liệu |

Chúng ta không thể xem xét tất cả các tổng hợp đường ống trong phần này, nhưng chúng ta có thể xem lại những điều cơ bản của tổng hợp đường ống bằng cách làm việc thông qua một vài tổng hợp phổ biến. Để bắt đầu, giả sử chúng ta muốn tìm doanh số bán cà phê tích lũy: ví dụ, có bao nhiêu tách cappuccino được bán mỗi ngày. Thay vì điểm số hàng ngày, chúng ta muốn tổng số tách cappuccino được bántừ ngày đầu tiên hoạt động, tích lũy hàng ngày. Tổng hợp cumulative\_sum là tổng hợp đường ống cha tiện dụng, lưu giữ tổng số cho ngày hiện tại, theo dõi tổng số cho ngày tiếp theo, v.v. Hãy cùng xem nó hoạt động như thế nào.

* + 1. Tổng hợp cha cumulative\_sum

Để thu thập tổng số lượng cà phê đã bán, chúng ta có thể liên kết doanh số bán cà phê theo ngày và chuyển kết quả đến tổng hợp đường ống cumulative\_sum. Danh sách sau đây sẽ lấy tổng số lượng cà phê cappuccino đã bán.

**Danh sách 13.22 Tổng doanh số bán cà phê cappuccino đã bánhằng ngày**

NHẬN coffee\_sales/\_search

{

"kích thước": 0,

"aggs": { "bán hàng của coffee": {

"date\_histogram": { "field": "ngày", "calendar\_interval": "1 ngày"

},

"aggs": { "bán cà phê cappuccino": {

"tổng": {

"field": "sales.cappuccino"

}

}, **Tổng hợp cha mẹcái đó**

**"tổng\_số\_cà ...**

**tính toán tích lũy**

**tổng doanh số bán cappuccino**

**"buckets\_path": "bán cà phê cappuccino"**

**}**

**}**

}

}

}

}

***13.5 Đường ốngtổng hợp* 463**

Tổng hợp sales\_by\_coffee là một date\_histogram cung cấp tất cả các ngày và tài liệu nằm trong những ngày đó (cho đến nay, chúng tôi chỉ có hai ngày). Chúng tôi cũng có một tổng hợp phụ (cappuccino\_sales) tổng hợp các số liệu bán hàng cho cappuccino trong nhóm đó.

Phần in đậm của mã là tổng hợp đường ống cha (total\_ cappuccinos). Nó lấy tổng doanh số cappuccino tích lũy mỗi ngày. Đây được gọi là tổng hợp đường ống cha vì nó được áp dụng trong phạm vi của đường ống cha, tổng hợp cappuccino\_ sales. Sau đây là kết quả của tổng hợp:

"tổng hợp" : { "bán hàng theo cà phê" : {

"xô" : [

{

"key\_as\_string" : "2022-09-01T00:00:00.000Z", "key" : 1661990400000,

"doc\_count" : 1, "cappuccino\_sales" : {

"giá trị" : 23.0

},

"tổng\_số\_cà ...

}

},

{

"key\_as\_string" : "2022-09-02T00:00:00.000Z", "key" : 1662076800000,

"doc\_count" : 1, "cappuccino\_sales" : {

"giá trị" : 40.0

},

"tổng\_số\_cà ...

}

}

]

}

}

Chúng ta hãy xem kết quả trong giây lát. Các thùng được phân loại theo ngày (kiểm trakey\_as\_string) do tổng hợp date\_histogram ở đầu truy vấn. Chúng tôi cũng tạo một tổng hợp phụ (cappuccino\_sales) để lấy số lượng cap-puccino được bán hàng ngày (trên mỗi thùng). Phần cuối cùng của kết quả là tổng số cappuccino tích lũy (total\_cappuccino) được thêm vào thùng hiện có. Lưu ý rằng vào ngày thứ 2, tổng số cappuccino là 63 (23 từ ngày đầu tiên và 40 từ ngày thứ hai).

Trong khi tổng tích lũy của cappuccino nằm ở cấp độ nhóm cha mẹ hiện có, thì việc tìm ra lượng cà phê tối đa hoặc tối thiểu được bán trong các nhóm lại nằm ở cấp độ anh chị em. Để làm được điều đó, chúng ta cần tạo một tập hợp ở cùng cấp độ với tập hợp chính, đó là lý do tại sao tập hợp này được gọi là anh chị em. Giả sử chúng ta muốn xác định ngày bán được nhiều cappuccino nhất hoặc ngược lại, ít cappuccino nhất

**464 CPHẦN13*Tổng hợp***

đã bán. Để làm được điều này, chúng ta cần tổng hợp đường ốngthùng\_tối\_đaVàmy\_bucket

tổng hợp, sẽ được đề cập ở phần tiếp theo.

* + 1. Tổng hợp đường ống anh chị em max\_bucket và min\_bucket

Elasticsearch cung cấp một tổng hợp đường ống gọi là max\_bucket để lấy bucket hàng đầu từ tập hợp các bucket được lấy từ các tổng hợp khác. Hãy nhớ rằng, một tổng hợp đường ống lấy đầu vào của các tổng hợp khác để tính tổng hợp của riêng nó.

**TAnh ấy tối đa****\_TỔNG HỢP BUCKET**

Truy vấn trong danh sách sau đây tăng cường tổng hợp mà chúng tôi đã thực hiện trong lần trướcphần. Nó thực hiện điều này bằng cách thêm hàm max\_bucket.

**Liệt kê 13.23 Tổng hợp đường ống để tìm doanh số bán hàngcà phê cappuccino**

NHẬN coffee\_sales/\_search

{

"kích thước": 0,

"aggs": { "bán hàng của coffee": {

"date\_histogram": { "field": "ngày", "calendar\_interval": "1 ngày"

},

"aggs": { "bán cà phê cappuccino": {

"tổng": {

"field": "sales.cappuccino"

}

}

}

},

**"thùng bán cappuccino cao nhất":{"thùng\_tối\_đa": {**

**"buckets\_path": "sales\_by\_coffee>cappuccino\_sales"**

**}**

**}**

}

}

Như bạn có thể thấy trong mã in đậm,thùng bán cappuccino cao nhấtlà tên tùy chỉnh được đặt cho tổng hợp đường ống anh chị em mà chúng ta sắp thực hiện. Chúng tôi khai báothùng\_tối\_đatổng hợp ở cùng cấp độ nhưbán\_bởi\_cà\_phêtổng hợp, vì vậy nó được gọi là tổng hợp anh chị em. Điều này mong đợi mộtđường dẫn buckets, kết hợp các tập hợpbán\_bởi\_cà\_phêVàcappuccino\_bán. (Hai kết quả này là kết quả của tổng hợp số liệu và nhóm dữ liệu.) Khi truy vấn được thực thi, chúng ta nhận được phản hồi này:

"tổng hợp" : { "bán hàng theo cà phê" : { "thùng" : [{

"key\_as\_string" : "2022-09-01T00:00:00.000Z",

***13.5 Đường ốngtổng hợp***

"chìa khóa" : 1661990400000,

"doc\_count" : 1, "cappuccino\_sales" : { "giá trị" : 23.0

},{

"key\_as\_string" : "2022-09-02T00:00:00.000Z", "key" : 1662076800000,

"doc\_count" : 1, "cappuccino\_sales" : {

"giá trị" : 40.0

}

}]

**465**

},

**"highest\_cappuccino\_sales\_bucket" : { "giá trị" : 40.0,**

**"chìa khóa" : [**

**"2022-09-02T00:00:00.000Z"**

**]**

**}**

}

Phần in đậm chứa đựngthùng bán cappuccino cao nhấtthông tin. Ngày2022-09-02(Ngày 2 tháng 9 năm 2022) là thời điểm bán được nhiều cappuccino nhất.

**TAnh Mẫn****\_TỔNG HỢP BUCKET**

Chúng ta cũng có thể lấy những ngày có ít cappuccino được bán hơn. Để làm điều này, chúng ta cần sử dụng tổng hợp đường ống min\_bucket. Thay thế mã được tô sáng trong danh sách

13.23 với mã trong đoạn mã sau:

..

"thùng bán cappuccino thấp nhất":{"min\_bucket": {

"buckets\_path": "sales\_by\_coffee>cappuccino\_sales"

}

}

Kết quả cho thấy số lượng cappuccino được bán ra ít nhất vào ngày 1 tháng 9 năm 2022:

"lowest\_cappuccino\_sales\_bucket" : { "giá trị" : 23.0,

"chìa khóa" : [

"2022-09-01T00:00:00.000Z"

]

}

Có một số tổng hợp pipeline khác như tổng hợp metric và bucket. Mặc dù không thực tế khi thảo luận tất cả chúng trong chương này, nhưng các mẫu mã của cuốn sách bao gồm hầu hết các tổng hợp. Ngoài ra, hãy kiểm tra tài liệu chính thức khi bạn làm việc với một tổng hợp cụ thể. Sau đây là liên kết đến tổng hợp pipeline:<http://mng.bz/XNzE>.

Và thế là xong phần tổng hợp! Chúng ta hãy kết thúc ở đây.

**466 CPHẦN13*Tổng hợp***

#### Bản tóm tắt

* Trong khi tìm kiếm tìm ra câu trả lời trong dữ liệu thu thập được dựa trên tiêu chí tìm kiếm, tổng hợp biên soạn các mẫu, hiểu biết sâu sắc và thông tin cho dữ liệu do các tổ chức thu thập.
* Elasticsearch cho phép chúng ta thực hiện tổng hợp lồng nhau và tổng hợp liên quan trên dữ liệu.
* Elasticsearch phân loại tổng hợp thành ba loại: số liệu, nhóm và đường ống.
* Tổng hợp số liệu lấy số liệu giá trị đơn lẻ nhưtrung bình,phútVàtối đa,tổng hợp, và vân vân.
* Tổng hợp bucket phân loại dữ liệu thành nhiều bucket khác nhau dựa trên chiến lược bucket. Với chiến lược bucket, chúng ta có thể yêu cầu Elasticsearch chia dữ liệu thành các bucket khi cần.
* Chúng ta có thể để Elasticsearch tạo các thùng được xác định trước dựa trên khoảng thời gian chúng ta cung cấp hoặc tạo các phạm vi tùy chỉnh:
  + Ví dụ, nếu khoảng cách là 10 cho một nhóm tuổi, Elasticsearch sẽ chia dữ liệu thành các bước gồm 10.
  + Nếu chúng ta muốn tạo một phạm vi như từ 10 đến 30 hoặc từ 30 đến 100, trong đó khoảng cách khác nhau, chúng ta có thể tạo một phạm vi tùy chỉnh.
* Tổng hợp đường ống hoạt động trên đầu ra từ các tổng hợp số liệu và thùng khác để tạo ra tổng hợp hoặc thùng mới.

*Sự quản lý*

***Chương này bao gồm***

* Mở rộng cụm theo chiều ngang
* Giao tiếp liên nút
* Mảnh vỡ và kích thước bản sao
* Làm việc với ảnh chụp nhanh và phục hồi
* Cấu hình nâng cao
* Hiểu vai trò chính trong một cụm

Cho đến nay, chúng ta đã thấy hoạt động bên trong của Elasticsearch, bao gồm các truy vấn tuyệt vời và các tính năng khác. Chúng ta chưa bận tâm đến các cấu hình nâng cao như cách các nút giao tiếp với nhau, các phân đoạn cần lớn đến mức nào và những thiết lập nào cần thay đổi để sửa đổi cổng Kibana. Trong chương này, chúng ta thảo luận về các tính năng quản trị này. Chúng ta giải quyết một số tính năng trong số chúng bằng cách sử dụng các truy vấn mà chúng ta đã tạo khi chạy máy chủ tìm kiếm.

Một tính năng mạnh mẽ của Elasticsearch là khả năng mở rộng máy chủ để cung cấp petabyte dữ liệu. Không có sự phức tạp nào trong việc thiết lập điều này ngoài việc mua thêm các nút. Chúng tôi sẽ đề cập đến cách mở rộng cụm trong phần đầu tiên của bài viết này

**467**

**468 CPHẦN14*Sự quản lý***

chương. Chúng tôi cũng thử nghiệm với kích thước của các phân đoạn và xem tại sao việc phân bổ nhiều bản sao hơn lại làm giảm các vấn đề về hiệu suất đọc.

Sau đó, chúng ta sẽ thảo luận về cách các nút giao tiếp nội bộ và hình thành cụm. Chúng ta sẽ xem xét các thiết lập mạng và tầm quan trọng của chúng trong phần thứ hai của chương.

Bất kỳ máy chủ nào có dữ liệu giao dịch và cấu hình đều phải được sao lưu thường xuyên để tránh mất dữ liệu trong những trường hợp không lường trước được. Elasticsearch cung cấp các chức năng để chụp nhanh dữ liệu bất kỳ lúc nào hoặc thường xuyên như chúng ta muốn và khôi phục dữ liệu khi cần. Chúng tôi thảo luận chi tiết về các chức năng chụp nhanh và khôi phục tinh vi.

Chúng tôi cũng xem xét cấu hình nâng cao để điều chỉnh các thuộc tính Elasticsearch. Chúng tôi kiểm tra tệp cấu hình elasticsearch.yml thường được sử dụng và nội dung của nó, và chúng tôi thảo luận về cách thay đổi cài đặt mạng, tăng bộ nhớ heap và kiểm tra nhật ký của một thành phần ở cấp độ theo dõi.

Cuối cùng, chúng tôi thảo luận về vai trò của cụm chủ và cách cụm đưa ra quyết định dựa trên số lượng đủ điều kiện, cùng với các chi tiết khác. Chúng tôi xem xét kịch bản chia não để hiểu số lượng tối thiểu các nút đủ điều kiện làm chủ cần thiết trong một cụm khỏe mạnh.

Việc đưa Elasticsearch vào sản xuất là một nhiệm vụ phức tạp và chuyên môn. Elasticsearch có nhiều bộ phận chuyển động và việc nắm bắt từng bộ phận là một nhiệm vụ rất lớn—nhưng không phải là không thể. Mặc dù hầu hết các tính năng của Elasticsearch đều hoạt động ngay khi xuất xưởng, nhưng điều đó là không đủ để đưa ứng dụng vào sản xuất; các tác vụ quản trị cũng phải được xử lý. Nhiều tùy chọn phải được điều chỉnh và tinh chỉnh để đưa Elasticsearch (hoặc Elastic Stack) vào trạng thái sản xuất. Việc đề cập đến tất cả các tác vụ quản trị này trong một chương sẽ rất tẻ nhạt và không thực tế. Tuy nhiên, tôi đã đưa vào các chức năng quản trị phổ biến và quan trọng mà hầu hết các nhà phát triển và quản trị viên phải hiểu. Tham khảo tài liệu phù hợp để tìm hiểu sâu hơn về các chức năng không được đề cập trong chương này. Chúng ta hãy bắt đầu bằng cách tìm hiểu về việc mở rộng quy mô của các cụm.

#### Mở rộng cụm

Các cụm Elasticsearch có thể mở rộng lên bất kỳ số lượng nút nào, từ một nút đến hàng trăm nút, dựa trên trường hợp sử dụng, dữ liệu và yêu cầu kinh doanh. Mặc dù chúng ta có thể làm việc với cụm một nút trên máy cá nhân khi tìm hiểu về Elasticsearch, nhưng chúng ta hiếm khi có cụm một nút trong sản xuất.

Một lý do chúng tôi chọn Elasticsearch là khả năng phục hồi và chịu lỗi của nó. Chúng tôi không muốn mất dữ liệu khi một nút bị sập. May mắn thay, Elasticsearch có thể đối phó với các lỗi phần cứng và phục hồi ngay khi phần cứng hoạt động trở lại. Việc chọn kích thước cụm là một chiến lược CNTT quan trọng đối với bất kỳ tổ chức nào và nhiều biến, yếu tố và đầu vào được đưa vào để định cỡ các cụm Elasticsearch cho nhu cầu dữ liệu của chúng tôi. Mặc dù chúng tôi có thể thêm tài nguyên (bộ nhớ hoặc nút mới) vào cụm hiện có,

việc dự báo những nhu cầu như vậy là rất quan trọng.

Trong phần này, chúng ta sẽ tìm hiểu cách định cỡ và mở rộng cụm. Chúng ta có thể thêm các nút vào cụm hiện có để tăng thông lượng đọc hoặc hiệu suất lập chỉ mục. Chúng ta cũng có thể giảm quy mô cụm bằng cách xóa các nút, có thể là do nhu cầu lập chỉ mục hoặc thông lượng đọc đã giảm.

***14.1 Mở rộng quy môcụm* 469**

* + 1. Thêm các nút vào cụm

Mỗi nút trong cụm về cơ bản chạy một phiên bản Elasticsearch trên một máy chủ chuyên dụng. Không có gì ngăn cản bạn tạo nhiều nút trên một máy chủ duy nhất, nhưng làm như vậy sẽ phá vỡ mục đích phục hồi dữ liệu: nếu máy chủ đó bị sập, bạn sẽ mất tất cả các nút trên máy chủ đó.

**GHI CHÚ** Nên triển khai và chạy Elasticsearch trên một máy chủ chuyên dụng có đủ sức mạnh tính toán theo yêu cầu của bạn, thay vì đóng gói nó với các ứng dụng khác, đặc biệt là các ứng dụng ngốn nhiều tài nguyên.

Khi chúng ta khởi chạy máy chủ Elasticsearch lần đầu tiên, một cụm được hình thành với một nút duy nhất. Cụm một nút này là thiết lập điển hình trong môi trường phát triển để thử nghiệm và dùng thử sản phẩm. Hình 14.1 cho thấy một cụm một nút.

**Nút này là một phiên bản đang chạy của Elasticsearch tạo thành một cụm nút đơn.**

**Nút**

**Cụm nút đơn**

**Hình 14.1 Cụm nút đơn**

Khi chúng ta đưa ra nhiều nút hơn (với điều kiện tất cả các nút đều được đặt tên bằng cùng một tên cụm), tất cả chúng sẽ tham gia để tạo thành một cụm nhiều nút. Hãy xem cách các mảnh được tạo và phân phối trên cụm khi chúng ta tiếp tục thêm các nút, từ một nút duy nhất đến, chẳng hạn, ba nút.

Giả sử chúng ta muốnđể tạo một chỉ mục có tên là chats với một shard và một bản sao cho mỗi mục. Để thực hiện việc này, chúng ta cần xác định số lượng shard và bản sao trong quá trình tạo chỉ mục bằng cách cấu hình các thiết lập trên chỉ mục.

**Liệt kê 14.1 Tạotrò chuyệnchỉ số**

ĐẶT trò chuyện

{

"cài đặt": { "số\_mảnh\_phân\_biệt": 1,

"số\_bản\_sao\_lại": 1

}

}

**470 CPHẦN14*Sự quản lý***

Tập lệnh này tạo chỉ mục trò chuyện với một phân đoạn chính trên một nút duy nhất. Elastic-search không tạo bản sao của chỉ mục này trên cùng một nút nơi phân đoạn chính tồn tại. Thật vậy, không có ích gì khi tạo ổ đĩa sao lưu ở cùng vị trí với ổ đĩa chính. Hình 14.2 cho thấy điều này (phân đoạn được tạo, nhưng bản sao thì không).

**Cụm nút đơn**

**Bản sao không được tạo trên cùng một nút (trạng thái sức khỏe = VÀNG)**

**Nút A**

Mảnh vỡ 1

Mục lục

trò chuyện

Bản sao 1

**Hình 14.2 Cụm nút đơn không có bản sao nào được tạo**

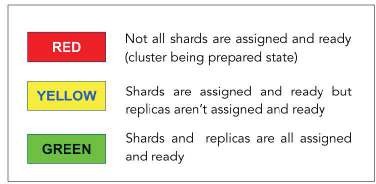
Nếu bản sao không được tạo, cụm không được phân loại là ở trạng thái khỏe mạnh. Chúng ta có thể sử dụng API sức khỏe cụm để có được chế độ xem cấp cao về trạng thái cụm. GET

Lệnh gọi \_cluster/health sẽ lấy thông tin về tình trạng của cụm và xuất ra dưới dạng JSON, trong đó nêu chi tiết các phân đoạn chưa được chỉ định, trạng thái của cụm, số lượng nút và nút dữ liệu, v.v. Nhưng chúng ta hiểu thế nào là tình trạng của cụm?

* + 1. Sức khỏe cụm

Như chúng ta đã thảo luận trong chương 3, Elasticsearch sử dụng hệ thống đèn giao thông đơn giản để cho chúng ta biết trạng thái của cụm: XANH LÁ, ĐỎ và VÀNG. Khi chúng ta lần đầu tiên tạo một chỉ mục trên máy chủ một nút, nó

sức khỏe là VÀNG vì các mảnh bản sao vẫn chưa được chỉ định (trừ khi chúng tacố ý đặt bản sao thành số không trên tất cả các chỉ mục trên nút này, điều này có thể thực hiện được nhưng là một phản mẫu). Nếu cần, hãy tham khảo phần 3.2.5 để ôn lại về hệ thống đèn giao thông phân mảnh. Hình



Không phải tất cả các phân đoạn đều được chỉ định và sẵn sàng (cụm đang được chuẩn bị).

Các mảnh đã được phân công và sẵn sàng, nhưng các bản sao chưa được phân công và sẵn sàng.

Các mảnh vỡ và bản sao đều đã được phân công và sẵn sàng.

14.3 được lặp lại từ chương 3 và

xác định tình trạng sức khỏe của cụm dựa trên các phân đoạn được chỉ định.

**Nhân vật14.3 Biểu diễn sức khỏe của một cụmsử dụng đèn giao thông**

***14.1 Mở rộng quy môcụm* 471**

Hiểu được điều này, chúng ta có thể yêu cầu Elasticsearch giải thích lý do tại sao cụm không lành mạnh (hoặc tại sao các mảnh không được chỉ định). Chúng ta có thể truy vấn máy chủ bằng API phân bổ cụm để giải thích lý do tại sao một mảnh ở trạng thái hiện tại. Ví dụ, truy vấn sau đây yêu cầu giải thích về chỉ mục trò chuyện.

**Liệt kê 14.2 Yêu cầu cụm giải thích về lỗi phân mảnh**

NHẬN \_cluster/phân bổ/giải thích

{

"index": "trò chuyện", "shard": 0, "primary": false

}

Truy vấn này trả về lời giải thích chi tiết về trạng thái của chỉ mục này. Chúng ta đã biết bản sao sẽ không được tạo hoặc chỉ định trong máy chủ một nút, đúng không? Để xác minh, hãy hỏi cụm xem có đúng như vậy không. Đoạn mã sau đây hiển thị phản hồi cô đọng từ máy chủ cho truy vấn trong danh sách 14.2:

{

"index" : "trò chuyện", "shard" : 0, "primary" : false,

"current\_state" : "chưa được gán",

"allocate\_explanation" : "không thể phân bổ vì

➥không được phép phân bổ cho bất kỳ nút nào",

"node\_allocation\_decisions": [{

...

"người quyết định" : [{

"người quyết định": "same\_shard",

"giải thích": "một bản sao của phân đoạn này đã được phân bổ cho

➥nút này ..]"}

]

...

}

Trạng thái hiện tại của chỉ mục trò chuyện được đề cập trong phản hồi trả về cho biết nó chưa được gán. Phản hồi giải thích rằng máy chủ không thể phân bổ bản sao của phân đoạn vì trình quyết định same\_shard. (Kiểm tra giá trị của mảng trình quyết định trong đoạn mã trước đó.)

Theo mặc định, nút trong cụm phải đảm nhiệm các vai trò khác nhau: master, ingest, data, ml, transform và các vai trò khác. Tuy nhiên, chúng ta có thể chỉ định các vai trò của nút bằng cách thiết lậpThuộc tính node.roles với các vai trò thích hợp trong tệp elasticsearch.yml.

Chúng ta có thể lập chỉ mục dữ liệu vào chỉ mục trò chuyện của mình và thực hiện truy vấn tìm kiếm trên phiên bản nút đơn này. Vì không có bản sao nên có nguy cơ mất dữ liệu và tạo ra các nút thắt cổ chai về hiệu suất. Để giảm thiểu rủi ro này, chúng ta có thể thêm nút(s) để mở rộng cụm.

**472 CPHẦN14*Sự quản lý***

Việc thêm một nút cũng đơn giản như khởi động Elasticsearch trên một máy khác nhưngtrong cùng một mạng có cùng cluster.name (thuộc tính trong tệp elasticsearch.yml), với điều kiện bảo mật bị vô hiệu hóa.

**CẢNH BÁO**Kể từ phiên bản 8.0, theo mặc định, cài đặt Elasticsearch cho phépsecurity, trong đó xpack.security.enabled là true. Khi bạn mở máy chủ Elasticsearch lần đầu tiên, nó sẽ tạo các khóa và mã thông báo cần thiết và hướng dẫn bạn về các bước cần thực hiện để có được kết nối Kibana thành công. Nếu bạn đang thử nghiệm Elasticsearch trên máy cục bộ của mình, bạn có thể tắt bảo mật, nhưng tôi thực sự khuyên bạn không nên nhảy vào sản xuất với thiết lập không an toàn bằng cách đặt thuộc tính xpack

.security.enabled thành false trong tệp elasticsearch.yml. Thiết lập không an toàn rất nguy hiểm và bạn đang tự chuốc lấy rắc rối.

Việc đưa ra một nút thứ hai giúp Elasticsearch tạo ra mảnh bản sao trên đó. Như hình

14.4 cho thấy, Elasticsearch ngay lập tức tạo bản sao 1 phân đoạn, là bản sao chính xác của phân đoạn 1, khi nút thứ hai bắt đầu và tham gia cụm. Nội dung của phân đoạn 1 được đồng bộ hóa với bản sao 1 ngay lập tức và sau khi đồng bộ hóa, mọi hoạt động ghi trên phân đoạn 1 đều được sao chép vào bản sao 1 trong tương lai. Điều tương tự cũng áp dụng cho nhiều phân đoạn và nhiều bản sao.

**Cụm đa nút**

**Hình 14.4 Mảnh bản sao làđược tạo trên một nút thứ hai.**

**Nút A**

**Nút B**

Bản sao 1

Mảnh vỡ 1

Nếu chúng ta thêm nhiều nút hơn vào cụm, Elasticsearch sẽ mở rộng cụm một cách tinh tế với các nút bổ sung. Nó tự động phân phối các mảnh và bản sao với việc thêm (hoặc xóa) các nút. Elasticsearch quản lý mọi thứ một cách minh bạch đằng sau hậu trường để chúng ta—người dùng hoặc quản trị viên thông thường—không phải lo lắng về cơ chế giao tiếp giữa các nút, cách các mảnh và dữ liệu của chúng được phân phối, v.v. Hình 14.5 minh họa cách một mảnh (mảnh 2) được di chuyển đến một nút thứ hai mới tham gia (do đó tạo thành một cụm nhiều nút) và các bản sao được tạo ra.

* 1. ***Mở rộng quy môcụm* 473**

**Cụm nút đơn**

**Cụm đa nút**



**Nút A**

Mảnh vỡ 2

Mảnh vỡ 1

**Không có bản sao nào được tạo trên máy chủ một nút.**

BẢN SAO 2

Bản sao 1

|  |  |  |
| --- | --- | --- |
| Bản sao 2 |  | Mảnh vỡ 2 |
|  |

**Mảnh 2 được di chuyển từNút A đến Nút B.**

**Nút A**

**Nút B**

Bản sao 1

Mảnh vỡ 1

**Hình 14.5 Một nút mới tham gia sẽ nhận được các phân đoạn mới được di chuyển từ cụm nút đơn.**

Nếu node A bị sập, bản sao 1 trên node B sẽ ngay lập tức được thăng cấp lên shard 1 và do đó quay trở lại cụm node đơn cho đến khi node thứ hai trực tuyến trở lại. Chúng ta có thể tiếp tục thêm các node mới vào cụm khi cần thiết bằng cách đưa máy chủ Elasticsearch lên node mới.

* + 1. Tăng thông lượng đọc

Việc tăng số lượng bản sao có thêm lợi ích về hiệu suất. Các bản sao tăng thông lượng đọc: các bản sao có thể phục vụ các lần đọc (truy vấn) trong khi các phân đoạn thực hiện các hoạt động lập chỉ mục. Nếu một ứng dụng có cường độ đọc cao (tức là ứng dụng nhận được khối lượng truy vấn tìm kiếm cao hơn khối lượng dữ liệu mà nó lập chỉ mục, thường là ứng dụng thương mại điện tử), thì việc tăng số lượng bản sao sẽ làm giảm tải cho ứng dụng.

Vì mỗi bản sao là một bản sao chính xác của phân đoạn, chúng ta có thể chia các phân đoạn của ứng dụng thành hai loại: phân đoạn chính xử lý dữ liệu và phân đoạn bản sao xử lý phía đọc của dữ liệu. Khi một truy vấn đến từ máy khách, nút điều phối sẽ chuyển hướng yêu cầu đó đến các nút đọc để trả lời. Do đó, việc thêm nhiều bản sao hơn vào chỉ mục giúp cải thiện thông lượng đọc. Tuy nhiên, hãy lưu ý đến các tác động về bộ nhớ: vì mỗi bản sao là một bản sao chính xác của phân đoạn, chúng ta cần định cỡ cụm cho phù hợp (xem phần 14.3).

Để tăng số bản sao (và do đó là thông lượng đọc) để chống lại tình trạng tắc nghẽn hiệu suất truy vấn đọc, một chiến lược là cập nhật thiết lập number\_of\_replicas trên chỉ mục trực tiếp. Đây là thiết lập động mà chúng ta có thể điều chỉnh ngay cả khi chỉ mục đang trực tiếp và đang trong quá trình sản xuất. Ví dụ, giả sử chúng ta thêm 10 nút vào cụm 5 nút của mình, với năm phân đoạn để chống lại các vấn đề về hiệu suất truy vấn đọc đang khiến máy chủ

**474 CPHẦN14*Sự quản lý***

đầu gối của nó. Như danh sách sau đây cho thấy, chúng ta có thể tăng số lượng bản sao để tăng cài đặt bản sao trên chỉ mục trực tiếp.

**Liệt kê 14.3 Tăng số lượng bản sao trên chỉ mục trực tiếp**

PUT trò chuyện/\_cài đặt

{

"số\_bản\_sao\_lại": 10

}

Các bản sao bổ sung này là cho mỗi phân đoạn và được tạo trên các nút mới hình thành với dữ liệu được sao chép vào chúng. Bây giờ chúng ta có các bản sao bổ sung trong kho vũ khí của mình, chúng xử lý mọi truy vấn đọc một cách hiệu quả. Elasticsearch có trách nhiệm định tuyến các yêu cầu của khách hàng đến các bản sao, do đó cải thiện hiệu suất tìm kiếm và truy vấn của ứng dụng.

**GHI CHÚ** Tóm tắt nhanh về việc thay đổi kích thước các phân đoạn chính: sau khi một chỉ mục được tạo và hoạt động, chúng ta không thể thay đổi kích thước vì number\_of\_shards là thuộc tính tĩnh của chỉ mục. Nếu chúng ta phải thay đổi cài đặt này, chúng ta phải đóng chỉ mục, tạo chỉ mục mới với kích thước mới và lập chỉ mục lại dữ liệu của chúng ta từ chỉ mục cũ sang chỉ mục mới.

Mặc dù việc tăng số bản sao đọc sẽ làm tăng thông lượng đọc, nhưng nó sẽ làm căng bộ nhớ và không gian đĩa của cụm. Đó là vì mỗi bản sao tiêu thụ nhiều tài nguyên như bản sao tương ứng (phân đoạn chính).

#### Giao tiếp nút

Elasticsearch ẩn đi những chi tiết nhỏ nhặt về những gì diễn ra đằng sau hậu trường, từ việc khởi động nút đến việc tạo cụm, lập chỉ mục dữ liệu, sao lưu và ảnh chụp nhanh và truy vấn. Việc thêm nút mở rộng cụm và mang lại cho chúng ta lợi ích về khả năng phục hồi ngay từ đầu. Chúng ta đã làm việc thông qua rất nhiều API trong các chương trước và các API đó cho phép giao tiếp giữa máy khách và máy chủ. Phương tiện giao tiếp dự kiến ​​trong trường hợp này là giao diện HTTP qua API RESTful. Mặt khác của đồng xu là giao tiếp giữa các nút: cách mỗi nút giao tiếp với các nút khác, cách máy chủ đưa ra quyết định trên toàn cụm, v.v. Đối với điều này, Elasticsearch sử dụng hai loại giao tiếp:

* Giao diện HTTP cho các tương tác giữa máy khách và nút sử dụng API RESTful. (Chúng tôi đã xem xét những điều này khi chạy truy vấn.)
* Giao diện lớp vận chuyển cho việc truyền thông giữa các nút.

Theo mặc định, cụm được hiển thị trên cổng 9200 để giao tiếp HTTP (hoặc HTTPS), mặc dù chúng ta có thể thay đổi điều này bằng cách điều chỉnh tệp cấu hình (elastic-search.yml). Mặt khác, lớp vận chuyển được đặt thành cổng 9300, nghĩa là giao tiếp giữa các nút diễn ra trên cổng đó. Cả hai giao diện đều được đặt trong tệp cấu hình cho từng nút trong thuộc tính mạng, nhưng chúng ta có thể thay đổi điều này dựa trên yêu cầu của mình.

* 1. ***Mảnh vỡkích thước* 475**

Khi chúng ta khởi động Elasticsearch trên một máy, nó sẽ liên kết với localhost theo mặc định. Chúng ta có thể thay đổi liên kết này thành một địa chỉ mạng cụ thể bằng cách thay đổi network.host và network.port (cũng như transport.port cho mạng node-to-node) nếu cần.

Việc thay đổi các thiết lập này trên một trang trại máy tính là một điều phiền toái, đặc biệt là khi bạn cần thiết lập một cụm với hàng trăm nút. Hãy đảm bảo rằng bạn có các tập lệnh quản lý tiện dụng để giảm bớt sự phiền toái này. Một cách lý tưởng là tạo cấu hình trongthư mục trung tâm và trỏ biến ES\_PATH\_CONF tới các thiết lập đó. (Chúng ta cũng có thể sử dụng Ansible, Azure Pipelines, GitOps, v.v. cho các mục đích như vậy.) Việc xuất biến này cho phép Elasticsearch chọn cấu hình từ thư mục này.

Quay lại việc thiết lập các thuộc tính mạng, chúng ta có thể sử dụng các giá trị đặc biệt trong tệp cấu hình để thiết lập máy chủ mạng thay vì cấu hình máy chủ theo cách thủ công. Thiết lậpthuộc tính network.host thành \_local\_ cho phép Elasticsearch tự động đặt địa chỉ của nó. Điều này đặt địa chỉ vòng lặp (127.0.0.1) làm máy chủ mạng. Giá trị đặc biệt \_local\_ là giá trị mặc định cho thuộc tính network.host. Đề xuất của tôi là để nguyên nó.

Ngoài ra còn có một\_địa điểm\_giá trị thiết lậpmạng.máy chủthuộc tính cho các địa chỉ cục bộ của site (ví dụ: 192.168.0.1). Chúng ta có thể thiết lậpmạng.máy chủ, mặc định những giá trị đặc biệt này bằng cách thiết lậpmạng.máy chủ: [\_local\_, \_site\_]trong cấu hình.

#### Kích thước mảnh vỡ

Một câu hỏi luôn nảy sinh khi chúng ta nói về shard là kích thước. Hãy xem xét chi tiết về kích thước shard, đặc biệt là dấu chân bộ nhớ mà chúng ta cần xem xét và các biến khác, để hiểu rõ hơn về chủ đề này. Để lập luận, chúng ta tập trung vào trường hợp sử dụng với một chỉ mục trên cụm năm nút và sau đó tìm kích thước shard với nhiều chỉ mục trên cụm.

* + 1. Thiết lập mộtchỉ số

Giả sử chúng ta có một cụm 5 nút với một chỉ mục bao gồm 10 phân đoạn chính và 2 bản sao. Điều này có nghĩa là mỗi phân đoạn có 2 bản sao và tổng số phân đoạn là 30 cho cả nút chính và bản sao. Hình 14.6 cho thấy cấu hình này.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bản sao 3\_1 Bản sao 4\_1  Mảnh vỡ 1  Mảnh vỡ 2  Bản sao 9\_2 Bản sao 10\_2 | Bản sao 1\_1 Bản sao 2\_1  Mảnh vỡ 3  Mảnh vỡ 4  Bản sao 5\_1 Bản sao 6\_1 | Bản sao 3\_2 Bản sao 4\_2  Mảnh vỡ 5  Mảnh vỡ 6  Bản sao 7\_1 Bản sao 8\_1 | Bản sao 5\_2 Bản sao 6\_2  Mảnh vỡ 7  Mảnh vỡ 8  Bản sao 9\_1 Bản sao 10\_1 | Bản sao 1\_2 Bản sao 2\_2  Mảnh vỡ 9  Mảnh vỡ 10  Bản sao 7\_2 Bản sao 8\_2 |

NútAnode BNode CNode DNode E

**Hình 14.6 Các mảnh vỡ và bản sao trải rộng trên một cụm nhiều nút**

**476 CPHẦN14*Sự quản lý***

Chúng ta cũng giả sử rằng các phân đoạn chính bao gồm một vài triệu tài liệu chiếm 300 GB bộ nhớ. Do đó, chúng ta tạo một chỉ mục gồm 10 phân đoạn, với mỗi phân đoạn được phân bổ khoảng 50 GB bộ nhớ. Elasticsearch phân phối 300 GB dữ liệu trong số 10 phân đoạn, do đó mỗi phân đoạn nhận được khoảng 30 GB tài liệu. Chúng ta cũng có 2 bản sao cho mỗi phân đoạn, tổng cộng là 20 bản sao.

Bản sao cần được thiết lập với cùng bộ nhớ như phân đoạn chính vì chúng là bản sao của phân đoạn. Do đó, 20 bản sao tiêu thụ 20 lần 50 GB hoặc 1000 GB. Và đừng quên thêm bộ nhớ được phân bổ cho các phân đoạn chính: 10 phân đoạn nhân 50 GB là 500 GB. Chúng ta cần ít nhất 1500 GB (1,5 TB) để chạy cụm này với một chỉ mục. Hình 14.7 cho thấy các phép tính này.

Kích thước bộ nhớ cụm cho**một**chỉ mục (10 mảnh, 2 bản sao)

**Chỉ số đơn với 10 mảnh**

Số lượng chỉ mục

Số mảnh trên mỗi chỉ mục Bộ nhớ của một mảnh Tổng bộ nhớ của mảnh

Số lượng bản sao Tổng số bản sao

Tổng bộ nhớ sao chép

Tổng bộ nhớ cho mỗi nút

= 1 chỉ số

= 10 mảnh vỡ

= 50GB

= 10 × 50GB

= 500GB

= 2 bản sao/mảnh

= 20

= 20 × 50GB

= 1000GB

= 500GB + 1000GB

= 1,5 TB

**Tổng bộ nhớ**

**1,5 TB cho mỗi nút cho một chỉ mục**

**Hình 14.7 Kích thước bộ nhớcủa một chỉ mục duy nhất với 10 mảnh và 2 bản sao**

Cụm này được tạo thành từ các nút riêng lẻ. Hãy nhớ rằng, các nút này cần thêm bộ nhớ cho mục đích vận hành, bao gồm các chỉ mục hệ thống, cấu trúc dữ liệu trong bộ nhớ và các hoạt động khác. Vì vậy, luôn nên thêm bộ nhớ ngoài bộ nhớ được phân bổ cho kích thước phân mảnh. Như đã đề cập, chúng tôi đang xây dựng một cụm năm nút, vì vậy mỗi nút có 400 GB sẽ tạo thành cụm có 2000 GB. Điều này sẽ đủ cho trường hợp sử dụng hiện tại.

* + 1. Thiết lập nhiều chỉ mục

Trong ví dụ của chúng tôi, chúng tôi chỉ có một chỉ mục để quản lý, vì vậy chúng tôi cố gắng tính toán chi phí bộ nhớ dựa trên chỉ mục duy nhất đó. Điều này hiếm khi xảy ra trong thế giới thực. Bất kỳ số lượng chỉ mục nào cũng có thể tồn tại trên máy chủ và ít nhất, chúng tôi cần cung cấp máy chủ để tạo nhiều chỉ mục trong tương lai. Nếu chúng tôi ngoại suy các phép tính chi phí cho năm chỉ mục, với mỗi chỉ mục có 10 phân đoạn và 2 bản sao cho mỗi chỉ mục, hình 14.8 chỉ ra tổng bộ nhớ của chúng tôi.

* 1. ***Ảnh chụp nhanh* 477**

Kích thước bộ nhớ cụm cho**năm**chỉ mục (10 mảnh, 2 bản sao)

**Năm chỉ mục với 50 mảnh vỡ**

Số lượng chỉ mục

Tổng số chỉ số

Bộ nhớ của một mảnh Tổng bộ nhớ mảnh

Số lượng bản sao Tổng số bản sao

Tổng bộ nhớ bản sao Tổng bộ nhớ trên mỗi nút

= 5 chỉ số

= 5 × 10

= 50 mảnh vỡ

= 50GB

= 50 × 50GB

= 2,5 TB

= 2 bản sao/mảnh

= 2 × 50

= 100 bản sao

= 100 × 50GB

= 5 TB

= 2,5 TB + 5 TB

= 7,5 TB

**7,5 TB cho mỗi nút cho năm chỉ mục**

**Hình 14.8 Sử dụng bộ nhớ theo cấp số nhân cho năm chỉ mục trên một nút**

Như bạn có thể thấy, bộ nhớ tăng theo cấp số nhân khi chúng ta xem xét các chỉ mục bổ sung. Trong trường hợp trước đó của một chỉ mục với 10 phân đoạn và 20 bản sao (hình 14.7), chúng tôi đã ủy quyền cho một cụm 5 nút để xử lý 2000 GB yêu cầu không gian. Trong trường hợp mới này với nhiều chỉ mục (hình 14.8), chúng tôi cần một cụm lớn có thể xử lý khoảng 10 TB không gian. Chúng tôi có thể xử lý vấn đề này theo hai cách: mở rộng theo chiều dọc hoặc mở rộng theo chiều ngang.

**V.QUY MÔ ERTICAL**

Chúng ta có thể sử dụng cùng một cụm phục vụ cho yêu cầu bộ nhớ 2 TB và hỗ trợ bộ nhớ bổ sung để xử lý các yêu cầu bộ nhớ mới: ví dụ, tăng mỗi máy chủ lên 2 TB (năm nút với 2 TB = 10 TB). Đây là hoạt động mở rộng theo chiều dọc. Mặc dù đây không phải là vấn đề ở cấp độ kỹ thuật, nhưng chúng ta có thể cần phải hạ cấp máy chủ để nâng cấp.

**HCHIỀU HƯỚNG HƯỚNG**

Cách tiếp cận thay thế (và có lẽ được ưa thích hơn) là thêm các nút vào cụm. Ví dụ, chúng ta có thể thêm 20 nút nữa vào máy chủ, tổng cộng là 25 nút. Sau đó, chúng ta có một cụm mới được hình thành với 25 nút, trong đó mỗi nút có 400 GB bộ nhớ: tổng số có thể xử lý yêu cầu bộ nhớ 10 TB của chúng ta.

Mặc dù không có giải pháp nào phù hợp với tất cả mọi người, nhưng việc có một chiến lược có tư duy tiến bộ với cách tiếp cận đã được thử nghiệm và kiểm tra sẽ hiệu quả đối với hầu hết các tổ chức. Việc định cỡ phân mảnh rất tẻ nhạt và chúng ta cần hết sức thận trọng để định cỡ chúng một cách phù hợp.

Vì yêu cầu quản trị chính là sao lưu chỉ mục hoặc toàn bộ cụm theo thời gian, chúng tôi sẽ đề cập đến điều này trong phần tiếp theo. Ngoài ra, chúng tôi sẽ xem xét khôi phục bản sao lưu khi cần. Elasticsearch cung cấp một cơ chế gọn gàng để sao lưu và khôi phục—ảnh chụp nhanh.

#### 14.4Snapsh ots

Chạy ứng dụng trong môi trường sản xuất mà không có chức năng sao lưu và khôi phục là rất rủi ro. Dữ liệu trong cụm của chúng ta phải được lưu trữ trong bộ lưu trữ bền ở đâu đó ngoài cụm. May mắn thay, Elasticsearch cung cấp chức năng chụp nhanh và khôi phục dễ dàng để sao lưu dữ liệu của chúng ta và khôi phục khi cần.

**478 CPHẦN14*Sự quản lý***

Snapshot giúp lưu trữ các bản sao lưu gia tăng thường xuyên. Chúng ta có thể lưu trữ các snapshot trong một kho lưu trữ, thường được gắn trên một hệ thống tệp cục bộ hoặc một dịch vụ dựa trên đám mây như AWS S3, Microsoft Azure hoặc Google Cloud Platform. Như hình 14.9 cho thấy, các quản trị viên snapshot các cụm thường xuyên vào một phương tiện lưu trữ và sau đó khôi phục chúng theo yêu cầu.

**Chụp ảnh nhanh kịp thời**

**Ảnh chụp nhanh được lưu trữ trên các ổ đĩa bền: ổ đĩa lưu trữ đám mây hoặc ổ đĩa hệ thống tập tin.**



**Nút A**

**Nút B**

**Nút C**

Kho lưu trữ ảnh chụp nhanh

**Cụm**

**Ảnh chụp nhanh được khôi phụctheo yêu cầu.**

**Hình 14.9 Cơ chế chụp nhanh và khôi phục trên cụm**

Ảnh chụp cụm thông thường là một tác vụ quản trị lý tưởng nhất là nên được tự động hóa bằng các tập lệnh nội bộ hoặc các công cụ tiện dụng. Tuy nhiên, trước khi chúng ta bắt đầu sao lưu ảnh chụp nhanh, chúng ta phải đảm bảo rằng kho lưu trữ ảnh chụp nhanh của chúng ta là một loại cụ thể và đã được đăng ký. Phần này thảo luận về cơ chế thiết lập kho lưu trữ và cách chụp nhanh và khôi phục dữ liệu từ kho lưu trữ vào cụm.

* + 1. Bắt đầu

Cần thực hiện một vài bước trước khi chúng ta có thể sử dụng chức năng snapshot và restore. Nói chung, chúng ta cần thực hiện ba hoạt động sau:

* + - * *Đăng ký kho lưu trữ ảnh chụp nhanh.*Ảnh chụp nhanh được lưu trữ trên các vùng lưu trữ bền vững như hệ thống tệp, Hệ thống tệp phân tán Hadoop (HDFS) hoặc lưu trữ đám mây như kho lưu trữ Amazon S3.
      * *Chụp nhanh dữ liệu.*Sau khi đã đăng ký kho lưu trữ với cụm, chúng ta có thể chụp nhanh dữ liệu để sao lưu.
      * *Khôi phục từ cửa hàng.*Khi cần khôi phục dữ liệu, chúng ta chỉ cần chọn chỉ mục, tập hợp chỉ mục hoặc toàn bộ cụm cần khôi phục và bắt đầu hoạt động khôi phục từ kho lưu trữ ảnh chụp nhanh đã đăng ký trước đó.

***14.4 Ảnh chụp nhanh* 479**

Là một phần của snapshot, tất cả các chỉ mục, tất cả các luồng dữ liệu và toàn bộ trạng thái cụm đều được sao lưu. Lưu ý rằng sau khi snapshot đầu tiên được thực hiện, các bản sao lưu tiếp theo sẽ là các bản cập nhật gia tăng, không phải bản sao đầy đủ. Có hai cách chúng ta có thể làm việc với snapshot:

* + - * + Chụp nhanh và khôi phục API RESTful
        + Tính năng chụp nhanh và khôi phục Kibana

Bước đầu tiên là chọn loại kho lưu trữ và đăng ký nó. Chúng ta hãy sử dụng cả hai cách tiếp cận để đăng ký kho lưu trữ ảnh chụp nhanh, như đã thảo luận trong phần tiếp theo.

* + 1. Đăng ký kho lưu trữ ảnh chụp nhanh

Để đơn giản, hãy chọn hệ thống tệp làm loại kho lưu trữ của chúng ta: chúng ta muốn lưu trữ ảnh chụp nhanh của mình trên đĩa được gắn trên hệ thống tệp được chia sẻ. Chúng ta bắt đầu bằng cách gắn hệ thống tệp với bộ nhớ khả dụng trên tất cả các nút chính và dữ liệu trong cụm. Sau khi máy chủ đã gắn hệ thống tệp này, chúng ta cần cho Elasticsearch biết vị trí của nó bằng cách chỉ định vị trí đó trong tệp cấu hình.

Chỉnh sửa tệp cấu hình elasticsearch.yml để sửa đổi thuộc tính path.repo, trỏ nó đến vị trí gắn kết. Ví dụ, nếu đường dẫn gắn kết là /volumes/ es\_snapshots, path.repo trông như thế này: path.repo: /volumes/es\_snapshots. Sau khi thêm đường dẫn gắn kết, chúng ta cần khởi động lại các nút tương ứng để gắn kết này khả dụng cho các nút.

**RĐĂNG KÝ KHO LƯU TRỮ BẰNG SNAPSHOT****Giao diện lập trình ứng dụng (API)S**

Khi các nút trực tuyến trở lại sau khi khởi động lại, bước cuối cùng là gọi API kho lưu trữ snap-shot. Danh sách sau đây hiển thị mã.

**Liệt kê 14.4 Đăng ký kho lưu trữ ảnh chụp nhanh dựa trên hệ thống tập tin**

ĐẶT \_snapshot/es\_cluster\_snapshot\_repository

**Đặt tên cho kho lưu trữ**

{

"kiểu": "fs", "cài đặt": {

**Đặt loại kho lưu trữ thành hệ thống tập tin ("fs")**

**cung cấp cho**

**\_ảnh chụp nhanhĐiểm cuối API**

"vị trí": "/volumes/es\_snapshots"

}

}

**Xác định vị trí của kho lưu trữ như một hệ thống tập tin được gắn kết**

Elasticsearch cung cấp API \_snapshot để thực hiện các hành động liên quan đến snapshot và khôi phục. Trong danh sách 14.4, chúng tôi tạo một kho lưu trữ snapshot có tên là es\_cluster\_ snapshot\_repository. Nội dung của yêu cầu mong đợi loại kho lưu trữ mà chúng tôi đang tạo và các thuộc tính cần thiết để đặt loại kho lưu trữ. Trong ví dụ của chúng tôi, chúng tôi đặt "fs" (cho hệ thống tệp) làm loại kho lưu trữ của chúng tôi và cung cấp đường dẫn hệ thống tệp làm "vị trí" trong đối tượng cài đặt.

Bởi vì chúng ta đã thêm điểm gắn kết vào tệp cấu hình và tất nhiên là đã khởi động lại nút, nên mã trong danh sách 14.4 sẽ thực thi thành công để đăng ký.là kho lưu trữ đầu tiên của chúng tôi. Phát hành lệnh GET \_snapshot sẽ trả về ảnh chụp nhanh đã đăng ký:

**480 CPHẦN14*Sự quản lý***

{

"es\_cluster\_snapshot\_repository" : { "kiểu" : "fs",

"cài đặt" : {

"vị trí" : "/volumes/es\_snapshots"

}

}

}

Phản hồi cho thấy có một kho lưu trữ ảnh chụp nhanh đã được đăng ký và có sẵn cho các ảnh chụp nhanh của chúng tôi.

**GHI CHÚ**Nếu bạn đang chạy Elasticsearch trên máy cục bộ của mình, bạn có thể đặt thư mục tạm thời làm vị trí kho lưu trữ. Ví dụ: bạn có thể sử dụng /tmp/ es\_snapshots cho hệ điều hành dựa trên \*nix hoặc c:/temp/es\_snapshots cho Windows.

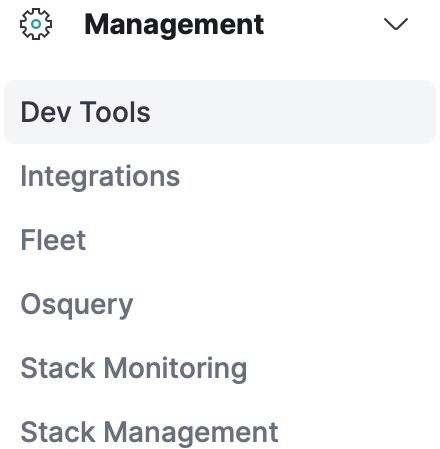
Như đã đề cập, chúng ta có thể sử dụng Kibana's Console để làm việc với tính năng snapshot và khôi phục. Như chúng ta đã làm với API, chúng ta cũng có thể đăng ký kho lưu trữ. Mặc dù chi tiết về cách làm việc với Kibana nằm ngoài phạm vi của cuốn sách này, tôi cung cấp một vài gợi ý để bạn có thể làm việc với chức năng snapshot và khôi phục trên Kibana.

**RĐĂNG KÝ KHO ẢNH CHỤP TRÊN****KNHỮNG ĐỨA TRẺ**

Kibana có hỗ trợ rộng rãi cho việc làm việc với tính năng snapshot và restore, bao gồm đăng ký snapshot, thực thi snapshot và khôi phục snapshot. Hãy cùng xem cách chúng ta có thể đăng ký repository trên Kibana.

Đi đến menu trên cùng bên trái của Kibana Console và mở rộng menu Management, tại đó bạn sẽ thấy, cùng với Dev Tools, một liên kết điều hướng Stack Management (xem hình 14.10). Nhấp vào liên kết để điều hướng đến trang Stack Management. Sau đó, chọn mục menu Data > Snapshot and Restore. Trang kết quả cung cấp các kho lưu trữ hiện tại, snapshot và trạng thái của chúng.

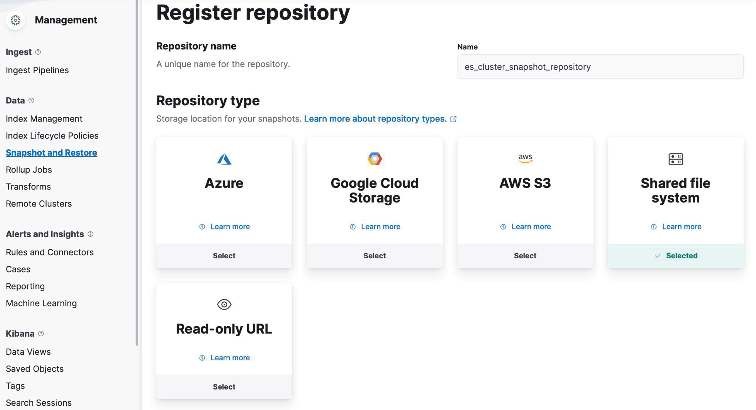
**Chức năng chụp nhanh và khôi phục được ẩn trong liên kết Quản lý ngăn xếp.**



**Hình 14.10 Truy cập trang Quản lý ngăn xếp để sử dụng chức năng chụp nhanh**

***14.4 Ảnh chụp nhanh* 481**

Đi đến tab thứ hai, Repositories, và nhấp vào Register a Repository. Trang tiếp theo được hiển thị trong hình 14.11.



**Chọn loại**

**của kho lưu trữ.**

**Đặt tên cho**

**kho lưu trữ.**

**Chúng tôi đã chọn một**

**kho lưu trữ dựa trên hệ thống tập tin tại đây.**

**Hình 14.11 Đặt tên cho kho lưu trữ và chọn loại của nó**

Đặt tên cho kho lưu trữ là bắt buộc, sau đó chọn loại: hệ thống tệp được chia sẻ (fs), AWS S3, lưu trữ blob của Azure, v.v. Tiếp tục với ví dụ về việc chọn hệ thống tệp cục bộ làm kho lưu trữ của chúng tôi, hãy nhấp vào Hệ thống tệp được chia sẻ để điều hướng đến trang tiếp theo. Tại đó, hãy nhập vị trí hệ thống tệp và các thuộc tính được yêu cầu khác (số byte tối đa và tối thiểu mỗi giây cho ảnh chụp nhanh vàkhôi phục, kích thước khối, v.v.). Nhấp vào nút Đăng ký ở cuối trang để tạo kho lưu trữ này.

* + 1. Tạo ảnh chụp nhanh

Bây giờ chúng ta đã trải qua quá trình đăng ký kho lưu trữ ảnh chụp nhanh, bước tiếp theo là tạo ảnh chụp nhanh để dữ liệu được sao lưu vào kho lưu trữ mà chúng ta vừa tạo. Có một số cách để tạo ảnh chụp nhanh. Hãy bắt đầu bằng một kỹ thuật thủ công đơn giản: sử dụng API \_snapshot.

**Liệt kê 14.5 Tạo ảnh chụp nhanhbằng tay**

ĐẶT \_snapshot/es\_cluster\_snapshot\_repository/prod\_snapshot\_oct22

Chúng tôi yêu cầu\_ảnh chụp nhanhAPI để tạo ảnh chụp nhanh có tênprod\_snapshot\_tháng mười 22dưới kho lưu trữkho lưu trữ es\_cluster\_snapshot.Bản chụp nhanh thủ công một lần này sẽ sao lưu toàn bộ dữ liệu (chỉ mục, luồng dữ liệu và thông tin cụm) vào bản chụp nhanh trên đĩa trong hệ thống tệp của kho lưu trữ.

**482 CPHẦN14*Sự quản lý***

Chúng ta cũng có thể chụp nhanh tùy chỉnh một vài chỉ mục, thay vì toàn bộ dữ liệu như chúng ta vừa làm. Đính kèm một phần thân yêu cầu vào danh sách 14.5 sẽ chỉ định một tập hợp các chỉ mục: chẳng hạn như tất cả các bộ phim và tất cả các bài đánh giá.

**Liệt kê 14.6 Tạo ảnh chụp nhanh với cácchỉ số**

ĐẶT \_snapshot/es\_cluster\_snapshot\_repository/custom\_prod\_snapshots

{

"chỉ số": ["\*phim\*","\*đánh giá\*"]

}

**Sao lưu tất cả các phimvà các chỉ số liên quan đến đánh giá**

Thuộc tính chỉ mục lấy một chuỗi hoặc mảng các chuỗi biểu diễn tập hợp các chỉ mục cụ thể mà chúng ta muốn sao lưu. Trong ví dụ của chúng ta, chúng ta sao lưu bất kỳ chỉ mục nào với mẫu glob gồm \*movies\* và \*reviews\*. Theo mặc định, tất cả các chỉ mục và luồng dữ liệu đều được bao gồm ([\*]) nếu chúng ta không chỉ định những gì chúng ta muốn sao lưu. Để bỏ qua một số, chúng ta có thể sử dụng mẫu với dấu trừ (hoặc dấu gạch ngang), như thế này: -\*.old. Trong trường hợp của chúng ta, mẫu này bỏ qua tất cả các chỉ mục kết thúc bằng .old.

Chúng ta cũng có thể đính kèm các thuộc tính do người dùng xác định trong thuộc tính siêu dữ liệu. Ví dụ, chúng ta muốn ghi chú chi tiết sự cố của yêu cầu của người dùng khi chụp ảnh nhanh. Danh sách sau đây hiển thị điều này dưới dạng truy vấn.

**Liệt kê 14.7 Thêm chi tiết tùy chỉnh vàoảnh chụp nhanh**

ĐẶT \_snapshot/es\_cluster\_snapshot\_repository/prod\_snapshots\_with\_metadata

{

"chỉ số": ["\*phim\*","\*đánh giá\*", "-\*.cũ"],

**Bao gồm hoặc loại trừ những điều này**

"siêu dữ liệu":{

"lý do":"yêu cầu của người dùng", "incident\_id":"ID12345", "người dùng":"mkonda"

**Xác định khối thông tin tùy chỉnh theo đối tượng siêu dữ liệu**

**chỉ mục trong ảnh chụp nhanh**

}

}

Chúng tôi cải thiện danh sách các chỉ mục bằng cách xóa các chỉ mục "cũ" như một phần của quy trình chụp nhanh. Chúng tôi cũng thêm siêu dữ liệu với thông tin yêu cầu của người dùng và chúng tôi có thể tạo càng nhiều chi tiết càng tốt trong đối tượng này. Bước cuối cùng trong vòng đời của chức năng chụp nhanh và khôi phục của Elasticsearch là khôi phục các ảnh chụp nhanh, như chúng tôi sẽ thảo luận trong phần tiếp theo.

* + 1. Khôi phục ảnh chụp nhanh

Việc khôi phục ảnh chụp nhanh khá đơn giản. Tất cả những gì chúng ta cần làm là gọi

\_khôi phụctrên\_ảnh chụp nhanhGiao diện lập trình ứng dụng (API).

**Liệt kê 14.8 Khôi phục dữ liệu từ mộtảnh chụp nhanh**

ĐẶT \_snapshot/es\_cluster\_snapshot\_repository/custom\_prod\_snapshots/\_restore

***14.4 Ảnh chụp nhanh* 483**

\_Khôi phụcđiểm cuối sao chép dữ liệu từ kho lưu trữ vào cụm. Tất nhiên, chúng ta có thể đính kèm một đối tượng JSON để chỉ định thêm chi tiết về chỉ mục hoặc luồng dữ liệu nào mà chúng ta muốn khôi phục. Truy vấn sau đây cung cấp một ví dụ về yêu cầu như vậy.

**Liệt kê 14.9 Khôi phục chỉ mục từ mộtảnh chụp nhanh**

ĐĂNG \_snapshot/es\_cluster\_snapshot\_repository/custom\_prod\_snapshots/\_restore

{

"chỉ số":["phim\_mới"]

}

**Liệt kê các chỉ mục cần khôi phục từ ảnh chụp nhanh**

* + 1. Xóa ảnh chụp nhanh

Chúng ta không cần phải giữ snapshot trên đĩa mọi lúc. Một chiến lược mà hầu hết các tổ chức đều theo là tạo snapshot cho từng chỉ mục dựa trên yêu cầu của người dùng. Chúng ta có thể cần cập nhật bản đồ hoặc thay đổi các phân đoạn chính của một chỉ mục nhất định. Thật không may, chúng ta không thể làm điều này miễn là

vì chỉ mục đang ở trạng thái hoạt động.

Xóa chỉ mục và ảnh chụp nhanh cũ.

Cách tiếp cận tốt nhất là tạo một chỉ mục mới với các phân đoạn và ánh xạ thích hợp, sau đó chụp nhanh chỉ mục hiện tại, khôi phục chỉ mục đó vào chỉ mục mới được tạo từ ảnh chụp nhanh và xóa ảnh chụp nhanh. Hình 14.12 cho thấy hoạt động này.

Tạo một chỉ mục mới với các phân đoạn và ánh xạ được cập nhật.

Tạo ảnh chụp nhanh từ chỉ mục cũ.

Khôi phục ảnh chụp nhanh vào một chỉ mục mới.

Chúng ta có thể di chuyển dữ liệu từ một chỉ mục cũ sang một chỉ mục mới bằng

**Hình 14.12 Vòng đời ảnh chụp nhanh, từ khi tạo đến khi xóa**

chức năng chụp nhanh và khôi phục. Sau khi sử dụng ảnh chụp nhanh, chúng ta có thể xóa chúng để giải phóng dung lượng lưu trữ.

Việc xóa snapshot khá đơn giản: chúng ta sử dụng hành động HTTP DELETE, cung cấp ID snapshot. Danh sách sau đây xóa snapshot mà chúng ta đã tạo trước đó.

**Liệt kê 14.10 Xóa mộtảnh chụp nhanh**

XÓA \_snapshot/es\_cluster\_snapshot\_repository/custom\_prod\_snapshots

Nếu chúng ta đưa ra lệnh hành động HTTP DELETE trong khi ảnh chụp nhanh đang được tiến hành, Elasticsearch sẽ ngay lập tức dừng hoạt động, xóa ảnh chụp nhanh và xóa nội dung khỏi kho lưu trữ.

* + 1. Tự động hóa ảnh chụp nhanh

Chúng ta vừa thấy cơ chế tạo ảnh chụp nhanh, nhưng đây là ảnh chụp nhanh tùy ý: chúng ta tạo ảnh chụp nhanh theo yêu cầu (ví dụ: khi di chuyển dữ liệu, triển khai bản phát hành hotfix sản xuất, v.v.). Tuy nhiên, chúng ta có thể tự động hóa quy trình này để sao lưu thường xuyên

**484 CPHẦN14*Sự quản lý***

và chụp ảnh nhanh bằng tính năng quản lý vòng đời ảnh chụp nhanh (SLM). Elastic-search cung cấp API \_slm để quản lý vòng đời của ảnh chụp nhanh và tạo chính sách vòng đời được thực thi theo lịch trình được xác định trước.

Để thực hiện việc này, chúng ta có thể tạo chính sách bằng API \_slm. Chính sách này chứa thông tin như các chỉ mục cần sao lưu, lịch trình (cron job), thời gian lưu giữ, v.v. Chúng ta phải đăng ký kho lưu trữ (xem phần 14.3.2), vì đây là điều kiện tiên quyết để quản lý vòng đời ảnh chụp nhanh.

Giả sử chúng ta muốn sao lưu tất cả các chỉ mục phim của mình vào lúc nửa đêm mỗi đêm vào một kho lưu trữ nhất định. Chúng ta cũng cần lưu giữ các ảnh chụp nhanh này trong một tuần. Chúng ta có thể viết một chính sách và tạo tự động hóa bằng cách sử dụng API \_slm.

**Liệt kê 14.11 Tạo chính sách cho lịch trìnhảnh chụp nhanh**

**API \_slm mong đợi**

**Đăng ký**

ĐẶT \_slm/policy/prod\_cluster\_daily\_backups

{

"tên":"<prod\_daily\_backups-{now/d}>","lịch trình": "0 0 0 \* \* ?",

**Mộtđịnh danh chính sách.**

**Tên duy nhất được đặt cho ảnh chụp nhanh**

**Lịch trình công việc Cron**

**kho lưu trữ**

"kho lưu trữ": "es\_cluster\_snapshot\_repository", "config": {

**vào nửa đêm**

"chỉ số":["\*phim\*", "\*đánh giá\*"], "include\_global\_state": sai

},

**Chỉ mục để chụp nhanh**

**Thêm trạng thái cụm vào ảnh chụp nhanh**

"giữ lại":{

"Hết hạn sau":"7 ngày"

}

}

**Giữ lại ảnh chụp nhanhtrong một tuần (7 ngày)**

API \_slm tạo ra một chính sách được lưu trữ trong cụm để thực thi khi lịch trình bắt đầu. Chúng ta phải cung cấp ba phần: tên duy nhất, lịch trình và kho lưu trữ mà chúng ta đã đăng ký trước đó để lưu trữ ảnh chụp nhanh. Hãy xem xét chi tiết các bit này. Tên duy nhất (<prod\_daily\_backups-{now/d}> trong danh sách 14.11) là tên được xây dựng với toán học dữ liệu trong đó. Trong trường hợp này, <prod\_daily\_backups-{now/d}> làđược phân tích thành prod\_daily\_backups-5.10.2022 nếu nó được chạy vào ngày 5 tháng 10 năm 2022, bởi vì

{now/d} chỉ ra ngày hiện tại. Mỗi lần lịch trình bắt đầu, một tên duy nhất mới được tạo ra với ngày hiện tại: prod\_daily\_backups-6.10.2022, prod\_ daily\_backups-7.10.2022, v.v. Vì chúng tôi sử dụng toán học ngày trong tên, chúng tôiphải bao quanh tên bằng dấu ngoặc nhọn (< >) để trình phân tích cú pháp phân tích mà không có bất kỳ vấn đề nào. Tham khảo tài liệu Elasticsearch (<http://mng.bz/QPwe>) để biết thêm chi tiết về phép tính ngày tháng trong tên.

Như danh sách 14.11 cho thấy, chúng tôi đã cung cấp một lịch trình dưới dạng công việc cron: "schedule": 0 0 0 \* \* ?. Biểu thức cron này nêu rằng công việc phải được thực hiện chính xác vào lúc nửa đêm. Do đó, chúng ta có thể mong đợi quy trình chụp nhanh của mình bắt đầu vào lúc nửa đêm mỗi đêm.

Khối cấu hình trong danh sách 14.11 bao gồm các chỉ mục và trạng thái cụm mà chúng tôi muốn sao lưu (trong ví dụ này, tất cả các chỉ mục liên quan đến phim và đánh giá). Nếu chúng tôi không

***14.4 Ảnh chụp nhanh* 485**

bao gồm khối cấu hình, tất cả các chỉ mục và luồng dữ liệu được bao gồm trong ảnh chụp nhanhups theo mặc định. Thuộc tính include\_global\_state chỉ ra rằng chúng ta muốn bao gồm trạng thái cụm trong ảnh chụp nhanh. Trong danh sách, chúng ta bỏ qua trạng thái cụm (include\_ global\_state được đặt thành false) như một phần của ảnh chụp nhanh.

Phần cuối cùng là thông tin lưu giữ ("lưu giữ":), chỉ định thời gian chúng ta muốn giữ ảnh chụp nhanh trong kho lưu trữ. Chúng ta thiết lập thời gian lưu giữ ảnh chụp nhanh hiện tạithời gian tồn tại là một tuần bằng cách đặt thuộc tính expire\_after thành 7 ngày.

Khi chúng tôi thực hiện truy vấn này, tiện ích chụp nhanh tự động vẫn được duy trì cho đến khi chúng tôi xóa chính sách. Nó được thực hiện dựa trên lịch trình. Đây là cách dễ dàng và được ưa chuộng hơn để sao lưu toàn bộ cụm mà không cần can thiệp thủ công.

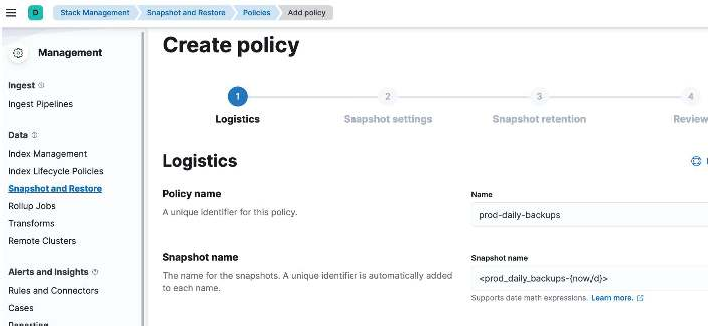
**SLMSỬ DỤNGKNHỮNG ĐỨA TRẺ**

Chúng ta cũng có thể tạo chính sách SLM bằng Kibana. Hãy cùng xem sơ qua cách thực hiện:

**1** Trong Kibana, điều hướng đến liên kết tính năng Quản lý > Ảnh chụp nhanh và Khôi phục.

**2**Nhấp vào tab Chính sách và tạo chính sách mới bằng cách nhấp vào nút Tạo chính sách.

**3** Điền thông tin chi tiết của trang này như thể hiện trong hình 14.13.



**Tên của chính sách**

**ID duy nhất của Snapshot**

**Kho lưu trữ ảnh chụp nhanh**

**Biểu thức Cron**

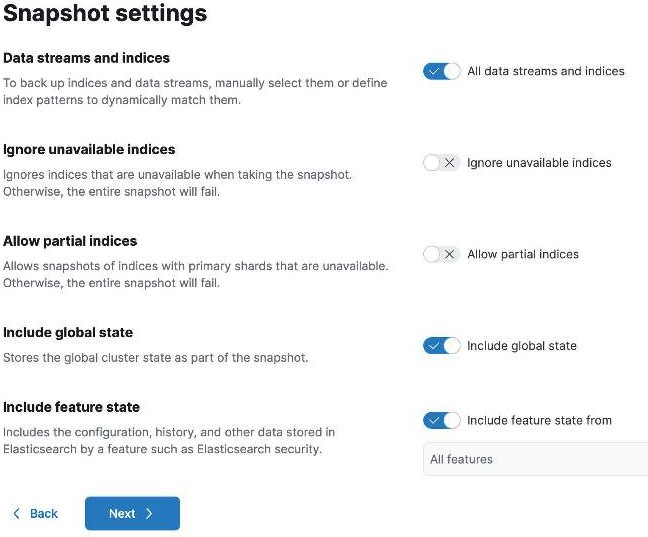
**Hình 14.13 Tạo chính sách SLM bằng Bảng điều khiển Kibana**

**4**Điều hướng đến trang tiếp theo bằng cách nhấp vào nút Tiếp theo. Điền thông tin chi tiết liên quan đến khối cấu hình của truy vấn trong danh sách 14.11, bao gồm bất kỳ chỉ mục và luồng dữ liệu cụ thể nào (hoặc tất cả), trạng thái cụm toàn cầu (hoặc không), v.v. Hình

14.14 hiển thị cấu hình Cài đặt ảnh chụp nhanh.

**486 CPHẦN14*Sự quản lý***

**Chỉ định tất cả hoặc các chỉ mục và luồng dữ liệu cụ thể.**



**Chúng ta có muốn trạng thái cụm được sao lưu không?**

**Hình 14.14Cấu hình cài đặt ảnh chụp nhanh**

**5**Nhấp vào Tiếp theo và điền thông tin chi tiết về việc lưu giữ. Chúng tôi có thể cung cấp ba thông tin (tất cả đều là tùy chọn) để dọn dẹp ảnh chụp nhanh dựa trên chính sách lưu giữ. Trong hình 14.15, chúng tôi yêu cầu người quản lý ảnh chụp nhanh xóa ảnh chụp nhanh này sau một tuần (bảy ngày). Chúng tôi cũng chỉ định rằng ít nhất ba ảnh chụp nhanh phải luôn được



**Cung cấp số lượng tối đa**

**ngày (thời gian) trước khi ảnh chụp nhanh bị xóa.**

**Chúng tôi muốn giữ bao nhiêu bản sao**

**thậm chí sau thời hạn bảy ngày**

**Số lượng bản sao tối đa**

**chúng tôi muốn giữ lại ngay cả khi chúng chưa quá bảy ngày tuổi**

**Hình 14.15 Cấu hình cài đặt lưu giữ ảnh chụp nhanh**

* 1. ***Ảnh chụp nhanh* 487**

có sẵn trong kho lưu trữ của chúng tôi, vì vậy chúng không bao giờ bị xóa hết. Thiết lập minimum\_count này đảm bảo rằng ba ảnh chụp nhanh đó không bao giờ bị xóa, ngay cả khi chúng cũ hơn một tuần. Tương tự, thiết lập maximum\_count đảm bảo rằng chúng tôi không có nhiều hơn số lượng bản sao ảnh chụp nhanh đã cho (6, trong trường hợp này), ngay cả khi chúng cũ hơn bảy ngày.

**6** Xem lại các tùy chọn và tạo chính sách SLM.

**Tôi****THỰC HIỆN HÀNG NĂMSLM**

Chúng ta không cần phải đợi đến thời gian đã lên lịch trong chính sách để khởi động hành động chụp nhanh. Nếu chúng ta có các ảnh chụp nhanh hàng tuần được lên lịch trong chính sách và cần sao lưu do bản sửa lỗi sản xuất, chúng ta có thể sử dụng khởi động thủ công. Danh sách sau đây hiển thị cách thực hiện chính sách SLM theo cách thủ công bằng cách gọi điểm cuối \_execute trên API.

**Liệt kê 14.12 Thực hiện thủ công theo lịch trìnhảnh chụp nhanh**

POST \_slm/policy/prod\_cluster\_daily\_backups/\_execute

Chạy lệnh này sẽ bắt đầu tạo trước đóprod\_cluster\_daily\_backups

chính sách ngay lập tức. Chúng ta không cần phải chờ đợi đến thời điểm đã định.

**Ảnh chụp nhanh có thể tìm kiếm (Phiên bản doanh nghiệp)**

Trong phiên bản 7.12, Elasticsearch đã giới thiệu một tính năng hoàn toàn mới có tên là*ảnh chụp nhanh có thể tìm kiếm.*Chức năng này giúp chạy truy vấn tìm kiếm trên snapshot. Ý tưởng là sử dụng bản sao lưu làm chỉ mục cho một số truy vấn nhất định. Vì snapshot có thể được ghi vào kho lưu trữ giá rẻ, không chỉ sử dụng chúng để khôi phục dữ liệu mà còn gắn chúng hiệu quả làm chỉ mục để chạy truy vấn tìm kiếm là một chiến thắng lớn.

Chúng tôi biết rằng việc thêm bản sao vào cụm là một cách để cải thiện hiệu suất đọc,nhưng có một chi phí liên quan: bản sao tốn thời gian và tiền bạc do không gian bổ sung mà chúng yêu cầu. Bằng cách tạo ảnh chụp nhanh có thể gắn kết (sử dụng\_núiAPI), chúng có sẵn-có khả năng tìm kiếm, thay thế hiệu quả các bản sao và do đó giảm chi phí của chúng tôi xuống gần một nửa.

Tính năng snapshot có thể tìm kiếm chỉ khả dụng cho giấy phép Enterprise và không miễn phí với giấy phép cơ bản. Vì lý do này, chúng tôi không đề cập đến chúng trong cuốn sách này. Nếu bạn quan tâm, hãy xem tài liệu để biết chi tiết về cách triển khai snapshot có thể tìm kiếm (<http://mng.bz/XN7M>).

Elasticsearch hoạt động theo quy ước hơn là cấu hình, giúp chúng ta có ít quyết định hơnđể thực hiện khi làm việc với thiết lập hoặc chạy Elasticsearch và khía cạnh hoạt động của nó. Tuy nhiên, chạy hệ thống theo mặc định sẽ gây ra rắc rối. Chúng ta phải điều chỉnh cấu hình khi cần để cung cấp thêm bộ nhớ hoặc cải thiện hiệu suất. Chúng ta sẽ thảo luận về các thiết lập cấp cao, ý nghĩa của chúng và cách chúng ta có thể thay đổi chúng trong phần tiếp theo.

**488 CPHẦN14*Sự quản lý***

#### Trình độ cao cấu hình

Elasticsearch có nhiều thiết lập và cấu hình có thể làm khó ngay cả các kỹ sư chuyên nghiệp. Mặc dù nó sử dụng mô hình quy ước trên cấu hình và hoạt động với các mặc định hầu hết thời gian, nhưng việc tùy chỉnh cấu hình trước khi đưa ứng dụng vào sản xuất là điều bắt buộc.

Trong phần này, chúng tôi sẽ giới thiệu một số thuộc tính thuộc nhiều danh mục khác nhau và thảo luận về tầm quan trọng của chúng cũng như cách tinh chỉnh chúng. Chúng ta có thể sửa đổi ba tệp cấu hình:

* *elasticsearch.yml*—Tệp cấu hình được chỉnh sửa phổ biến nhất, nơi chúng ta có thể đặt tên cụm, thông tin nút, đường dẫn dữ liệu và nhật ký, cũng như cài đặt mạng và bảo mật
* *log4j2.thuộc tính*—Nơi chúng ta có thể thiết lập mức ghi nhật ký của nút Elasticsearch
* *jvm.tùy chọn*—Nơi chúng ta có thể thiết lập bộ nhớ heap của nút đang chạy

Các tệp này được nút Elasticsearch đọc từ thư mục cấu hình, đây là một thư mục trong thư mục cài đặt Elasticsearch. Thư mục này mặc định là

$ES\_HOME/config cho các bản cài đặt nhị phân (zip hoặc tar.gz) (biến ES\_HOME trỏ đến thư mục cài đặt Elasticsearch). Nếu chúng ta cài đặt bằng trình quản lý gói như Debian hoặc bản phân phối RPM, mặc định là /etc/elasticsearch/config.

Nếu chúng ta mong đợi truy cập các tệp cấu hình của mình từ một thư mục khác, chúng ta có thể đặt và xuất một biến đường dẫn có tên là ES\_PATH\_CONF, biến này trỏ đến vị trí tệp cấu hình mới. Trong một vài tiểu mục tiếp theo, chúng ta sẽ xem xét một số cài đặt quan trọng mà cả quản trị viên và nhà phát triển đều cần hiểu.

* + 1. Tệp cấu hình chính

Mặc dù những người ở Elastic đã phát triển Elasticsearch để chạy với các giá trị mặc định (quy ước thay vì cấu hình), chúng ta rất khó có thể dựa vào các giá trị mặc định khi đưa nút vào sản xuất. Chúng ta nên điều chỉnh các thuộc tính để thiết lập thông tin mạng cụ thể, đường dẫn dữ liệu hoặc nhật ký, khía cạnh bảo mật, v.v. Để làm như vậy, chúng ta có thể sửa đổi tệp elastic-search.yml để thiết lập hầu hết các thuộc tính bắt buộc cho các ứng dụng đang chạy của mình.

Chúng tôi đã đề cập sơ qua về các thuộc tính mạng ở đầu chương này khi thảo luận về giao tiếp nút. Elasticsearch phơi bày các thuộc tính mạng dưới dạng các thuộc tính network.\*. Chúng ta có thể đặt tên máy chủ và số cổng bằng thuộc tính này. Ví dụ, chúng ta có thể thay đổi số cổng của Elasticsearch thành 9900 thay vì giữ nguyênđến cổng mặc định 9200: http.port :9900. Chúng ta cũng có thể thiết lập transport.port để thay đổi cổng mà các nút giao tiếp nội bộ.

Bạn có thể cần thay đổi nhiều thuộc tính, tùy thuộc vào yêu cầu của bạn. Tham khảo tài liệu chính thức để tìm hiểu thêm về các thuộc tính này một cách chi tiết:[http://](http://mng.bz/yQNE) [mng.bz/yQNE](http://mng.bz/yQNE).

* + 1. Tùy chọn ghi nhật ký

Elasticsearch được phát triển bằng Java và giống như hầu hết các ứng dụng Java, nó sử dụng Log4j 2 làm thư viện ghi nhật ký. Một nút đang chạy sẽ đưa thông tin ghi nhật ký ở cấp độ INFO vào bảng điều khiển và tệp (sử dụng Kibana Console và trình thêm tệp lăn tương ứng).

* 1. ***Trình độ caocấu hình* 489**

Tệp thuộc tính Log4j (log4j2.properties) bao gồm các biến hệ thống (sys:es.logs.base\_path, sys:es.logs.cluster\_name, v.v.) được giải quyết tại thời điểm chạy ứng dụng. Vì Elasticsearch phơi bày các thuộc tính này nên chúng có sẵn cho Log4j, cho phép Log4j thiết lập vị trí thư mục tệp nhật ký, mẫu tệp nhật ký,và các thuộc tính khác. Ví dụ, sys:es.logs.base\_path trỏ đến đường dẫn mà Elasticsearch ghi nhật ký, giải quyết đến thư mục $ES\_HOME/logs.

Theo mặc định, hầu hết Elasticsearch chạy ở cấp INFO, nhưng chúng ta có thể tùy chỉnh cài đặt dựa trên từng gói. Ví dụ, chúng ta có thể chỉnh sửa tệp log4j2.properties và thêm trình ghi nhật ký cho gói chỉ mục, như danh sách sau đây hiển thị.

**Liệt kê 14.13 Thiết lập mức ghi nhật ký cho một mục cụ thểbưu kiện**

logger.index.name = org.elasticsearch.index logger.index.level = GỠ LỖI

Thực hiện thao tác này cho phép gói chỉ mục xuất nhật ký ở cấp độ DEBUG. Thay vì chỉnh sửa tệp này trên một nút cụ thể và khởi động lại nút đó (chúng ta có thể cần thực hiện thao tác này cho mọi nút nếu chưa thực hiện trước khi tạo cụm được quản lý), chúng ta có thể đặt cấp độ nhật ký DEBUG ở cấp độ cụm cho gói này.

**Liệt kê 14.14 Thiết lập mức nhật ký tạm thời trên toàn cầu**

ĐẶT \_cluster/cài đặt

{

"tạm thời": {

**Sự tạm thời**

**cài đặt Thiết lập gói chỉ mục này**

"logger.org.elasticsearch.index":"GỠ LỖI"

}

}

**mức ghi nhật ký để DEBUG**

Như truy vấn cho thấy, chúng tôi đặt thuộc tính cấp độ logger cho gói chỉ mục thành DEBUG trong khối tạm thời. Khối tạm thời cho biết thuộc tính này không bền (chỉ khả dụng khi cụm đang hoạt động). Nếu chúng tôi khởi động lại cụm hoặc cụm bị sập, cài đặt sẽ bị mất vì nó không được lưu trữ vĩnh viễn trên đĩa.

Chúng ta có thể thiết lập thuộc tính này bằng cách gọi tới API cài đặt cụm (\_cluster/cài đặt), như danh sách 14.14 cho thấy. Khi thuộc tính này được thiết lập, bất kỳ thông tin ghi nhật ký nào khácliên quan đếnchỉ sốtrongorg.elasticsearch.indexgói nguồn được xuất ra tạiGỠ LỖImức độ.

Elasticsearch cũng cung cấp một phương tiện lưu trữ các thuộc tính cụm một cách lâu dài. Để lưu trữ các thuộc tính vĩnh viễn, chúng ta có thể sử dụng khối liên tục. Danh sách sau đâythay thế khối tạm thời bằng khối cố định.

**Liệt kê 14.15 Thiết lập mức nhật ký vĩnh viễn**

ĐẶT \_cluster/cài đặt

{

"dai dẳng": { "logger.org.elasticsearch.index":"GỠ LỖI",

**490 CPHẦN14*Sự quản lý***

"logger.org.elasticsearch.http":"DẤU VẾT"

}

}

Mã này thiết lậpGỠ LỖImức độ trênorg.elasticsearch.indexgói vàDẤU VẾTmức độ trênorg.elasticsearch.httpgói. Vì cả hai đều là thuộc tính cố định nên trình ghi nhật ký sẽ ghi nhật ký chi tiết ở các cấp độ này theo thiết lập trên các gói, các thuộc tính vẫn tồn tại sau khi cụm khởi động lại (hoặc gặp sự cố).

Hãy cẩn thận khi thiết lập các thuộc tính như vậy vĩnh viễn bằng thuộc tính liên tục. Tôi đề xuất bật mức ghi nhật ký DEBUG hoặc TRACE khi khắc phục sự cố hoặc trong quá trình gỡ lỗi. Khi bạn hoàn tất việc xử lý tập trong quá trình sản xuất, hãy đặt lại mức thành INFO để tránh ghi hàng loạt yêu cầu vào đĩa.

* + 1. Java tùy chọn máy ảo

Vì Elasticsearch sử dụng ngôn ngữ lập trình Java, nhiều tinh chỉnh tối ưu hóa có thể được thực hiện ở cấp máy ảo Java (JVM). Vì những lý do hiển nhiên, việc thảo luận về một chủ đề lớn như vậy trong cuốn sách này sẽ không công bằng. Tuy nhiên, nếu bạn tò mò và muốn hiểu JVM hoặc tinh chỉnh hiệu suất ở cấp độ thấp hơn, hãy tham khảo một cuốn sách như Optimizing Java (Ben Evan và Jame Gough, 2018, O'Reilly) hoặc Java Performance (Scott Oaks, 2020, O'Reilly). Tôi thực sự khuyên bạn nên đọc chúng vì chúng cung cấp các nguyên tắc cơ bản cùng với các mẹo và thủ thuật vận hành.

Elasticsearch cung cấp tệp jvm.options trong thư mục /config với các thiết lập JVM. Tuy nhiên, tệp này chỉ dùng cho mục đích tham khảo (ví dụ: để kiểm tra thiết lập bộ nhớ của nút) và không bao giờ được chỉnh sửa. Bộ nhớ heap được tự động thiết lập cho máy chủ Elasticsearch dựa trên bộ nhớ khả dụng của nút.

**CẢNH BÁO**Không bao giờ chỉnh sửa tệp jvm.options trong bất kỳ trường hợp nào. Làm như vậy có thể làm hỏng hoạt động bên trong của Elasticsearch.

Để nâng cấp bộ nhớ hoặc thay đổi cài đặt JVM, chúng ta phải tạo một tệp mới với phần mở rộng tên tệp là .options, cung cấp các tham số điều chỉnh phù hợp và đặt tệp vào thư mục có tên là jvm.options.d trong thư mục cấu hình để cài đặt lưu trữ (tar hoặc zip). Chúng ta có thể đặt bất kỳ tên nào cho tệp tùy chỉnh, nhưng chúng ta phải bao gồm tên cố định

Phần mở rộng .options.

Đối với cài đặt gói RPM/Debian, tệp này phải nằm trong thư mục /etc/elastic search/jvm.options.d/. Tương tự, hãy gắn tệp options vào thư mục /usr/ share/elasticsearch/config đối với cài đặt Docker.

Chúng ta có thể chỉnh sửa các thiết lập trong tệp tùy chọn JVM tùy chỉnh này. Ví dụ, để nâng cấp bộ nhớ heap trong tệp có tên jvm\_custom.options, chúng ta có thể sử dụng mã sau.

**Liệt kê 14.16 Nâng cấp bộ nhớ heap**

-Xms4g

-Xmx8g

* 1. ***Cụmbậc thầy* 491**

Cờ -Xms thiết lập bộ nhớ heap ban đầu và -Xmx điều chỉnh bộ nhớ heap tối đa. Quy tắc bất thành văn là không để cài đặt -Xms và -Xmx vượt quá 50% tổng RAM của nút. Apache Lucene chạy ngầm sử dụng nửa bộ nhớ còn lại cho phân đoạn, lưu trữ đệm và các quy trình khác.

Như chúng ta đã biết, Elasticsearch là một cụm phân tán với một nút chính điều khiển cụm trong khi các nút còn lại thực hiện các công việc riêng lẻ của chúng. Rất nhiều suy nghĩ đã được đưa vàothiết kế và phát triển nút chính cùng các tính năng của nó, và chúng tôi sẽ thảo luận về nó trong phần tiếp theo.

#### Cụm chủ

Mỗi nút trong cụm có thể có nhiều vai trò được chỉ định: master, data, ingest, ml (học máy) và các vai trò khác. Việc chỉ định vai trò master cho biết nút này là nút đủ điều kiện master. Trước khi thảo luận về điều kiện master, hãy xem xét tầm quan trọng của master node.

* + 1. Các nút chính

Master node chịu trách nhiệm cho các hoạt động trên toàn cụm như phân bổ các mảnh cho các nút, quản lý chỉ mục và các hoạt động nhẹ khác. Master node là thành phần quan trọng chịu trách nhiệm duy trì cụm khỏe mạnh. Nó cố gắng giữ nguyên trạng thái của cụm và cộng đồng nút. Chỉ có một master node tồn tại cho một cụm và công việc duy nhất của nó là lo lắng về các hoạt động của cụm—không hơn, không kém.

Các nút đủ điều kiện làm chủ là các nút được gắn thẻ với vai trò chủ. Việc chỉ định vai trò chủ cho một nút không có nghĩa là nút đó trở thành chủ cụm, nhưng nó sẽ tiến gần hơn một bước đến việc trở thành chủ cụm, nếu chủ được bầu gặp sự cố. Các nút đủ điều kiện làm chủ khác cũng sẽ trở thành chủ, nếu có cơ hội, vì vậy chúng cũng tiến gần hơn một bước đến việc trở thành chủ.

Bạn có thể hỏi, nút đủ điều kiện làm chủ có tác dụng gì? Mỗi nút đủ điều kiện làm chủthực hiện quyền bỏ phiếu để chọn master của cụm. Đằng sau hậu trường, khi chúng ta khởi động các nút lần đầu tiên để tạo cụm hoặc khi master chết, một trong những bước đầu tiên là bầu master. Chúng ta sẽ xem xét các cuộc bầu cử cho cụm master trong phần sau.

* + 1. Bầu cử tổng thống

Cluster master được bầu chọn theo hình thức dân chủ thông qua bầu cử! Một cuộc bầu cử được tổ chức để chọn master khi cluster được thành lập lần đầu tiên hoặc khi master hiện tại chết. Nếu master bị sập vì bất kỳ lý do gì, các node đủ điều kiện làm master sẽ yêu cầu bầu cử. Các thành viên bỏ phiếu để bầu một master mới. Sau khi được bầu, master node sẽ đảm nhiệm nhiệm vụ quản lý cluster.

Không phải ngày nào cũng là ngày vui—những tình huống ngoài tầm kiểm soát có thể đánh bật một nút chủ. Do đó, các nút đủ điều kiện làm chủ liên tục giao tiếp với nút chủ để đảm bảo nút chủ còn hoạt động và thông báo cho nút chủ về trạng thái của chúng. Khi nút chủ không còn nữa, công việc trước mắt của các nút đủ điều kiện làm chủ là kêu gọi bầu cử để bầu ra một nút chủ mới.

**492 CPHẦN14*Sự quản lý***

Một số thuộc tính, chẳng hạn nhưcụm.bầu cử.thời gianVàcụm.bầu cử

.initial\_timeout, giúp chúng tôi cấu hình tần suất bầu cử và thời gian chờ đợitrước khi các nút đủ điều kiện chủ gọi cho một cuộc bầu cử. Ví dụ, thuộc tính initial\_timeout là khoảng thời gian mà một nút đủ điều kiện chủ đợi trước khi gọi cho một cuộc bầu cử. Theo mặc định, giá trị này được đặt thành 500 ms. Ví dụ, giả sử nút đủ điều kiện chủ A không nhận được nhịp tim từ nút chủ trong 500 ms. Sau đó, nó gọi cho một cuộc bầu cử vì nó nghĩ rằng nút chủ đã bị sập.

Ngoài việc bầu ra một master, các node đủ điều kiện master sẽ làm việc cùng nhau để đưa các hoạt động cụm vào hoạt động. Mặc dù master là vua của cụm, nhưng nó cần sự hỗ trợ và mua vào từ các node đủ điều kiện master. Nhiệm vụ của master là duy trì và quản lý trạng thái cụm, như chúng ta thấy tiếp theo.

* + 1. Trạng thái cụm

Trạng thái cụm bao gồm tất cả siêu dữ liệu về các phân đoạn, bản sao, lược đồ, lệnh map-ping, thông tin trường, v.v. Các chi tiết này được lưu trữ dưới dạng trạng thái toàn cục trong cụm và cũng được ghi vào từng nút. Nút master là nút duy nhất có thể cam kết trạng thái cụm. Nó có trách nhiệm duy trì cụm với thông tin cập nhật. Các nút master cam kết dữ liệu cụm theo từng giai đoạn (tương tự như giao dịch cam kết hai giai đoạn trong kiến ​​trúc phân tán):

**1**Máy chủ sẽ tính toán các thay đổi của cụm, công bố chúng tới từng nút riêng lẻ, sau đó chờ xác nhận.

**2**Mỗi nút nhận được các bản cập nhật cụm, nhưng các bản cập nhật chưa được áp dụng cho trạng thái cục bộ của các nút. Khi nhận được, chúng sẽ gửi xác nhận đến máy chủ.

**3**Khi máy chủ nhận được đủ số lượng xác nhận từ các nút đủ điều kiện là máy chủ, nó sẽ cam kết thực hiện các thay đổi để cập nhật trạng thái cụm (máy chủ không cần phải chờ xác nhận từ mọi nút, chỉ cần chờ các nút đủ điều kiện là máy chủ).

**4**Sau khi cam kết thành công các thay đổi cụm, nút chính sẽ phát một thông báo cuối cùng tới từng nút riêng lẻ để hướng dẫn chúng cam kết các thay đổi cụm đã nhận trước đó.

**5**Các nút riêng lẻ xác nhận các bản cập nhật cụm.

Thuộc tính cluster.publish.timeout đặt giới hạn thời gian (mặc định là 30 giây) để cam kết mỗi đợt cập nhật cụm thành công. Khoảng thời gian này chạy từ thời điểm đăng bài đăng tin nhắn cập nhật cụm đầu tiên lên các nút cho đến khi trạng thái cụm được cam kết. Nếu các bản cập nhật cụm toàn cục được cam kết thành công trong vòng 30 giây mặc định, máy chủ sẽ đợi cho đến khi thời gian này trôi qua trước khi bắt đầu đợt cập nhật cụm tiếp theo. Tuy nhiên, câu chuyện không kết thúc ở đây.

Nếu các bản cập nhật cụm không được cam kết trong vòng 30 giây, máy chủ có thể đã chết. Việc bầu chọn máy chủ mới sẽ bắt đầu như một kết quả.

Mặc dù các bản cập nhật cụm toàn cầu đã được cam kết, máy chủ vẫn chờ các nút chưa trả về xác nhận. Trừ khi nó nhận được xác nhận,

***14.6 Cụmbậc thầy* 493**

master không thể đánh dấu cập nhật cụm này là thành công. Trong những trường hợp như vậy, master theo dõi các nút này và chờ thời gian gia hạn được thiết lập bởi thuộc tính cluster.follower\_lag.timeout, mặc định là 90 giây. Nếu các nút không phản hồi trong thời gian gia hạn 90 giây này, chúng được đánh dấu là các nút không thành công và master sẽ xóa chúng khỏi cụm.

Như bạn có thể đã biết, có rất nhiều thứ diễn ra bên trong Elasticsearch. Cập nhật cụm thường xuyên xảy ra và master chịu trách nhiệm duy trì các phần chuyển động. Trong kịch bản cập nhật cụm trước đó, master chờ xác nhận từ một nhóm các nút đủ điều kiện là master được gọi là quorum trước khi cam kết trạng thái, thay vì chờ các nút còn lại. quorum là số lượng tối thiểu các nút master cần thiết để master hoạt động hiệu quả, như đã thảo luận trong phần tiếp theo.

* + 1. Từ ai

Master kiểm soát việc duy trì và quản lý cụm. Tuy nhiên, nó tham khảo ý kiến ​​của một nhóm các nút đủ điều kiện làm master để cập nhật trạng thái cụm và bầu chọn master. Nhóm là một tập hợp con các nút đủ điều kiện làm master được lựa chọn cẩn thận, cần thiết để master vận hành cụm hiệu quả. Đây là phần lớn các nút được master tham khảo ý kiến ​​để đạt được sự đồng thuận về các vấn đề liên quan đến trạng thái cụm và các vấn đề khác.

Mặc dù chúng ta đang tìm hiểu về số lượng đủ điều kiện, tin tốt là chúng ta (người dùng/quản trị viên) không phải lo lắng về cách tạo số lượng đủ điều kiện. Các cụm tự động xây dựng số lượng đủ điều kiện dựa trên các nút đủ điều kiện chủ có sẵn. Có một công thức đơn giản để tìm số lượng tối thiểu các nút đủ điều kiện chủ (số lượng đủ điều kiện) cần thiết, cho một tập hợp các nút đủ điều kiện chủ:

Số lượng tối thiểu các nút đủ điều kiện làm chủ = (số lượng các nút đủ điều kiện làm chủ / 2) + 1

Giả sử chúng ta có một cụm 20 nút và 8 nút được chỉ định là các nút đủ điều kiện làm chủ (vai trò nút được đặt thành chủ). Bằng cách áp dụng công thức này, cụm của chúng ta tạo ra một số lượng đủ điều kiện với năm nút đủ điều kiện làm chủ (được lựa chọn cẩn thận) (8 / 2 + 1 = 5). Ý tưởng là chúng ta cần ít nhất năm nút đủ điều kiện làm chủ để tạo thành một số lượng đủ điều kiện.

Nguyên tắc chung là số lượng tối thiểu được khuyến nghị của các thành viên đủ điều kiện làm chủ trong bất kỳ cụm nút nào là ba. Đặt ba nút đủ điều kiện làm chủ là một cách chắc chắn để quản lý cụm. Một lợi thế lớn khác của việc có ít nhất ba nút trong một cụm quorum là điều này làm giảm vấn đề não chia đôi, mà chúng ta sẽ xem xét trong phần sau.

* + 1. Vấn đề não chia đôi

Sức khỏe cụm của Elasticsearch phụ thuộc rất nhiều vào nhiều yếu tố: mạng, bộ nhớ, thu gom rác JVM, v.v. Trong một số trường hợp, cụm được chia thành hai cụm với một số nút trong một cụm và một số nút trong cụm khác. Ví dụ, hình 14.16 cho thấy một cụm có hai nút đủ điều kiện làm chủ, nhưng một (Nút A) được bầu làm nút chủ. Miễn là chúng ta ở trạng thái vui vẻ, cụm sẽ khỏe mạnh và chủ thực hiện trách nhiệm của mình một cách siêng năng.

**494 CPHẦN14*Sự quản lý***

**Một nút chính**

**Cụm**

**Một nút đủ điều kiện làm chủ**

**Hình 14.16**

**Một nút haicụmvới một bậc thầy**

**Một cụm hai nút hoạt động tốt**



**Nút A**

**Nút B**

Bây giờ, hãy thử làm một việc khác. Giả sử Node B chết do sự cố phần cứng. Vì Node A là master, nên nó tiếp tục hoạt động để phục vụ các truy vấn với một node: về cơ bản, chúng ta có một cụm một node trong khi chờ Node B khác khởi động để tham gia cụm.

Đây là nơi mọi thứ có thể trở nên khó khăn. Trong khi Node B đang khởi động, giả sử kết nối mạng bị ngắt, khiến Node B không thể thấy sự tồn tại của Node A. Điều nàydẫn đến việc Node B đảm nhận vai trò chủ vì nó nghĩ rằng không có chủ trong cụm, mặc dù Node A tồn tại như chủ. Điều này dẫn đến tình huống não chia đôi (xem hình 14.17).

**Cụm não chia đôi**

**Nút chính**



**Nút A**

**Hai bậc thầy**

**Nút B**

**Nút chính mới**

**Hình 14.17**

**Cụm não chia đôi: một cụm có hai chủ nhân**

**phục vụ khách hàng**

Hai nút không giao tiếp được với nhau do vấn đề mạng, vì vậy chúng hoạt động tốt như một phần của cụm. Nhưng vì cả hai nút đều là nút chủ, nên bất kỳ yêu cầu nào đến một trong hai nút đều chỉ được thực hiện bởi nút nhận. Tuy nhiên, dữ liệu trong một nút không hiển thị với nút kia, điều này làm tăng sự khác biệt về dữ liệu. Đây là một trong những lý do để có ít nhất ba nút đủ điều kiện làm nút chủ trong một cụm. Có ba nút như vậy sẽ tránh được việc hình thành cụm não chia đôi.

* + 1. Thầy tận tụycác nút

Vì một nút có thể được gán nhiều vai trò, nên không có gì ngạc nhiên khi thấy một cụm có 20 nút trong đó tất cả các nút đều thực hiện tất cả các vai trò. Không có hại gì khi tạo kiểu kiến ​​trúc cụm này; tuy nhiên, thiết lập như vậy chỉ hoạt động đối với cụm nhẹ

***Bản tóm tắt* 495**

nhu cầu. Như chúng ta đã thấy, nút chính là nút quan trọng trong cụm và giữ cho cụm hoạt động.

Nếu dữ liệu được lập chỉ mục hoặc tìm kiếm với tốc độ tăng trưởng theo cấp số nhân, mọi nút, bao gồm cả các nút chính, đều bị ảnh hưởng về hiệu suất. Một nút chính hoạt động chậm đang yêu cầusự cố: các hoạt động cụm chạy chậm hơn hoặc thậm chí có thể bị đình trệ. Vì lý do này, luôn nên tạo một máy chuyên dụng để lưu trữ nút chính. Có một nút chính chuyên dụng cho phép các cụm chạy trơn tru và giảm thiểu mất dữ liệu và thời gian ngừng hoạt động của ứng dụng.

Như đã đề cập, nguyên tắc chung là phải có ít nhất ba người đủ điều kiện làm chủ chuyên tráchcác nút trong một cụm. Khi tạo cụm, hãy đảm bảo đặt node.roles thành master, như được hiển thị trong đoạn mã sau, để chỉ định master dành riêng cho nút:

node.roles: [ chủ ]

Theo cách này, vai trò chính chuyên dụng sẽ không bị quá tải khi thực hiện các hoạt động liên quan đến dữ liệu hoặc thu thập mà chỉ quản lý cụm toàn thời gian.

Vậy là xong! Chúng ta đã xem xét phần quản trị của Elasticsearch trong chương này.

Ở chương tiếp theo, chúng ta sẽ xem xét cách điều chỉnh hiệu suất.

#### Bản tóm tắt

* Elasticsearch mở rộng theo chiều ngang khi thêm các nút mới vào cụm. Các nút mới tham gia cụm miễn là chúng được liên kết với cùng tên cụm và cài đặt mạng.
* Một cách để cải thiện thông lượng đọc và do đó, hiệu suất là thêm bản sao vào cụm. Các bản sao này sẽ tiếp nhận các lượt đọc và phục vụ dữ liệu nhanh chóng.
* Các nút giao tiếp với nhau trên một cổng vận chuyển, được đặt thành 9300 theo mặc định. Điều này có thể được sửa đổi bằng cách điều chỉnhvận chuyển.cảngthuộc tính trong tệp elastic-search.yml.
* Máy khách HTTP giao tiếp với Elasticsearch trên http.port (mặc định là 9200) bằng cách sử dụng điểm cuối RESTFul.
* Một nút có thể bao gồm nhiều chỉ mục và mỗi chỉ mục có thể bao gồm nhiều phân đoạn. Kích thước lý tưởng của một phân đoạn không nên quá 50 GB bộ nhớ.
* Các mảnh vỡ và bản sao chiếm nhiều không gian, vì vậy chiến lược tổ chức của chúng ta nên xác định kích thước phù hợp dựa trên các yêu cầu hiện tại và mục đích sử dụng trong tương lai.
* Mặc dù việc thêm bản sao cải thiện hiệu suất đọc/truy vấn của máy khách, nhưng nó đi kèm với một chi phí. Chúng ta phải đảm bảo thực hiện các thử nghiệm phù hợp và theo dõi các đột biến trước khi chỉ định kích thước chuẩn cho mỗi phân đoạn.
* Elasticsearch cho phép sao lưu và khôi phục dữ liệu bằng tính năng snapshot và restore. Snapshot cho phép chúng ta sao lưu cụm vào một kho lưu trữ.
* Kho lưu trữ có thể là hệ thống tệp cục bộ hoặc kho lưu trữ đối tượng trên nền tảng đám mây như AWS S3.
* Ảnh chụp nhanh có thể bao gồm các chỉ mục, luồng dữ liệu (cũng như trạng thái cụm, chẳng hạn như liên tục hay tạm thời), mẫu lập chỉ mục, chính sách quản lý vòng đời chỉ mục (ILM) và nhiều hơn nữa.

**496 CPHẦN14*Sự quản lý***

* Điểm cuối quản lý vòng đời ảnh chụp nhanh (SLM) được khai báo là\_slmtạo chính sách xác định ảnh chụp nhanh cùng với lịch trình và các thuộc tính khác.
* Chúng ta có thể bắt đầu khôi phục dữ liệu từ ảnh chụp nhanh theo cách thủ công bằng cách gọi

\_khôi phụcGiao diện lập trình ứng dụng (API).

* Chúng ta có thể sử dụng giao diện phong phú của Kibana để xác định ảnh chụp nhanh và chính sách và khôi phục chúng. Các chính sách có sẵn trong menu điều hướng Stack Management.
* Elasticsearch cung cấp nhiều thiết lập khác nhau thông qua các tệp elasticsearch.yml, jvm.options và log4j2.properties.
* Chúng ta có thể điều chỉnh nhiều thuộc tính, chẳng hạn như thay đổi tên cụm, di chuyển nhật ký và đường dẫn dữ liệu, thêm cài đặt mạng, v.v., bằng cách chỉnh sửa tệp elastic-search.yml.
* Tệp config/jvm.options định nghĩa dữ liệu liên quan đến JVM cho nút, nhưng chúng ta không bao giờ được chỉnh sửa tệp này.
* Để tùy chỉnh các thiết lập cho tùy chọn JVM, chúng ta cần tạo một tệp mới có đuôi là \*.options (như custom\_settings.options) và thả vào thư mục có tên là jvm.options.d, thư mục này sẽ có trong thư mục cấu hình cho các bản cài đặt đã lưu trữ (tar hoặc zip).
* Chúng ta có thể thiết lập bộ nhớ heap cần thiết bằng cách sử dụng-XmsVà-Xmxcờ, nơi

-Xms đặt kích thước heap tối đa và -Xmx đặt kích thước heap tối thiểu. Theo nguyên tắc chung, không bao giờ đặt kích thước heap vượt quá 50% RAM khả dụng.

* Nút chính trong cụm là nút quan trọng có chức năng quản lý và duy trì các hoạt động liên quan đến cụm và cập nhật trạng thái cụm phân tán.
* Nút chính sẽ tham khảo ý kiến ​​của một số nút nhất định để xác nhận trạng thái cụm hoặc bầu một nút chính mới.
* Số lượng tối thiểu các nút đủ điều kiện làm chủ được cụm lựa chọn cẩn thận để giảm thiểu lỗi nút khi đưa ra quyết định.
* Chúng ta nên cung cấp tối thiểu ba nút đủ điều kiện làm chủ khi hình thành một cụm, để tránh cụm não bị chia tách.

*Hiệu suất vàxử lý sự cố*

***Chương này bao gồm***

* Hiểu lý do khiến tìm kiếm và truy vấn lập chỉ mục chậm
* Điều chỉnh và cải thiện hiệu suất của chậmtruy vấn và lập chỉ mục
* Xử lý sự cố cụm không ổn định

Khi cụm Elasticsearch đã sẵn sàng để sản xuất, vô số thứ có thể xảy ra sai sót, từ người dùng phàn nàn về tìm kiếm chậm đến các nút không ổn định, sự cố mạng, sự cố phân mảnh quá mức, sự cố bộ nhớ, v.v. Duy trì tình trạng của cụm ở trạng thái XANH (khỏe mạnh) là tối quan trọng. Liên tục theo dõi tình trạng và hiệu suất của cụm là một trong những công việc chính của quản trị viên.

Để khắc phục sự cố cụm không ổn định, cần hiểu rõ về hoạt động bên trong của Elasticsearch, các khái niệm về mạng, giao tiếp giữa các nút, cài đặt bộ nhớ và nhiều API cho các nút, cụm, phân bổ cụm và các mục đích sử dụng khác. Tương tự như vậy, việc điều chỉnh cấu hình để hiểu các mô hình tài liệu, thời gian làm mới phù hợp, v.v. giúp điều chỉnh cụm để có hiệu suất cao hơn.

**497**

**498 CPHẦN15 *Hiệu suất và xử lý sự cố***

Trong chương này, chúng ta sẽ xem xét các vấn đề phổ biến như tốc độ truy vấn và tốc độ thu thập dữ liệu chậm để hiểu lý do đằng sau chúng. Vì Elasticsearch là một kiến ​​trúc phân tán phức tạp nên có một số nơi cần tìm cách khắc phục. Chúng ta sẽ thảo luận về các giải pháp rõ ràng nhất và thường được áp dụng nhất trong chương này.

Chúng tôi cũng xem xét các vấn đề phát sinh từ các cụm không ổn định và khắc phục sự cố với sự trợ giúp của sức khỏe cụm, phân bổ cụm và API nút. Sau đó, chúng tôi thảo luận về cách Elasticsearch thiết lập ngưỡng sử dụng bộ nhớ và đĩa để giữ cho cụm hoạt động và hoạt động. Hãy bắt đầu bằng cách xem lý do tại sao các truy vấn tìm kiếm của chúng tôi không mang lại kết quả nhanh hơn và các tùy chọn khắc phục sự cố có sẵn.

**GHI CHÚ**Như đã đề cập trong chương 14, tương tự như quản trị Elasticsearch, điều chỉnh hiệu suất là một chủ đề nâng cao đòi hỏi phải có chuyên gia. Trước khi điều chỉnh ứng dụng của bạn để có hiệu suất, hãy tham khảo ý kiến ​​chuyên gia, đọc tài liệu và thử nghiệm trong môi trường phòng thí nghiệm của bạn. Chương này trình bày góc nhìn tổng quan về bối cảnh hiệu suất.

#### Vấn đề tìm kiếm và tốc độ

Mặc dù Elasticsearch là công cụ tìm kiếm gần như thời gian thực, chúng ta phải cẩn thận thuần hóa nó để đảm bảo nó hoạt động như mong đợi trong nhiều tình huống khác nhau. Theo thời gian, hiệu suất của Elasticsearch có thể giảm nếu không được thiết kế tốt với các yêu cầu dữ liệu trong tương lai và nếu không được bảo trì liên tục. Sự suy giảm hiệu suất của máy chủ gây bất lợi cho sức khỏe cụm tổng thể, ảnh hưởng đến hiệu suất truy vấn tìm kiếm và lập chỉ mục.

Các vấn đề chính mà người dùng Elasticsearch thường báo cáo nhất là phục vụ chậm các truy vấn tìm kiếm và tốc độ lập chỉ mục chậm. Một điểm bán hàng độc đáo của Elasticsearch là các truy vấn cực nhanh. Tuy nhiên, chúng ta không thể mong đợi giải pháp có sẵn này đáp ứng được mọi nhu cầu của mình. Nhiều biến số như phân phối hợp lý các nút, phân đoạn và bản sao tùy thuộc vào quản lý bộ nhớ và tính khả dụng của phần cứng cho phép chúng ta tạo một cụm lành mạnh khi chúng ta chuẩn bị và sản xuất cơ sở hạ tầng và ứng dụng.

Thông thường, Elasticsearch được thiết lập không đúng cho tình huống tìm kiếm hiện tại. Ví dụ, một số ứng dụng tập trung vào tìm kiếm, trong khi những ứng dụng khác cần phục vụ các tìm kiếm ít thường xuyên hơn.

* + 1. Phần cứng hiện đại

Elasticsearch sử dụng Lucene để lập chỉ mục và lưu trữ dữ liệu. Dữ liệu được lưu trữ trong hệ thống tệp và mặc dù Lucene thực hiện việc này một cách hiệu quả, nhưng các trợ giúp bổ sung như cung cấp ổ đĩa thể rắn (SSD) thay vì ổ đĩa cứng (HDD), phân bổ bộ nhớ dồi dào, hỗ trợ hợp nhất không thường xuyên, v.v. sẽ có hiệu quả lâu dài.

Mỗi nút bao gồm các mảnh vỡ và bản sao cũng như dữ liệu cụm khác, bao gồm các cấu trúc dữ liệu nội bộ mà Elasticsearch duy trì. Vì Elasticsearch được phát triển bằng Java, việc phân bổ bộ nhớ heap có dung lượng lớn nói chung giúp ứng dụng chạy trơn tru. Heap là vị trí bộ nhớ nơi các đối tượng mới được lưu trữ trong không gian thế hệ trẻ. Khi không gian này đầy lên, bất kỳ đối tượng nào sống sót sau rác nhỏ

***15.1 Tìm kiếm và tốc độvấn đề* 499**

các lần chạy thu thập (GC) được chuyển đến không gian thế hệ cũ. Cung cấp một đống lớn hơn cho Elasticsearch giúp tránh lấp đầy không gian thế hệ trẻ một cách nhanh chóng, do đó tránh được các lần chạy thu gom rác.

Nguyên tắc chung là cung cấp ít nhất một nửa bộ nhớ dưới dạng bộ nhớ heap cho ứng dụng. Ví dụ, nếu máy của bạn được cấu hình với RAM 16 GB, hãy đảm bảo heap được đặt thành ít nhất 8 GB. Chúng ta có thể đặt điều này bằng cách cấu hình tệp .optionsvới thiết lập -Xmx (xem phần 14.5.3).

Ngoài ra, việc gắn một đĩa lưu trữ cục bộ cho phép hiệu suất tốt hơn khi ghi vào đĩa thay vì sử dụng hệ thống tệp và đĩa dựa trên mạng. Cuối cùng, việc cung cấp các nút riêng lẻ với các ổ đĩa cục bộ là một chiến lược tốt hơn.

Hãy nhớ rằng chúng ta phải hiểu các yêu cầu về lập chỉ mục và phân mảnh để phân bổ bộ nhớ ban đầu. Ví dụ, dựa trên số lượng phân mảnh và bản sao mà chúng ta dự định giữ trên nút, chúng ta có thể cần tính toán và cung cấp bộ nhớ đĩa vật lý.

Số lượng phân đoạn tối ưu giúp tránh việc xây dựng một cụm có quá nhiều phân đoạn nhỏ. Khi số lượng phân đoạn tăng lên, giao tiếp giữa các nút cũng tăng theo. Do đó, hiệu suất cũng phụ thuộc vào dung lượng mạng và băng thông. Việc thêm bản sao làm tăng thông lượng đọc truy vấn tìm kiếm, nhưng việc có quá nhiều bản sao sẽ nhanh chóng tiêu tốn bộ nhớ và số lượng bản sao lớn hơn sẽ khó quản lý.

* + 1. Mô hình hóa tài liệu

Elasticsearch là cơ sở dữ liệu NoSQL; dữ liệu phải được phi chuẩn hóa, không giống như trong cơ sở dữ liệu quan hệ, nơi dữ liệu được chuẩn hóa. Ví dụ, nếu một nhân viên được tạo, bản ghi bao gồm thông tin đầy đủ về nhân viên đó.

Mọi tài liệu trong Elasticsearch đều độc lập, do đó không cần phải có liên kết trên dữ liệu. Nếu dữ liệu của bạn chủ yếu là cha-con, bạn có thể muốn xem xét lại việc sử dụng Elastic-search. Các hoạt động lồng nhau và cha-con chậm và làm giảm hiệu suất ngay từ đầu. Nếu bạn đến từ thế giới cơ sở dữ liệu quan hệ, hãy đảm bảo rằng bạn hiểu các nguyên tắc mô hình hóa dữ liệu NoSQL.

Ngoài ra, chúng ta nên hướng đến mục tiêu hạn chế tìm kiếm nhiều trường vì tìm kiếm truy vấn trên nhiều trường làm chậm thời gian phản hồi truy vấn. Thay vào đó, hãy kết hợp nhiều trường thành một trường duy nhất và tìm kiếm trên trường duy nhất này. May mắn thay, Elasticsearch cung cấp thuộc tính copy\_to trên các trường, giúp ích trong trường hợp này.

Hãy xem một ví dụ. Giả sử chúng ta có một vài trường (tiêu đề và tóm tắt) trong tài liệu sách của mình và chúng được lập chỉ mục trong chỉ mục programming\_books1. Khi chúng ta viết truy vấn để tìm kiếm trong tiêu đề và tóm tắt, chúng ta thường tìm kiếm theo hai trường riêng lẻ này.

**Liệt kê 15.1 Tìm kiếm không hiệu quả trên nhiều trường**

ĐẶT sách lập trình\_1/\_doc/1

{

"title":"Elasticsearch trong hành động",

**Lập chỉ mục một tài liệu mẫu trước khi thực hiện tìm kiếm**

"synopsis": "Một cuốn sách hành động trực tiếp, thực hành, có ví dụ minh họa"

}

**500 CPHẦN15 *Hiệu suất và xử lý sự cố***

NHẬN programming\_books1/\_search

**Tìm kiếm tài liệu**

{

"truy vấn": { "multi\_match": {

**Gọi truy vấn multi\_match để tìm kiếm trên**

**sử dụng truy vấn multi\_match**

"query": "Ví dụ thực hành Elasticsearch được điều khiển",

"các trường": ["tiêu đề","tóm tắt"]

}

}

}

**Các trường để tìm kiếm**

Truy vấn này tìm kiếm tiêu chí trên nhiều trường. Đây là truy vấn tốn kém, dựa trên số lượng trường và tài liệu (hãy tưởng tượng tìm kiếm hơn một chục trường trong nhiều cuốn sách). Chúng ta có thể đưa một mẫu trường kết hợp vào hoạt động bằng cách sử dụng thuộc tính copy\_to trên từng trường để giảm bớt các vấn đề như vậy.

Danh sách sau đây tạo ra một bản đồ chosách\_lập\_trình2chỉ mục. Điều đáng chú ý là việc bổ sung một trường riêng biệt được gọi làtiêu đề\_tóm tắt, đó là mộtchữkiểu dữ liệu.

**Liệt kê 15.2 Định nghĩa lược đồ nâng cao vớisao chép đếnthuộc tính**

PUT lập trình\_sách2

{

"ánh xạ": { "thuộc tính": {

"tiêu đề":{

"loại": "văn bản",

"copy\_to": "title\_synopsis"

},

"tóm tắt":{ "loại": "văn bản",

"copy\_to": "title\_synopsis"

},

**Sao chép thông tin tiêu đề vào title\_synopsis**

**Sao chép thông tin tóm tắt vào title\_synopsis**

"tóm tắt tiêu đề":{"loại": "văn bản"

**Biểu thị title\_synopsis dưới dạng trường văn bản**

}

}

}

}

Như bạn có thể thấy từ định nghĩa lược đồ trong danh sách này, mỗi trường trong hai trường đều có một thuộc tính copy\_to bổ sung với giá trị trỏ đến trường thứ ba, title\_synopsis. Bất kỳ dữ liệu nào được lập chỉ mục vào trường title và synopsis đều được sao chép vào title\_synopsis ẩn. Danh sách tiếp theo hiển thị việc lập chỉ mục một tài liệu sách mẫu vào chỉ mục programming\_books2.

**Liệt kê 15.3 Lập chỉ mục một mẫutài liệu**

ĐẶT sách lập trình\_2/\_doc/1

{

"title":"Elasticsearch trong hành động",

"synopsis": "Một cuốn sách hành động trực tiếp, thực hành, có ví dụ minh họa"

}

* 1. ***Tìm kiếm và tốc độvấn đề* 501**

Chúng tôi lập chỉ mục cho cuốn sách bằng hai trường mà không đề cập đến trường title\_synopsis trong tài liệu. Nó được lập chỉ mục như thế nào?

Trong quá trình lập chỉ mục, Elasticsearch sẽ điền vào trường thứ ba này (title\_synopsis) bằng cách kết hợp các trường title và synopsis (bạn còn nhớ thuộc tính copy\_to trên các trường này không?). Do đó, một trường kết hợp có tên là title\_synopsis sẽ chứa toàn bộ dữ liệu từ các trường khác.

Vì chúng tôi đã cải thiện lược đồ bằng chức năng copy\_to này, nên chúng tôi có thể viết lại truy vấn tìm kiếm để trở thành truy vấn khớp đơn giản thay vì khớp nhiều lần.

**Liệt kê 15.4 Acuộc thi đấutruy vấn chống lạitiêu đề\_tóm tắtcánh đồng**

NHẬN programming\_books2/\_search

{

"truy vấn": {

"phù hợp": { "tóm tắt tiêu đề": {

**Chỉ định trường title\_synopsis cho truy vấn khớp**

"query": "Ví dụ thực hành Elasticsearch được điều khiển", "operator": "OR"

}

}

}

}

Bây giờ chúng ta sử dụng truy vấn khớp đơn giản (trái ngược với truy vấn multi\_match tốn kém được mô tả trong danh sách 15.1) để lấy kết quả của chúng ta. Vì công việc khó khăn là sao chép dữ liệu vào một trường kết hợp đã được thực hiện tại thời điểm lập chỉ mục, nên tìm kiếm chỉ cần truy vấn đối với trường đã có sẵn này.

Tìm kiếm qua nhiều trường bằng multi\_match (hoặc thậm chí là truy vấn query\_string, như đã thảo luận trong chương 9) là một hoạt động tốn kém. Thay vào đó, chúng tôi đã sử dụng com-phương pháp bined fields với cấu trúc copy\_to. Việc khớp một vài trường thay vì tìm kiếm trên hàng chục trường là một ứng cử viên tốt cho việc cải thiện hiệu suất.

* + 1. Chọn loại từ khóa thay vì loại văn bản

Các tìm kiếm toàn văn trải qua giai đoạn phân tích văn bản: chúng được chuẩn hóa và mã hóa giống như lập chỉ mục, trước khi kết quả được lấy. Đây là một hoạt động đòi hỏi nhiều tính toánvà chúng ta có thể tránh nó trên các trường là loại từ khóa. Các trường từ khóa không trải qua phân tích văn bản, do đó giúp chúng ta tiết kiệm thời gian và công sức khi tìm kiếm.

Hãy cân nhắc khai báo một kiểu từ khóa cho bất kỳ trường văn bản nào nếu trường hợp sử dụng cho phép tìm kiếm theo từ khóa. Ví dụ, chúng ta có thể tạo một cuốn sách có tiêu đề được xác định là kiểu dữ liệu văn bản cũng như trường từ khóa bằng cách sử dụng tính năng nhiều trường.

**Liệt kê 15.5 Truy vấn đa trường sử dụngtừ khóakiểu**

PUT lập trình\_sách3

{

"ánh xạ": { "thuộc tính": {

**502 CPHẦN15 *Hiệu suất và xử lý sự cố***

"tiêu đề":{

"kiểu": "văn bản", "trường": {

"thô":{

"loại":"từ khóa"

}

}

}

}

}

}

**Xác định trường title.raw là một loại từ khóa**

Ngoài việc khai báo trường tiêu đề là kiểu văn bản, chúng tôi cũng khai báo nó là kiểu từ khóa dưới tên title.raw. Theo cách này, khi chúng tôi lập chỉ mục tài liệu, trường tiêu đề được lưu trữ dưới dạng kiểu văn bản (do đó đang trải qua quá trình phân tích văn bản) và kiểu từ khóa (không có phân tích văn bản). Tìm kiếm có thể được thực hiện trên trường title.raw (kiểu từ khóa) để tránh phân tích. Danh sách sau đây cho thấy điều này; chúng tôi cũng lập chỉ mục một tài liệu mẫu trước khi chạy truy vấn.

**Liệt kê 15.6 Acuộc thi đấutruy vấn để giảm thiểu chi phí phân tích**

ĐẶT sách lập trình\_3/\_doc/1

{

"title":"Elasticsearch trong hành động"

**Lập chỉ mục một cuốn sách mẫu**

}

NHẬN programming\_books3/\_search

{

"truy vấn": {

"cuộc thi đấu": {

**Tìm kiếm trên mộttrường từ khóa**

"title.raw": "Elasticsearch trong hành động"

}

}

}

Bởi vì chúng tôi chạy tìm kiếm theo trường từ khóa, chúng tôi phải đảm bảotitle.raw value được viết chính xác như những gì đã được lập chỉ mục. Chúng tôi không thể lấy kết quả nếu ngay cả một chữ cái cũng khác với bản gốc (ví dụ, nếu chữ cái đầu tiên là chữ thường). Hãy thử lại truy vấn tương tự, nhưng lần này với chữ thường "elasticsearch". Bạn sẽ không nhận được bất kỳ kết quả nào.

Có một số khuyến nghị chung khác như sử dụng bộ lọc tìm kiếm, lập chỉ mục trước dữ liệu, tránh truy vấn ký tự đại diện, v.v. Tham khảo tài liệu chính thức để biết chi tiết:<http://mng.bz/MBwm>.

Trong khi tốc độ tìm kiếm chậm hơn ảnh hưởng đến hiệu suất liên quan đến tìm kiếm, tốc độ thu thập chậm hơn ảnh hưởng đến phần ghi của ứng dụng. Chúng tôi thảo luận về các vấn đề về tốc độ lập chỉ mục này trong phần sau.

* 1. ***Tốc độ chỉ sốvấn đề* 503**

#### Vấn đề về tốc độ chỉ mục

Mặc dù chúng ta chủ yếu gặp phải các vấn đề tìm kiếm khi người dùng tìm kiếm dữ liệu, nhưng mặt trái của vấn đề là các vấn đề trong khi lập chỉ mục dữ liệu. Trong phần này, chúng ta sẽ xem xét một số lý do khiến hoạt động lập chỉ mục có thể hoạt động kém hiệu quả và các mẹo để cải thiện hiệu suất lập chỉ mục.

* + 1. Mã định danh do hệ thống tạo ra

Khi chúng ta sử dụng ID do người dùng cung cấp, Elasticsearch phải thực hiện thêm một bước: kiểm tra xem ID được cung cấp đã tồn tại trong chỉ mục hay chưa. Nếu câu trả lời là không, việc lập chỉ mục tài liệu với ID đó sẽ được tiến hành. Đây là một cuộc gọi mạng không cần thiết gây tốn kém khi chúng ta có hàng nghìn hoặc hàng triệu tài liệu để lập chỉ mục.

Tuy nhiên, đôi khi chúng ta không cần phải lo lắng về ID tài liệu; chúng ta có thể để Elasticsearch tạo ID được tạo ngẫu nhiên cho các tài liệu. Theo cách này, một nút nhận được yêu cầu lập chỉ mục tài liệu sẽ tạo ID duy nhất toàn cục và gán ID đó cho tài liệu ngay lập tức.

Nếu tổ chức của bạn có thể sử dụng ID ngẫu nhiên, việc sử dụng chúng thay vì ID do người dùng xác định sẽ mang lại lợi ích về hiệu suất. Tuy nhiên, có một nhược điểm: lập chỉ mục tài liệu mà không có khóa chính sẽ gây ra sự trùng lặp.

* + 1. Yêu cầu hàng loạt

Việc lập chỉ mục tài liệu bằng API tài liệu đơn (lập chỉ mục từng cái một) sẽ gây ra rắc rối, đặc biệt là nếu chúng ta cần lập chỉ mục nhiều tài liệu dữ liệu. May mắn thay, Elastic-search cung cấp API \_bulk giúp lập chỉ mục tài liệu theo lô. (Chúng tôi đã thảo luận về

\_bulk API trong chương 5.) Vì không có kích thước lô cụ thể nào được ghi rõ ràng nên lời khuyên của tôi là hãy thử hiệu suất trên cụm để tìm kích thước tối ưu.

Ví dụ, nếu bạn cần nhập lượng lớn dữ liệu qua đêm, hãy thử tăng cài đặt làm mới để có lợi ích tiềm năng theo góc độ hiệu suất sử dụng đĩa. Trong quá trình lập chỉ mục, đặc biệt là chèn hàng loạt, tăng thời gian làm mới là một điều chỉnh khác mà hầu hết các quản trị viên sử dụng. Chúng ta sẽ tìm hiểu về việc tăng thời gian làm mới tiếp theo.

* + 1. Điều chỉnh tốc độ làm mới

Khi một tài liệu được lập chỉ mục, thông thường nó sẽ có sẵn để tìm kiếm trong vòng chưa đầy một giây, nhưng có nhiều hoạt động khác nhau diễn ra bên trong. Ban đầu, các tài liệu trong bộ đệm trong bộ nhớ được chuyển đến các phân đoạn trước khi được lưu trữ trong bộ đệm hệ thống tệp. Sau đó, cuối cùng chúng được đưa vào đĩa.

Tất cả các hoạt động được thực hiện trên một chỉ mục được cam kết sau khi lệnh làm mới được gọi. Nếu chúng ta có một lượng lớn tài liệu để lập chỉ mục, việc làm mới đảm bảo rằng chúng được ghi vào đĩa cũng như có sẵn để tìm kiếm. Chúng ta cần biết liệu các tài liệu được lập chỉ mục mới có sẵn để tìm kiếm ngay lập tức hay không.

**504 CPHẦN15 *Hiệu suất và xử lý sự cố***

Nếu chúng ta tạm dừng làm mới, chúng ta sẽ loại trừ bất kỳ tài liệu nào được lập chỉ mục mới nhất: chúng không khả dụng để tìm kiếm. Về cơ bản, chúng ta sẽ giảm thiểu hoạt động I/O tốn nhiều tài nguyên của đĩa. Ví dụ, nếu chúng ta tạm dừng hoạt động làm mới trong 1 phút, chúng ta có khả năng dừng 60 vòng đồng bộ đĩa. Nhược điểm là bất kỳ truy vấn tìm kiếm nào trong khoảng thời gian này đều không chọn các tài liệu mới được lập chỉ mục trong phút này.

Ví dụ, nếu trường hợp sử dụng của chúng tôi cho phép giữ lại quá trình làm mới trong một khoảng thời gian xác định, chúng tôi có thể đặt lại thời gian làm mới mặc định bằng cách gọi cài đặt trên chỉ mục. Danh sách sau đây hiển thị mã.

**Liệt kê 15.7 Tùy chỉnh cài đặt làm mới**

ĐẶT programming\_books3/\_settings

{

"chỉ số":{ "khoảng thời gian làm mới":"1 phút"

}

}

**Đặt giá trị tùy chỉnh (ở đây là 1 phút) cho chu kỳ làm mới**

Ở đây, chúng tôi đặt khoảng thời gian một phút để làm mới, do đó bất kỳ cuốn sách nào được thêm vào trong phút này đều không khả dụng để tìm kiếm. Một cách làm thông thường là tắt hoạt động làm mới trước khi lập chỉ mục. Danh sách tiếp theo cho thấy cách chúng ta có thể tắt hoàn toàn tính năng làm mới.

**Liệt kê 15.8 Vô hiệu hóalàm cho khỏe lại**

ĐẶT programming\_books3/\_settings

{

"chỉ số":{ "khoảng thời gian làm mới":-1

}

}

Bằng cách đặt refresh\_interval thành -1, về cơ bản chúng ta vô hiệu hóa thao tác làm mới. Bất kỳ tài liệu nào được lập chỉ mục sau khi chúng ta vô hiệu hóa làm mới đều không khả dụng để tìm kiếm. Nếu bạn tò mò về tác dụng phụ của thiết lập này, hãy lập chỉ mục một tài liệu và thực hiện tìm kiếm, như danh sách sau đây cho thấy. Tìm kiếm sẽ không mang lại kết quả nào.

**Liệt kê 15.9 Tìm kiếm với chỉ mục làm mới bị vô hiệu hóa**

PUT lập trình\_books3/\_doc/10

{

"title":"Elasticsearch dành cho nhà phát triển Java", "synopsis":"Elasticsearch kết hợp với Java"

**Lập chỉ mục một tài liệu mẫu**

}

NHẬN programming\_books3/\_search

{

"truy vấn": {

"cuộc thi đấu": {

**Tìm kiếm khi chức năng làm mới bị vô hiệu hóa sẽ không trả về kết quả nào.**

* 1. ***Tốc độ chỉ sốvấn đề* 505**

"title": "Elasticsearch Java"

}

}

}

Có một vài điều đang diễn ra ở đây. Chúng tôi lập chỉ mục một tài liệu vào cùng một chỉ mục (programming\_books3) và sau đó tìm kiếm nó. Bởi vì chúng tôi đã hủy hoạt động làm mớiTrong danh sách 15.8, truy vấn khớp không mang lại kết quả nào.

Sau khi lập chỉ mục hoàn tất, hãy đảm bảo bật tính năng làm mới. (Nếu hoạt động làm mới bị vô hiệu hóa, sẽ không có truy vấn tìm kiếm nào tìm nạp các tài liệu được lập chỉ mục trong khoảng thời gian làm mới đó.) Chúng ta có thể khôi phục khoảng thời gian làm mới bằng cách gọi mã này.

**Liệt kê 15.10 Buộc làm mới chỉ mục**

POST programming\_books3/\_refresh

Gọi điểm cuối \_refresh vào cuối hoạt động lập chỉ mục (hoặc định kỳ) đảm bảo rằng các tài liệu được lập chỉ mục có sẵn để tìm kiếm. Chúng tôi cũng có thể phân bổ các luồng bổ sung khi làm việc với các yêu cầu hàng loạt. Để làm như vậy, chúng tôi đặtThuộc tính thread\_pool với các luồng ghi bổ sung trong tệp elasticsearch.yml, như danh sách tiếp theo hiển thị.

**Liệt kê 15.11 Sửa đổi kích thước luồng của người viết**

thread\_pool: viết:

kích thước: 50

kích thước hàng đợi: 5000

Elasticsearch có nhiều nhóm luồng mà chúng ta có thể thay đổi hoặc sửa đổi theo yêu cầu của mình. Ví dụ, trong danh sách 15.11, kích thước ghi (chỉ mục) của nhóm được đặt thành

50. Điều này có nghĩa là chúng ta có 50 luồng để giúp lập chỉ mục dữ liệu trong nhiều luồng.Thuộc tính queue\_size giữ các yêu cầu trong hàng đợi cho đến khi có một luồng để xử lý yêu cầu tiếp theo.

Chúng ta có thể lấy nhóm luồng hiện tại bằng cách gọi lệnh này: GET \_nodes/thread\_pool. Lệnh này sẽ lấy các nhóm luồng khác nhau được tạo ra như một phần của cấu hình nút.Đoạn mã sau đây hiển thị kích thước nhóm luồng tìm kiếm và ghi khi thực hiện lệnh này:

# NHẬN \_nodes/thread\_pool "thread\_pool": {

....

"tìm kiếm": { "loại": "cố định", "kích thước": 7,

"queue\_size": 1000

},

**506 CPHẦN15 *Hiệu suất và xử lý sự cố***

"viết": {

"loại": "cố định", "kích thước": 4,

"queue\_size": 10000

},

..

}

Như bạn có thể thấy, Elasticsearch đã tự động gán bảy và sau đó là bốn luồng cho các hoạt động tìm kiếm và ghi, dựa trên số lượng bộ xử lý khả dụng của máy tính của tôi. Các thiết lập này là tĩnh, vì vậy chúng ta phải chỉnh sửa tệp cấu hình và khởi động lại máy chủ để chúng có hiệu lực.

Ngoài các khuyến nghị trước đó, còn có một số khuyến nghị khác, chẳng hạn như tăng kích thước bộ đệm lập chỉ mục, tắt bản sao trong quá trình lập chỉ mục, vô hiệu hóa hoán đổi, v.v. Bạn có thểtìm danh sách các khuyến nghị để cải thiện hiệu suất của các hoạt động lập chỉ mục tại<http://mng.bz/a19Y>.

#### Các cụm không ổn định

Một vấn đề thường gặp khi làm việc với Elasticsearch là tính ổn định của cụm. Giữ cụm hoạt động tốt sẽ giúp ích khi phục vụ các yêu cầu của khách hàng. Nhiều thứ có thể xảy ra sai sót với cụm! Trong phần này, chúng ta sẽ xem xét một số mối quan tâm thường gặp nhất.

* + 1. Cụm không phải là XANH

Tình trạng của cụm được biểu thị bằng trạng thái đèn giao thông hiệu quả, cấp cao. Việc phát hành GET \_cluster/health cung cấp trạng thái thời gian thực của cụm: ĐỎ, VÀNG hoặc XANH LÁ. Công việc chính của quản trị viên là làm cho trạng thái của cụm luôn là XANH LÁ.

Nếu trạng thái là ĐỎ, phải xử lý bằng mọi giá vì một phần của cụm không hoạt động hoặc ngừng hoạt động. Trạng thái không lành mạnh này của cụm (được biểu thị bằng trạng thái ĐỎ) có thể do nhiều vấn đề như lỗi phần cứng, mất mạng, hệ thống tệp bị hỏng, v.v., dẫn đến mất các nút cụm. Trong tình huống như vậy, các kỹ sư DevOps dừng mọi việc họ đang làm để khắc phục sự cố này. Trạng thái VÀNG cho biết cụm không lành mạnh nhưng có thể chạy với rủi ro có thể kiểm soát được. Điều này có thể cho biết mất một số nút, các phân đoạn và bản sao chưa được chỉ định hoặc các vấn đề khác. Mặc dù cụm có thể hoạt động, nhưng nó có thể sớm chuyển sang trạng thái ĐỎ nếu chúng ta không khắc phục sự cố. Trạng thái VÀNG có nghĩa là sự cố đang chờ xảy ra. Nhịp tim thường xuyên bắt nguồn từ nút chính đến các nút và ngược lại thiết lập tình trạng chung của cụm; nhịp tim từ nút chính đến các nút khác trong cụm cung cấp cơ chế để tìm các nút bị ngắt kết nối hoặc không phản hồi. Trong

Tương tự như vậy, tất cả các nút đều ping nút chính thường xuyên để kiểm tra xem nó có còn hoạt động hay không. Nút chính thực hiện các hành động thích hợp như phân phối lại dữ liệu cụm,

sao chép chéo cụm, quản lý chỉ mục, sắp xếp lại phân đoạn, v.v. khi các thành viên của cụm xuất hiện hoặc biến mất. Khi chúng được coi là không phù hợp với mục đích hiện tại, nút(các nút) sẽ bị xóa khỏi cụm bởi nút chính.

***15.3 Không ổn địnhcác cụm* 507**

Một master ổn định là một tính năng quan trọng của một cụm lành mạnh. Nếu bất kỳ ping nào từ các nút trả về phản hồi tiêu cực (có nghĩa là master có thể đã ngừng hoạt động!), các nút không chờ cuộc gọi được định thời khác mà thay vào đó gọi để bầu cử ngay lập tức. Master được bầu ngay lập tức từ danh sách các nút đủ điều kiện làm master.

* + 1. Chưa được chỉ địnhmảnh vỡ

Khi chúng ta tạo một chỉ mục với một số lượng mảnh, Elasticsearch sẽ đưa ra một chiến lược để phân bổ các mảnh cho các nút khả dụng. Trong một ví dụ giả định về 5 nút có chỉ mục có 10 mảnh, mỗi nút được phân bổ 2 mảnh để cân bằng việc chỉ định mảnh. Các bản sao được cân bằng tương tự, mặc dù chúng không bao giờ được phân bổ trên cùng một nút với các đối tác của chúng.

Trong một số trường hợp, không thể gán các mảnh. Ví dụ, nếu chúng ta có một chỉ mục mới tạo với 10 mảnh trên cụm 5 nút và một hoặc tất cả các mảnh chưa được ký (chưa phân bổ) cho bất kỳ nút nào trong số 5 nút này, cụm sẽ phàn nàn về các mảnh chưa được gán.

Tuy nhiên, chúng ta đang nói về một tình huống giả định trong đó một hoặc tất cả các mảnh không được phân bổ cho bất kỳ nút nào. Các mảnh không được phân bổ hoặc không được phân bổ là một sự cố sản xuất. Điều này thường xảy ra trong quá trình cân bằng lại các mảnh do một nút (hoặc một vài nút) bị lỗi vì bất kỳ lý do gì. Các mảnh không được phân bổ có hậu quả hoạt động đối với các chỉ mục mới được tạo (chỉ mục mới) hoặc đã hoạt động sau khi được tạo trước đó.

Nếu các mảnh vỡ không được chỉ định cho một chỉ mục hiện có (nhiều khả năng là trong giai đoạn cân bằng lại sau khi một hoặc một vài nút chết), các hoạt động đọc (tìm kiếm) sẽ bị dừng lại vì phần tài liệu của mảnh vỡ đó có thể đã biến mất. Các hoạt động ghi cũng dừng lại vì cùng một vấn đề về các mảnh vỡ không được chỉ định. Nếu điều này đúng với một chỉ mục mới được tạo với các mảnh vỡ không được chỉ định, các hoạt động ghi sẽ tạm dừng cho đến khi việc phân bổ mảnh vỡ được khắc phục.

Elasticsearch cung cấp API phân bổ thuận tiện để kiểm tra bản chất chưa được gán của các phân đoạn. Việc phát hành lệnh sau sẽ lấy lời giải thích chi tiết về lý do tại sao Elasticsearch không thể gán các phân đoạn cho các nút có liên quan.

**Liệt kê 15.12 Nhận lời giải thích cho unassignedmảnh vỡ**

NHẬN \_cluster/phân bổ/giải thích

Vì chúng tôi không chỉ định bất kỳ phân đoạn nào với yêu cầu này, Elasticsearch sẽ chọn phân đoạn chưa được ký ngẫu nhiên và giải thích lý do tại sao việc phân bổ không thành công. Ví dụ, trên cụm phát triển một nút của tôi, việc gọi lệnh này sẽ tạo ra phản hồi được hiển thị trong hình 15.1.

Có rất nhiều thông tin cần phải xử lý trong phần giải thích phân bổ đầu ragọi. Hình 15.1 cho thấy rằngsách\_lập\_trình2chỉ mục chưa được gán (xemtrạng thái hiện tạithuộc tính). Cácgiải thích\_phân\_bổthuộc tính giải thích lý do

**508 CPHẦN15 *Hiệu suất và xử lý sự cố***

**Chỉ mục chưa được gán**



**Lý do hủy phân công**

**Giải thích chi tiết về phân bổ**

**Hình 15.1 Lý do phân đoạn không được chỉ định**

cho điều này, vàquyết định phân bổ nútchứangười quyết địnhđể cung cấp lời giải thích chính xác. Đoạn trích sau đây cho thấy những người quyết định này:

"người quyết định": [{

"decider": "same\_shard", "decision": "KHÔNG",

"giải thích": "một bản sao của phân đoạn này đã được phân bổ cho nút này [[programming\_books2][0],

nút[x2rSvN03Sfm6mSUMaltTrA],

[P], s[ĐÃ BẮT ĐẦU], a[id=eu1qh4I5THKRmD\_7OcmWYw]]"

},

{

"decider": "disk\_threshold", "decision": "KHÔNG",

"giải thích": "nút nằm trên cụm cài đặt mức nước thấp [cluster.routing.allocation.disk.watermark.low=85%], có dung lượng trống nhỏ hơn dung lượng trống tối thiểu bắt buộc [35gb], dung lượng trống thực tế: [24,9gb], dung lượng sử dụng thực tế: [89,3%]"

}]

* 1. ***Không ổn địnhcác cụm* 509**

Giải thích của người quyết định tiết lộ lý do tại sao việc phân bổ không thành công. Chúng ta cũng có thể lấy thông tin chi tiết của một phân đoạn chưa được chỉ định bằng cùng một API bằng cách cung cấp chỉ mục và phân đoạn.

**Liệt kê 15.13 Nhận lời giải thích cho một mục chưa được gánmảnh vỡ**

NHẬN \_cluster/phân bổ/giải thích

{

"index": "programming\_books2", "shard": 0,

"chính": đúng, "current\_node": "mkmac.local"

}

Truy vấn này lấy giải thích phân bổ cho chỉ mục đã cho. Dựa trên các giải thích, chúng ta có thể bắt đầu khắc phục sự cố để giải quyết các phân đoạn chưa được chỉ định và đưa cụm trở về trạng thái XANH.

* + 1. Sử dụng đĩa ngưỡng

Elasticsearch bảo vệ các cụm khỏi bộ nhớ đĩa thấp bằng cách kích hoạt ngưỡng sử dụng đĩa trên cụm. Các ngưỡng này được đưa ra để ngăn các nút hết dung lượng đĩa và do đó tạo ra một cụm bị lỗi. Ba ngưỡng được cung cấp là hình mờ đĩa ở giai đoạn thấp, cao và ngập lụt để quản trị viên sử dụng cho phù hợp.

**LOW-****DẤU MỜ ĐĨA**

Dấu mờ đĩa thấp là ngưỡng rào cản 85% có hiệu lực nếu bộ nhớ đĩa giảm xuống dưới 85% tổng bộ nhớ. Nếu ngưỡng này bị vượt qua trên một nút cụ thể, Elasticsearch sẽ không phân bổ bất kỳ phân đoạn mới nào trên nút đó cho đến khi bộ nhớ của nút giảm xuống dưới cài đặt này. Ví dụ: nếu chúng ta đã phân bổ 1 TB bộ nhớ đĩa cho một nút và 850 GB đã được sử dụng, Elasticsearch sẽ đưa ra cảnh báo này và thực hiện các biện pháp để ngăn chặn lỗi nút liên quan đến không gian đĩa trống không đủ. Dấu mờ cũng có thể là một giá trị tuyệt đối, chẳng hạn như đặt 200 GB thay vì ngưỡng bộ nhớ được biểu thị dưới dạng phần trăm.

Mặc dù ngưỡng watermark này được đặt ở mức 85% theo mặc định, chúng ta có thể sử dụng API cài đặt cụm để thay đổi nếu cần. Mã sau đây cho thấy cơ chế để thay đổi điều này. Chúng tôi giảm watermark dung lượng đĩa thấp xuống 80%, thấp hơn một chút so với mặc định.

**Liệt kê 15.14 Thiết lập hình mờ đĩa thấp trên cụm**

ĐẶT \_cluster/cài đặt

{

"tạm thời": { "cluster.routing.allocation.disk.watermark.low":"80%"

}

}

**510 CPHẦN15 *Hiệu suất và xử lý sự cố***

**HTÔI****-DẤU MỜ ĐĨA**

Ngưỡng watermark đĩa cao được đặt thành 90% theo mặc định, giúp bảo vệ bộ nhớ của nút không bị giảm xuống dưới 90% bộ nhớ đĩa khả dụng. Khi watermark này chạm đến, Elasticsearch sẽ nỗ lực hết sức để di chuyển (di dời) các mảnh từ nút bị lỗi này sang các nút khác có đủ bộ nhớ.

Tương tự như watermark đĩa thấp, đây là thiết lập động, do đó chúng ta có thể sử dụng API thiết lập cụm để sửa đổi ngưỡng. Danh sách tiếp theo giảm watermark đĩa cao xuống 85% từ mức mặc định là 90% để đảm bảo cảnh báo được phát hành khi vượt quá giới hạn. Lưu ý rằng chúng tôi sử dụng một cách thay thế để thể hiện 85%: 0,85.

**Liệt kê 15.15 Thiết lập hình mờ đĩa cao trên cụm**

ĐẶT \_cluster/cài đặt

{

"tạm thời": { "cluster.routing.allocation.disk.watermark.high":"0.85"

}

}

**FCÂU CHUYỆN-****SÂN KHẤU-DẤU MỜ ĐĨA**

Khi vượt qua ngưỡng watermark high-disk, Elasticsearch sẽ đợi thêm một ngưỡng nữa, ngưỡng watermark flood-stage-disk, trước khi chuyển sang chế độ hoảng loạn. Ngưỡng watermark flood-stage mặc định là 95%: nếu bộ nhớ đĩa trên nút giảm xuống dưới 95%, Elasticsearch sẽ đưa ra cảnh báo này và khiến tất cả các chỉ mục trên nút này không thể ghi. Bất kỳ phân đoạn nào được phân bổ trên nút có bật ngưỡng watermark flood-stage sẽ chuyển thành phân đoạn chỉ đọc.

Ngay khi dung lượng đĩa khả dụng và mức sử dụng đĩa không làm thay đổi ngưỡng hình mờ, hình mờ giai đoạn lũ sẽ được đặt lại. Cũng giống như các hình mờ khác, chúng ta có thể (đặt lại) ngưỡng mặc định của hình mờ giai đoạn lũ để phù hợp với yêu cầu của mình. Trong danh sách này, chúng ta vặn núm trên hình mờ giai đoạn lũ xuống.

**Liệt kê 15.16 Thiết lập hình mờ đĩa giai đoạn lũ trên cụm**

ĐẶT \_cluster/cài đặt

{

"tạm thời": { "cluster.routing.allocation.disk.watermark.flood\_stage":"90%"

}

}

Chúng ta có thể gọi GET \_cluster/settings để lấy tất cả các thiết lập hình mờ để đảm bảo chúng được thiết lập trên cụm:

"tạm thời": {

"cụm": {

"định tuyến": { "phân bổ": {

"đĩa": {

"hình mờ": { "thấp": "80%",

}

}

}

}

...

* 1. ***Bộ ngắt mạch***

"flood\_stage": "90%",

"cao": "85%"

}

}

**511**

Chúng tôi cũng có thể cung cấp tất cả các thiết lập này cùng một lúc. Tôi sẽ để bạn tự thử nghiệm (hoặc xem các tệp trong sách để biết giải pháp).

Các cuộc gọi máy chủ không phản hồi luôn là vấn đề trong các hệ thống phân tán. Đôi khi người dùng phải đợi lâu hơn mức cần thiết chỉ để nhận ra rằng cuộc gọi của họ đã thất bại, nhưng máy chủ không phản hồi đủ nhanh để thông báo cho họ. Để tránh các tình trạng lỗi như vậy, hệ thống phần mềm triển khai bộ ngắt mạch. Elasticsearch cũng có chúng, như chúng ta sẽ thảo luận trong phần sau.

#### Bộ ngắt mạch

Trong các kiến ​​trúc và ứng dụng phân tán, lỗi của các cuộc gọi từ xa từ các dịch vụ không phản hồi là điều hiển nhiên, mặc dù không nên coi đó là điều hiển nhiên. Đây là một vấn đề phổ biến trong thế giới dịch vụ vi mô khi máy khách phải chờ lâu hơn bình thường chỉ để nhận được lỗi. May mắn thay, có một mô hình ngắt mạch để giảm thiểu vấn đề này.

Circuit breaker là phương pháp dự phòng được kích hoạt khi phản hồi vượt ngưỡng thời gian do các vấn đề về phía máy chủ như hết bộ nhớ, khóa tài nguyên, v.v. Giống như việc phải xếp hàng dài để mua iPhone mới, chỉ để thất vọng khi cửa hàng hết hàng—nhưng họ lại tặng bạn một phiếu quà tặng.

Elasticsearch cũng không khác. Đây là một ứng dụng phân tán được kỳ vọng sẽ đưa ra lỗi và ngoại lệ vì nhiều lý do. Elasticsearch triển khai các bộ ngắt mạch để chống lại các vấn đề cản trở tiến trình của khách hàng. Nó đưa ra các lỗi có ý nghĩa cho khách hàng nếu các bộ ngắt mạch được kích hoạt.

Elasticsearch có sáu bộ ngắt mạch cho nhiều tình huống khác nhau, bao gồm một bộ ngắt mạch catch-all cấp độ cha. Ví dụ, nếu tổng bộ nhớ trên nút tăng do tổng mức tiêu thụ của các yêu cầu inflight hiện tại, bộ ngắt mạch inflight-requests sẽ kích hoạt để cứu nút khỏi lỗi. Truy vấn sau sử dụng API nút để lấy giới hạn bộ nhớ hiện tại được đặt trên nhiều bộ ngắt mạch khác nhau.

**Liệt kê 15.17 Đang lấy cài đặt bộ nhớ của máy cắt mạch**

NHẬN \_nodes/stats/breaker

Elasticsearch sẽ đưa ra lỗi khi bộ nhớ cần thiết để thực hiện một thao tác không đủ (hoặc hành động) và nó sẽ kích hoạt các bộ ngắt mạch. Khách hàng sẽ nhận được lỗi “Hết bộ nhớ” ngay lập tức từ các bộ ngắt mạch này. Bảng 15.1 liệt kê các loại bộ ngắt mạch khác nhau, giới hạn của chúng và các thuộc tính liên quan.

**512 CPHẦN15 *Hiệu suất và xử lý sự cố***

**Bảng 15.1 Các loại máy cắt mạch và cài đặt bộ nhớ của chúng**

|  |  |  |
| --- | --- | --- |
| **Bộ ngắt mạch** | **Mô tả và giới hạn bộ nhớ tối thiểu** | **Tài sản** |
| Cha mẹ | Tổng bộ nhớ có thể được sử dụng trên tất cả các bộ ngắt mạch khác. Mặc định là 70% bộ nhớ heap JVM nếu bộ nhớ thời gian thực được tính đến (indices.breaker.total.use  \_real\_memory=true); nếu không, nólà 95% bộ nhớ heap của JVM. | chỉ số.breaker.total.limit |
| Trên chuyến bayyêu cầu | Tổng bộ nhớ của tất cả các yêu cầu đang bay, không được vượt quá ngưỡng. Mặc định là 100% bộ nhớ heap JVM, mặc dù nó lấy phần trăm thực tế từ bộ ngắt mạch chính. | mạng.breaker.inflight  \_yêu cầu.giới hạn |
| Lời yêu cầu | Giúp ngăn chặn vượt quá bộ nhớ heap để phục vụ một yêu cầu duy nhất. Mặc định là 60% heap JVM. | chỉ số.breaker.request  .giới hạn |
| Dữ liệu thực địa | Giúp ngăn ngừa vượt quá bộ nhớ khiđang tảicác lĩnh vực vàocánh đồngbộ nhớ đệm. Giới hạn mặc định là 40% bộ nhớ heap. | chỉ số.breaker.fielddata  .giới hạn |
| Yêu cầu kế toán | Tránh tích tụbộ nhớ sau khi yêu cầu được phục vụ. Mặc định là 100%, nghĩa là nó thừa hưởng ngưỡng từ cha mẹ. | chỉ số.breaker.kế toán  .giới hạn |
| Biên soạn kịch bản | Trong khi tất cả các bộ ngắt mạch khác đều quan tâm đến bộ nhớ, bộ ngắt mạch này giới hạn số lần biên dịch nội tuyến trong một khoảng thời gian cố định. Mặc định là 150/5 phút (150 lần biên dịch tập lệnh trong khoảng thời gian 5 phút). | tập lệnh.max\_compilations  \_tỷ lệ |

Bộ ngắt mạch giúp tránh sử dụng bộ nhớ dư thừa cho một số hoạt động xảy ra thường xuyên. Điều này giúp duy trì cụm ổn định.

#### Lời cuối cùng

Vì có rất nhiều thứ có thể xảy ra sai sót trên một cụm sản xuất, nên không chỉ không thực tế mà còn không thể đề cập đến tất cả các vấn đề về hiệu suất và khắc phục sự cố trong chương này. Những vấn đề chúng ta đã xem xét chỉ là bề nổi. Hầu hết các vấn đề đều cần phải điều tra chi tiết, lập hồ sơ ứng dụng, sàng lọc qua nhật ký, thử và sai một vài tùy chọn, v.v. Một lời khuyên: hãy bình tĩnh và có phương pháp trong hành trình duy trì một cụm khỏe mạnh và ổn định.

Tôi cũng khuyên bạn nên sử dụng cụm Elasticsearch của mình để lái thử để bạn hiểu rõ nó từ trong ra ngoài. Thử nghiệm với các tập dữ liệu lớn hơn (có khả năng chống lỗi trong tương lai) trong

***Bản tóm tắt* 513**

Môi trường không chức năng đảm bảo rằng bạn không chỉ nắm bắt được khía cạnh cơ sở hạ tầng mà còn hiểu chi tiết về số liệu hiệu suất tìm kiếm và I/O của tệp.

Tài liệu chính thức, diễn đàn thảo luận (Stack Overflow) và bài đăng trên blog dành cho kỹ sư có sẵn để giúp bạn quản lý và giám sát cụm. Bạn có thể làm theo hướng dẫn của họ để điều chỉnh cấu hình bộ nhớ để có hiệu suất tối ưu, sử dụng đĩa và cụm chạy trơn tru.

Chương này kết thúc hành trình của chúng ta thông qua việc học, hiểu và làm việc với Elasticsearch. Elasticsearch là một con quái vật phức tạp; nó đòi hỏi kiến ​​thức chuyên sâu và chuyên môn tận tụy để duy trì và điều hành các trang trại sản xuất. Tin tốt là những người của Elastic đã tạo ra tài liệu phong phú về sản phẩm trong nhiều năm (mặc dù đôi khi hơi khô khan và quá sức), điều này có thể giúp bạn khi bạn bị lạc lối—và tất nhiên, bạn có thể tham khảo cuốn sách này!

#### Bản tóm tắt

* Elasticsearch là một công cụ tìm kiếm phức tạp và việc duy trì và quản lý các cụm hoạt động lành mạnh đòi hỏi chuyên môn.
* Tốc độ tìm kiếm chậm hơn là một vấn đề thường gặp mà khách hàng phàn nàn khi tìm kiếm dữ liệu bằng Elasticsearch. Cung cấp phần cứng hiện đại với sự phân bổ bộ nhớ và tài nguyên máy tính phù hợp giúp giảm bớt các vấn đề về tốc độ. Các tùy chọn như chọn kiểu dữ liệu từ khóa và điều chỉnh mô hình dữ liệu tài liệu cũng giúp tăng tốc độ tìm kiếm.
* Tốc độ lập chỉ mục cũng là một mối quan tâm, đặc biệt là khi hệ thống có khối lượng lớn dữ liệu thu thập. Nếu được phép, các biện pháp như sử dụng ID do hệ thống tạo cho tài liệu có thể tăng tốc quá trình lập chỉ mục. Tải dữ liệu bằng API số lượng lớn (thay vì API tài liệu đơn) là một cách chắc chắn để cải thiện hiệu suất thu thập.
* Chúng ta có thể tắt hoặc tăng tốc độ làm mới trong quá trình lập chỉ mục và sau đó bật lại khi quá trình lập chỉ mục hoàn tất. Làm như vậy sẽ khiến các tài liệu không khả dụng ngay lập tức để tìm kiếm nhưng cải thiện hiệu suất lập chỉ mục bằng cách giảm số lần truy cập I/O.
* Do bản chất phân tán của Elasticsearch, mọi thứ có thể trở nên không ổn ở nhiều khu vực, từ cụm đến hệ thống tệp, giao tiếp nút, bộ nhớ, v.v. Duy trì một cụm khỏe mạnh là điều tối quan trọng và người quản trị phải cố gắng duy trì trạng thái cụm XANH. Việc tuân thủ nghiêm ngặt hệ thống quản lý cụm dựa trên đèn giao thông giúp cụm hoạt động trơn tru.
* Đôi khi, các phân đoạn bị bỏ ngỏ và việc khắc phục sự cố để tìm ra nguyên nhân chính xác sẽ giúp chúng tôi gắn lại các nút nếu cần hoặc thực hiện hành động thích hợp.
* Elasticsearch cung cấp ngưỡng cho bộ nhớ đĩa khả dụng, bao gồm các watermark đĩa ở mức thấp, mức cao và mức ngập. Các watermark đĩa ở mức thấp và mức cao cho phép Elasticsearch quản lý việc phân bổ lại phân mảnh và cảnh báo người quản trị về các sự cố cụm sắp tới. Watermark đĩa ở mức ngập là một cảnh báo nghiêm trọng

**514 CPHẦN15 *Hiệu suất và xử lý sự cố***

về vấn đề bộ nhớ ngoài đĩa trên nút. Để giữ cho cụm hoạt động khi điều này xảy ra, Elasticsearch sẽ chuyển tất cả các phân đoạn thành chỉ đọc và không cho phép bất kỳ hoạt động lập chỉ mục nào trên các phân đoạn của nút đó.

* Elasticsearch là một ứng dụng ngốn bộ nhớ và cần triển khai các biện pháp kiểm soát phù hợp để lỗi được truyền ngay lập tức đến máy khách. Ứng dụng này sử dụng các mẫu ngắt mạch để tạo ra các ngắt mạch để máy khách không cần phải chờ lâu hơn mức cần thiết. Mỗi ngắt mạch sẽ kích hoạt Elasticsearch nếu bộ nhớ cho một hành động nhất định chiếm nhiều bộ nhớ hơn dự kiến. Một ngắt mạch con sẽ thừa hưởng ngưỡng bộ nhớ từ một ngắt mạch cha.

# Phụ lục A Cài đặt

Bước đầu tiên khi làm việc với bất kỳ sản phẩm nào là tải xuống và cài đặt sản phẩm đó. Trong phần phụ lục này, chúng tôi tải xuống, cài đặt, cấu hình và chạy Elasticsearch và Kibana. Hướng dẫn cài đặt có sẵn trên kho lưu trữ GitHub của cuốn sách ([http://](http://mng.bz/gBmn) [mng.bz/gBmn](http://mng.bz/gBmn)) và trên trang web của cuốn sách ([https://www.manning.com/books/](https://www.manning.com/books/elasticsearch-in-action-second-edition) [elasticsearch-in-action-phiên bản thứ hai](https://www.manning.com/books/elasticsearch-in-action-second-edition))

Theo mặc định, phiên bản Elasticsearch 8.x đi kèm với tính năng bảo mật được bật. Để đơn giản và để bảo mật không cản trở chúng ta, chúng tôi vô hiệu hóa tính năng này trong phần phụ lục này. Tuy nhiên, không vô hiệu hóa bảo mật trong sản xuất. Chỉnh sửa config/elasticsearch.yml để thêm thuộc tính sau vào cuối tệp:

xpack.security.enabled:SAI

#### Cài đặt Tìm kiếm đàn hồi

Cài đặt Elasticsearch rất dễ dàng. Những người ở Elastic đã nỗ lực rất nhiều để chúng ta có thể cài đặt các sản phẩm theo bất kỳ cách nào chúng ta muốn—sử dụng tệp nhị phân, trình quản lý gói, Docker hoặc thậm chí là đám mây. Một vài phần tiếp theo sẽ trình bày chi tiết các tùy chọn này. Phương pháp tiếp cận ưa thích của chúng tôi đối với cuốn sách này là cài đặt phần mềm bằng cách tải xuống tệp nhị phân, nhưng không có lý do gì bạn không thể thử các tùy chọn khác. Các hướng dẫn trên trang web Elastic rất hữu ích; hãy truy cập trang web để biết chi tiết về các phương pháp cài đặt mà chúng tôi không thảo luận.

* + 1. Tải xuống tệp nhị phân Elasticsearch

Cách dễ nhất để thiết lập máy chủ Elasticsearch trên máy tính của bạn chỉ đơn giản là tải xuống hiện vật đã nén, giải nén và chạy tập lệnh chạy. Truy cập trang tải xuống Elasticsearch tại<https://elastic.co/downloads/elasticsearch> và tải xuống tệp nhị phân tương ứng cho hệ điều hành của bạn. Trang web có các hiện vật có thể tải xuống cho hầu hết mọi hệ điều hành, bao gồm hỗ trợ Docker.

**515**

**516 PHỤ LỤCMỘT*Cài đặt***

Ở đây chúng tôi thử nghiệm trên Windows và macOS, nhưng hướng dẫn dành cho các hệ điều hành khác cũng tương tự và khá đơn giản.

**GHI CHÚ**Chúng tôi sử dụng một đường dẫn đơn giản là cài đặt từ tệp nhị phân và chạy trên máy cục bộ, nhưng thường thì nó được xử lý khác trong thế giới thực. Một nhóm riêng biệt (có thể là DevOps) có thể chịu trách nhiệm cài đặt và cấu hình thiết lập cần thiết và cung cấp cho các nhà phát triển các phiên bản được cấu hình sẵn. Phần mềm có thể ở tại cơ sở hoặc trên đám mây, tùy thuộc vào chiến lược cơ sở hạ tầng CNTT của tổ chức bạn. Hầu hết các nhà cung cấp đám mây như AWS và Azure đều cung cấp các dịch vụ được quản lý. Elastic cũng có dịch vụ đám mây được quản lý, triển khai phần mềm trong AWS, Azure hoặc Google Cloud.

Làm theo hướng dẫn ở đây để thiết lập môi trường phát triển cá nhân của bạn. Chọn đúng tệp nhị phân và tải xuống máy tính của bạn. Bạn có thể tải xuống bất kỳ thư mục nào bạn chọn. Ví dụ, tôi thường tải phần mềm xuống thư mục nền tảng tại <my\_home>/DEV/platform để thuận tiện. Phiên bản hiện tại của Elastic-search là 8.6 khi viết cuốn sách này, nhưng hãy lưu ý rằng Elastic phát hành các phiên bản mới hơn khá nhanh.

**GHI CHÚ** Elasticsearch binary được đóng gói cùng với Java JDK. Theo cách này, nếu bạn không cài đặt Java (hoặc Java không tương thích), Elasticsearch sẽ không phàn nàn về điều đó; nó sẽ tiếp tục công việc của mình, tham chiếu đến Java JDK đóng gói. Nếu bạn muốn cài đặt phiên bản không đóng gói, hãy kiểm tra trang tải xuống, nhấp vào liên kết thích hợp có nội dung là non-JDK binary và đảm bảo JAVA\_HOME được đặt và trỏ đến cài đặt cục bộ của bạn.

Bây giờ bạn đã tải xuống tệp nhị phân, bước tiếp theo là giải nén tệp nhị phân để cài đặt trên máy cục bộ của bạn trong thư mục bạn đã chọn. Chúng tôi sẽ thảo luận về điều này trong hai phần tiếp theo dành cho hệ điều hành Windows và macOS: chọn bất kỳ phần nào áp dụng cho nhu cầu của bạn và bỏ qua phần còn lại.

* + 1. Bắt đầu trên Windows

Sau khi tệp đã tải xuống thành công, hãy giải nén tệp zip vào thư mục cài đặt của bạn. Bảng A.1 giải thích cấu trúc thư mục mà Elasticsearch tuân thủ.

**Bảng A.1 Cấu trúc thư mục Elasticsearch**

|  |  |
| --- | --- |
| **Tên thư mục** | **Chi tiết** |
| thùng rác | Thư mục nhị phân chứa tất cả các tập lệnh để khởi động máy chủ cũng như nhiều tiện ích khác. Chúng ta thường không cần phải chạm vào các tệp thực thi khác, ngoại trừ tệp để khởi động máy chủ: elasticsearch.bat (hoặc .sh, tùy thuộc vào hệ điều hành). |
| cấu hình | Thư mục chứa cấu hình cho máy chủ của chúng tôi, đặc biệt là tệp elasticsearch.yml. Hầu hết các thuộc tính được thiết lập cho chúng tôi để máy chủ bắt đầu với các mặc định hợp lý. |
| các plugin | Một thư mục để lưu trữplugin—các mô-đun phần mềm bổ sung có thể được sử dụng để đưa vào các tính năng Elasticsearch mới, chẳng hạn như tạo trình phân tích văn bản mới |
| các mô-đun | Chứa các mô-đun |

***A.1 Cài đặtTìm kiếm đàn hồi* 517**

**Bảng A.1 Cấu trúc thư mục Elasticsearch (tiếp theo)**

|  |  |
| --- | --- |
| **Tên thư mục** | **Chi tiết** |
| nhật ký  dữ liệu | Thư mục nơi phiên bản Elasticsearch đang chạy của chúng tôi đưa ra dữ liệu ghi nhật ký, bao gồm nhật ký máy chủ và thu thập rác  Thư mục ghi dữ liệu, giống như kho lưu trữ dữ liệu cố định. Tất cả tài liệu được lưu trữ tại đây trên hệ thống tệp của máy tính. |

Sau khi giải nén tệp nhị phân vào thư mục của bạn, hãy khởi động dấu nhắc lệnh với quyền quản trị. Tại dấu nhắc lệnh, hãy đưa ra lệnh thay đổi thư mục (cd) để di chuyển vào thư mục bin của Elasticsearch:

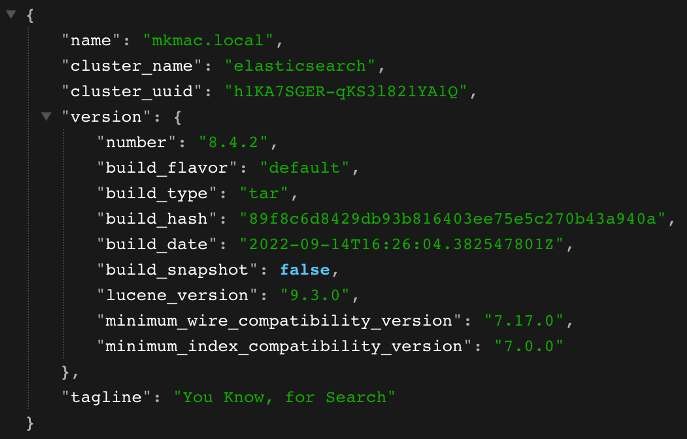
[cmd>cd <INSTALL\_DIR>\elasticsearch\bin](http://mng.bz/5wvD)

Thực hiện elasticsearch.bat để khởi động máy chủ:

cmd> elasticsearch.bat

Nếu mọi việc diễn ra tốt đẹp, bạn sẽ thấy đầu ra như "Server Started" được in trên bảng điều khiển. Ở đây, chúng tôi đã khởi động máy chủ của mình như một phiên bản một nút. Sau khi khởi động, nó sẽ tự động tham gia vào một cụm với một nút (chính nó). Máy chủ có sẵn theo mặc định tại https:/./localhost:9200 (cài đặt mặc định).

Mở trình duyệt web của bạn và truy cập trang chủ Elasticsearch tại https:/./ localhost:9200. Nếu máy chủ đang chạy tốt, phản hồi JSON sẽ trả về từ máy chủ, như thể hiện trong hình A.1. Thông báo “thành công” đơn giản này từ Elasticsearch cho biết máy chủ đang hoạt động và đang chạy.



**Hình A.1 Trang chủ của máy chủ Elasticsearch**

**518 PHỤ LỤCMỘT*Cài đặt***

Một vài thuộc tính có thể cần bạn chú ý. Tên là tên của phiên bản, mặc định là tên máy tính của bạn. Bạn có thể thay đổi bằng cách điều chỉnh cấu hình (như được hiển thị sau).

Thuộc tính quan trọng thứ hai là cluster\_name, chỉ ra tên của cụm mà nút này đã tham gia. Một lần nữa, Elasticsearch cung cấp mặc định, vì vậy tên của cụm hiện tại là elasticsearch, được đặt theo mặc định.

* + 1. Bắt đầu trên macOS

Tải xuống tệp nhị phân macOS (tar.gz) và giải nén vào vị trí bạn muốn, ví dụ /Users/<username>/DEV/platform. Mở terminal và điều hướng đến thư mục bin trong thư mục cài đặt của bạn:

$>cd ~/DEV/nền tảng/elasticsearch/bin

Thực hiệntìm kiếm đàn hồichạy tập lệnh:

$>./elasticsearch

Thao tác này sẽ đưa máy chủ Elasticsearch lên cụm một nút. Đầu ra cho bảng điều khiển với thông báo “Server Started” cho biết máy chủ đã được khởi động thành công. Điều hướng đến http:/./localhost:9200 trên trình duyệt yêu thích của bạn sau khi máy chủ được bật và chạy. Bạn sẽ thấy phản hồi được hiển thị trong hình A.2.

**Hình A.2**

**Phản hồi của máy chủ sử dụngcURL trên macOS**

**Truy cập máy chủ bằng cURL**

Ngoài việc truy cập URL của Elasticsearch qua trình duyệt, bạn có thể sử dụng cURL (tiện ích gọi URL dòng lệnh) để tương tác với máy chủ. Người dùng Unix thích sử dụng cURL để truy cập URL HTTP. Đây là tiện ích dòng lệnh rất phổ biến và hữu ích để giao tiếp với máy chủ qua HTTP. Hầu hết các hệ thống dựa trên Unix đều cài đặt tiện ích này theo mặc định

* 1. ***Cài đặtTìm kiếm đàn hồi* 519**

(hoặc bạn có thể tải xuống trực tuyến). Bạn cũng có thể tải xuống tệp nhị phân cho hệ điều hành Windows từ<https://curl.se/windows>.

Mở một cửa sổ terminal mới và nhập lệnh sau:

$>curl http://localhost:9200

Phản hồi cho biết máy chủ Elasticsearch đang hoạt động tốt (rõ ràng là số phiên bản sẽ khác khi bạn chạy trên thiết lập cục bộ của mình). Thuộc tính quan trọng nhất là cluster\_name. Giá trị mặc định là "elasticsearch". Nếu bạn đang khởi động một nút mới và muốn tham gia cụm này (elasticsearch), tất cả những gì bạn phải làm là đặtcluster\_name trong tệp thuộc tính để khớp với thuộc tính cluster\_name của máy chủ đầu tiên.

* + 1. Cài đặt thông qua Docker

Nếu bạn muốn sử dụng Docker, bạn có thể thực hiện theo hai cách như mô tả ở đây.

**TRONGHÁTDHÌNH ẢNH màu son đất**

Elastic xuất bản hình ảnh Docker của mình trong kho lưu trữ docker.elastic.co. Kéo hình ảnh từ kho lưu trữ docker của Elastic:

docker kéo docker.elastic.co/elasticsearch/elasticsearch:8.6.2

Thao tác này sẽ lấy hình ảnh (dựa trên CentOS) về máy cục bộ của bạn. Sau khihình ảnh được tải xuống, bạn có thể khởi động máy chủ của mình bằng cách gọi lệnh docker run:

docker chạy -p 9200:9200-p 9300:9300 -e "discovery.type=single-node" docker.elastic.co/elasticsearch/elasticsearch:8.6.2

Lệnh này khởi động máy chủ ở chế độ một nút, hiển thị máy chủ ở mức 9200 trên máy chủ cục bộ của bạn.

Sau lệnhthực hiện thành công, hãy truy cập http:/./localhost:9200 trên máy của bạntrình duyệt (hoặc đưa ra lệnh curl) để nhận được phản hồi tích cực từ máy chủ.

**TRONGHÁT DOCKER-SÁNG TẠO**

Các tập tin của cuốn sách bao gồm các tập tin Docker trong thư mục docker tại<http://mng.bz/5wvD>hoặc trên trang web của cuốn sách. Sao chép (hoặc kiểm tra trên máy cục bộ của bạn) tệp elasticsearch-docker-8-6-2.yml. Phát hành lệnh sau tại terminal:

docker-compose lên-f elasticsearch-docker-8-6-2.yml

Lệnh này khởi động vùng chứa Docker của bạn bằng cả dịch vụ Elasticsearch và Kibana.

* + 1. Kiểm tra máy chủ bằng API \_cat

Elasticsearch là một máy chủ web RESTful, nghĩa là giao tiếp với nó rất dễ dàng. Elasticsearch đưa ra một API đặc biệt gọi là cat (văn bản nhỏ gọn và căn chỉnh) với

\_cat endpoint. Mặc dù định dạng JSON thân thiện với máy tính, nhưng con người sẽ thấy khó chịu khi đọc dữ liệu ở định dạng đó. API \_cat không dành cho lập trình

**520 PHỤ LỤCMỘT*Cài đặt***

tiêu thụ nhưng được thiết kế để tạo ra đầu ra dưới dạng bảng (cột) dễ nhìn đối với mắt người.

Bạn có thể tìm danh sách các điểm cuối được API \_cat đưa ra bằng cách truy cập http:/./local host:9200/\_cat trong trình duyệt của bạn hoặc thực hiện lệnh curl này:

curl 'localhost:9200/\_cat'

[Làm như vậy sẽ trả về hơn ba chục điểm cuối (chúng tôi chỉ hiển thị một vài điểm cho ngắn gọn):](https://docker.elastic.co/)

/\_cat/mảnh vỡ

/\_cat/mảnh/{chỉ mục}

/\_cat/nút

/\_cat/chỉ số

/\_cat/chỉ số/{chỉ số}

/\_mèo/đếm

/\_cat/đếm/{chỉ mục}

/\_mèo/sức khỏe

/\_cat/bí danh

...

Hãy xác định tình trạng của cụm bằng cách sử dụng API \_cat. Truy cập http:/./localhost:9200/

\_cat/health hoặc đưa ra lệnh curl này:

curl 'localhost:9200/\_cat/health'

Điểm cuối /\_cat/health hiển thị tình trạng sức khỏe của cụm. Phản hồi cho truy vấn này tương tự như sau:

1615669944 21:12:24 elasticsearch màu vàng 1 1 1 1 0 0 1 0 - 50.0%

Đây là biểu diễn dữ liệu dạng cột, nhưng như bạn thấy, không có tiêu đề cột nào để xác định giá trị. Bạn có thể yêu cầu đầu ra với tiêu đề cột bằng cách thêmthêm av (để diễn đạt chi tiết) vào cuối truy vấn:

curl 'localhost:9200/\_cat/health?v'

Cái nàymang lại kết quả sau:

[**thời đại dấu thời gianclusterstatus node.total node.data shards pri relo init hủy gán pending\_tasks max\_task\_wait\_time active\_shards\_percent**](http://mng.bz/5wvD)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1615669875 21:11:15 elasticsearch màu vàng | 1 | 1 | 1 | 1 |
| 0010 | - |  |  | 50,0% |

Phản hồi cho biết tên cụm là elasticsearch (trường thứ ba), trạng thái của cụm là VÀNG (trường thứ tư), v.v.

#### Cài đặt Kibana

Bây giờ Elasticsearch đã được cài đặt và chạy, bạn cần cài đặt Kibana. Quá trình cài đặt theo cùng một đường dẫn như Elasticsearch và chúng tôi sẽ tóm tắt quy trình này cho Windows và Mac.

***A.2 Cài đặtKibana* 521**

* + 1. Tải xuống tệp nhị phân Kibana

Truy cập trang tải xuống của Kibana (<https://elastic.co/downloads/kibana>) để tải xuống phiên bản mới nhất (8.6.2 tại thời điểm viết cuốn sách này) của Kibana cho hệ điều hành yêu thích của bạn. Giải nén tệp lưu trữ vào thư mục cài đặt của bạn.

* + 1. Kibanatrên Windows

Phát lệnh thay đổi thư mục (cd) để di chuyển đến thư mục bin và thực thi (hoặc nhấp đúp) vào tệp kibana.bat:

cmd:>cd <KIBANA\_INSTALL\_DIR>\bin

**Thay đổi thư mục bin của cài đặt**

cmd:>kibana.bat

**Bắt đầu bảng điều khiển Kibana bằng cách thực thi tập lệnh**

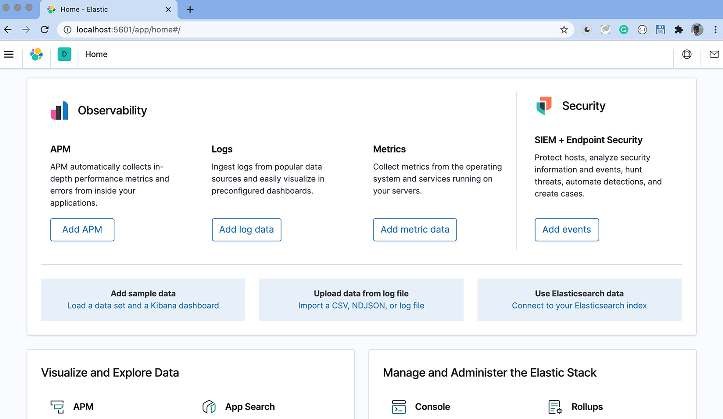
Kibana sẽ chạy trên máy cục bộ của bạn ở cổng 5601.

Hãy đảm bảo máy chủ Elasticsearch vẫn đang hoạt động trong cửa sổ lệnh khác. (Khi Kibana khởi động, nó sẽ tìm kiếm phiên bản Elasticsearch để kết nối. Vì vậy, điều quan trọng là phải giữ Elasticsearch chạy từ bản cài đặt trước đó.) Thao tác này sẽ khởi động máy chủ Kibana, như bạn có thể thấy trong nhật ký:

Trạng thái đã thay đổi từ vàng sang xanh lá cây - Sẵn sàng

nhật ký [23:17:16.980] [thông tin][lắng nghe] Máy chủ đang chạy tại http://localhost:5601 nhật ký [23:17:16.987] [thông tin][máy chủ][Kibana][http] máy chủ http đang chạy tại http:/./localhost:5601

Ứng dụng khởi động và kết nối với máy chủ Elasticsearch của bạn. Khi bạn thấy phản hồi tích cực trong cửa sổ lệnh, hãy mở trình duyệt web và truy cập http:/./ localhost:5601, tại đó bạn sẽ thấy ứng dụng web được hiển thị trong hình A.3.



**Hình A.3 Ứng dụng web Kibana chạy trên máy chủ cục bộ**

**522 PHỤ LỤCMỘT*Cài đặt***

* + 1. Kibana trên macOS

Sau khi giải nén tệp nhị phân, hãy thực thi tập lệnh shell Kibana để chạy tệp đó:

$>cd <KIBANA\_INSTALL\_DIR>/bin

**Thay đổi thư mục bin của cài đặt**

$>./kibana

**Bắt đầu bảng điều khiển Kibana bằng cách thực thi tập lệnh**

Thao tác này sẽ khởi động bảng điều khiển Kibana của bạn trên cổng mặc định 5601. Bạn sẽ thấy kết quả đầu ra sau trong thiết bị đầu cuối:

Trạng thái đã thay đổi từ vàng sang xanh lá cây - Sẵn sàng

nhật ký [23:17:16.980] [thông tin][lắng nghe] Máy chủ đang chạy tại http://localhost:5601 nhật ký [23:17:16.987] [thông tin][máy chủ][Kibana][http] máy chủ http đang chạy tại http:/./localhost:5601

Như nhật ký chỉ ra, khi truy cập http:/./localhost:5601 sẽ chuyển đến trang chủ của Kibana (xem hình A.3) theo mặc định.

* + 1. Cài đặt thông qua Docker

Tương tự như việc cài đặt Elasticsearch, có hai cách để làm việc với Docker như được mô tả tại đây.

**TRONGHÁTDHÌNH ẢNH màu son đất**

Elastic xuất bản hình ảnh Docker của mình trong[https://www.docker.elastic.co](https://www.docker.elastic.co/) kho lưu trữ. Kéo hình ảnh từ kho lưu trữ docker của Elastic:

docker kéo docker.elastic.co/elasticsearch/elasticsearch:8.6.2

Thao tác này sẽ tải hình ảnh về máy cục bộ của bạn. Sau khi hình ảnh được tải xuống, bạn có thể khởi động công cụ của mình bằng cách chạy các lệnh sau:

docker network tạo elastic

docker run --name es01-test --net elastic -p 9200:9200 -p 9300:9300 -e "discovery.type=single-node"

**Tạo ra**

docker.elastic.co/elasticsearch/elasticsearch:7.14.0

**Chạy container**

**một địa phươngmạng**

Lệnh này khởi động máy chủ web trên cổng 5601, kết nối với tìm kiếm Elasticsearch trên cổng 9200. Sau khi lệnh thực thi thành công, hãy truy cập http://localhost:5601 trên trình duyệt của bạn để xem ứng dụng web hoạt động.

**TRONGHÁT DOCKER-SÁNG TẠO**

Các tập tin của cuốn sách bao gồm các tập tin Docker trong thư mục docker tại<http://mng.bz/5wvD>hoặc trên trang web của cuốn sách. Sao chép (hoặc kiểm tra trên máy cục bộ của bạn) tệp elasticsearch-docker-8-6-2.yml. Phát hành lệnh sau tại terminal:

docker-compose lên-f elasticsearch-docker-8-6-2.yml

Thao tác này sẽ khởi động cả Elasticsearch và Kibana trong vùng chứa Docker.

# Phụ lục B Đường ống tiếp nhận

Dữ liệu đi vào Elasticsearch không phải lúc nào cũng sạch. Thông thường, dữ liệu cần được chuyển đổi, làm giàu hoặc định dạng. Có các tùy chọn để làm sạch dữ liệu trước khi đưa vào Elasticsearch để thu thập, chẳng hạn như viết các trình chuyển đổi tùy chỉnh hoặc sử dụng các công cụ ETL (trích xuất, chuyển đổi, tải). Elasticsearch cho phép các khả năng này thông qua các đường ống thu thập cung cấp hỗ trợ hạng nhất để thao tác dữ liệu—chúng ta có thể chia nhỏ, xóa, sửa đổi và nâng cao dữ liệu trước khi thu thập.

#### Tổng quan

Dữ liệu được lập chỉ mục vào Elasticsearch có thể cần phải trải qua quá trình chuyển đổi và thao tác. Hãy xem xét một ví dụ về việc tải hàng triệu tài liệu pháp lý được biểu diễn dưới dạng tệp PDF vào Elasticsearch để tìm kiếm. Mặc dù tải hàng loạt là một cách tiếp cận, nhưng nó không đủ, cồng kềnh và dễ xảy ra lỗi.

Nếu bạn nghĩ chúng ta có thể sử dụng các công cụ ETL cho các tác vụ xử lý dữ liệu như vậy, bạn hoàn toàn đúng. Có rất nhiều công cụ như vậy, bao gồm cả Logstash. Logstash xử lý dữ liệu của chúng ta trước khi nó được lập chỉ mục vào Elasticsearch hoặc được lưu vào cơ sở dữ liệu hoặc các đích khác. Tuy nhiên, nó không nhẹ và đòi hỏi phải thiết lập phức tạp, tốt nhất là trên một máy khác.

Cũng giống như chúng ta viết các truy vấn tìm kiếm bằng Query DSL, chúng ta có thể phát triển các đường ống thu thập dữ liệu với bộ xử lý sử dụng cùng cú pháp và áp dụng chúng vào dữ liệu đến để ETL dữ liệu. Quy trình làm việc rất đơn giản:

**1**Tạo một hoặc nhiều đường ống với logic dự kiến ​​dựa trên các yêu cầu kinh doanh liên quan đến chuyển đổi, cải tiến hoặc làm giàu sẽ được thực hiện trên dữ liệu.

**2** Gọi các đường ống trên dữ liệu đến. Dữ liệu đi qua một loạt các bộ xử lý trong một đường ống và được xử lý ở mọi giai đoạn.

**3** Lập chỉ mục cho dữ liệu đã xử lý.

**523**

**524 PHỤ LỤCB*Đường ống nạp***

Hình B.1 cho thấy hoạt động của hai đường ống độc lập với các bộ xử lý khác nhau. Các đường ống này được lưu trữ/tạo trên một nút nhập. Dữ liệu được xử lý trong khi đi qua các bộ xử lý này trước khi lập chỉ mục. Chúng ta có thể gọi các đường ống này trong quá trình tải hàng loạt hoặc khi lập chỉ mục các tài liệu riêng lẻ.

**Nút nhập**

**Một đường ống tiếp nhận được tạo ra**

**của một loạt bộ xử lý.**

**đính kèm**

bộ xử lý

**bộ**

bộ xử lý

**mò mẫm**

bộ xử lý

**Các tài liệu được đưa qua**

**tiếp nhận các đường ống để chuyển đổi, làm giàu và thao tác.**

**Đường ống nạp**

**Đã xử lý**

**tài liệu**

**làm giàu**

bộ xử lý

**làm rơi**

bộ xử lý

**csv**

bộ xử lý

**Hình B.1 Hai đường ống xử lý dữ liệu độc lập**

Bộ xử lý là một thành phần phần mềm thực hiện một hoạt động chuyển đổi trên dữ liệu đầu vào. Một đường ống được tạo thành từ một loạt các bộ xử lý. Mỗi bộ xử lý được dành riêng để thực hiện một nhiệm vụ của nó. Nó xử lý dữ liệu đầu vào dựa trên logic của nó và đưa ra dữ liệu đã xử lý cho giai đoạn tiếp theo. Chúng ta có thể nối nhiều bộ xử lý tùy theo yêu cầu của chúng ta.

Elasticsearch cung cấp hơn ba chục bộ xử lý ngay khi xuất xưởng. Hãy cùng xem xét cơ chế tạo và sử dụng pipeline.

#### Cơ chế của đường ống dẫn nước vào

Đường ống tiếp nhận giúp chúng ta chuyển đổi và thao tác dữ liệu với thiết lập ít tốn kém hoặc không tốn kém. Đường ống được tạo thành từ một chuỗi bộ xử lý, mỗi bộ xử lý thực hiện một thao tác cụ thể trên dữ liệu đến. Các bộ xử lý tiếp nhận này được tải lên các nút được chỉ định vai trò tiếp nhận; hãy nhớ từ chương 14 rằng mỗi nút trong cụm có thể được chỉ định một vai trò—chủ, dữ liệu, tiếp nhận, ml, v.v.

Hãy lấy ví dụ về dữ liệu hoạt động tuyệt mật của MI5 được tải từ cơ sở dữ liệu vào Elasticsearch để tích hợp tính năng tìm kiếm. Dữ liệu được trích xuất từcơ sở dữ liệu phải được đóng dấu “bảo mật” bằng cách sử dụng trường danh mục (chúng ta có thể mở rộng ví dụ này với các tính năng khác sau) trước khi được lập chỉ mục vào Elasticsearch.

* 1. ***Cơ chế hấp thụđường ống* 525**

Bước đầu tiên để đáp ứng yêu cầu này là tạo một đường ống với bộ xử lý.Chúng ta cần thêm một trường có tên là category với giá trị "confidential" bằng cách sử dụng một bộ xử lý set. Elasticsearch sẽ hiển thị API \_ingest để tạo và kiểm tra các pipeline. Chúng ta có thể tạo một pipeline mới bằng cách sử dụng URL \_ingest/pipeline/<pipeline\_name>. Đoạn mã sau sẽ tạo một pipeline mới với một bộ xử lý có tên là set.

**Liệt kê B.1 Tạo đường ống đóng dấu bảo mật**

ĐẶT \_ingest/pipeline/files\_bí mật\_pipeline

{

"description": "Đóng dấu bảo mật vào hồ sơ (tài liệu)",

"bộ xử lý": [

{

"bộ": {

"field": "category", "value": "bí mật"



**Bộ xử lý thiết lập một trường mới.**

**Chúng tôi cung cấp nhiều bộ xử lý theo chuỗi.**

}

}

]

}

Chúng tôi tạo ra một đường ống tiếp nhận được gọi làđường ống\_tệp\_bảo\_mậtđường ống với mộtbộbộ xử lý.bộcông việc của bộ xử lý là tạo ra một cái mớiloạitrường có giá trị"bảo mật". Khi một tài liệu mới được chuyển qua đường ống này,bộbộ xử lý sẽ thêm một trường được gọi làloạivào tài liệu một cách nhanh chóng.

Sau khi định nghĩa pipeline được tạo, việc thực thi nó sẽ lưu trữ các định nghĩa pipeline trong trạng thái cụm. Bây giờ nó đã sẵn sàng để sử dụng. Tuy nhiên, chúng ta có thể sử dụng lệnh gọi API \_simulate để chạy thử trước khi bắt đầu lập chỉ mục dữ liệu.

**Liệt kê B.2 Tạo đường ống đóng dấu tuyệt mật**

POST \_ingest/pipeline/confidential\_files\_pipeline/\_simulate

{

"tài liệu": [{

"\_nguồn": {

"op\_name": "Chiến dịch Cobra"

}

}

**Mảng docs mong đợi một tập hợp các tài liệu.**

**Trường \_source bao gồm trường bổ sung.**

**URL đường ống với API \_simulate**

]

}

Mã này mô phỏng quá trình thực thi đường ống. Chúng tôi thực thi một tài liệu với một cặp trường-giá trị trên đường ốngconfidential\_files\_pipeline. Việc thực thi mã không lập chỉ mục cho tài liệu; thay vào đó, nó kiểm tra logic của đường ống. Đây là phản hồi khi chúng tôi mô phỏng đường ống:

{

"tài liệu": [

{

"tài liệu": {

**526 PHỤ LỤCB*Đường ống nạp***

"\_index": "\_index",

"\_id": "\_id",

"\_version": "-3", "\_source": {

**"thể loại": "bí mật",**

"op\_name": "Chiến dịch Cobra"

},

"\_ingest": {

"dấu thời gian": "2022-11-03T23:42:33.379569Z"

}

}

}

]

}

Đối tượng \_source bao gồm tài liệu đã sửa đổi của chúng tôi: đường ống thêm

loạicánh đồng. Đây là phép thuật củabộbộ xử lý.

Như đã thảo luận trước đó, chúng ta có thể nối các bộ xử lý. Giả sử chúng ta muốn viết hoa con dấu (“BÍ MẬT”); tất cả những gì chúng ta cần làm là thêm một bộ xử lý khác—gọi là chữ hoa—vào đường ống và chạy lại truy vấn.

**Liệt kê B.3 Nối bộ xử lý thứ hai thành chữ hoa acánh đồng**

ĐẶT \_ingest/pipeline/files\_bí mật\_pipeline

{

"description": "Đóng dấu bí mật vào hồ sơ (tài liệu)", "processors": [

{

"bộ": {

"field": "category", "value": "bí mật"

},

"chữ hoa": { "trường": "thể loại"

}

}

]

}

**Bộ xử lý viết hoa là tính năng mới được bổ sung.**

Chúng tôi đã thêm một bộ xử lý chữ hoa để hai bộ xử lý được nối với nhau: đầu ra từ bộ xử lý đầu tiên trở thành đầu vào cho bộ xử lý thứ hai. Kết quả như sau:

"\_nguồn": {

"category": "BÍ MẬT", "op\_name": "Chiến dịch Cobra"

}

Trường danh mục được thêm vào bởibộ xử lý thiết lập và trường được viết hoa bằng bộ xử lý viết hoa, tạo ra dấu CONFIDENTIAL trên tài liệu cuối cùng. Trong phần tiếp theo, chúng ta hãy xem một ví dụ thực tế về việc tải PDF bằng đường ống nhập.

* 1. ***Đang tải PDF vàoTìm kiếm đàn hồi* 527**

#### B.3 Đang tảiPDF vào Elasticsearch

Giả sử yêu cầu kinh doanh của chúng ta là tải các tệp PDF (như tài liệu pháp lý hoặc tạp chí y khoa ở định dạng PDF) vào Elasticsearch, cho phép khách hàng thực hiện tìm kiếm trên các tệp đó. Elasticsearch cho phép chúng ta lập chỉ mục các tệp PDF bằng bộ xử lý nhập chuyên dụng được gọi là tệp đính kèm.

Bộ xử lý tệp đính kèm được sử dụng trong đường ống thu thập để tải tệp đính kèm: tệp PDF, tài liệu Word, email, v.v. Nó sử dụng Apache Tika ([https://](https://tika.apache.org/) [tika.apache.org](https://tika.apache.org/)) thư viện để trích xuất dữ liệu tệp. Dữ liệu nguồn phải được chuyển đổi sang định dạng Base64 trước khi tải vào đường ống. Chúng ta hãy xem điều này hoạt động như thế nào.

Tiếp tục ví dụ MI5 của chúng tôi, chúng tôi dự kiến ​​sẽ tải tất cả dữ liệu bí mật được trình bày trong các tệp PDF vào Elasticsearch. Các bước sau đây giúp trực quan hóa quy trình:

**1**Xác định một đường ống với bộ xử lý tệp đính kèm. Nội dung Base64 của tệp được lập chỉ mục vào một trường (trong ví dụ của chúng tôi, chúng tôi xác định trường là secret\_file\_ data).

**2**Chuyển đổi nội dung tệp PDF thành byte và đưa vào tiện ích mã hóa Base64 (sử dụng bất kỳ bộ công cụ nào bạn có).

**3**Gọi đường ống cho dữ liệu đến để bộ xử lý tệp đính kèm xử lý dữ liệu.

Mã sau đâytạo một đường ống với bộ xử lý đính kèm.

**Danh sách B.4 Đường ống có bộ xử lý đính kèm**

ĐẶT \_ingest/pipeline/files\_pdf\_bí mật\_pipeline

{

"description": "Đường ống để tải tài liệu PDF", "bộ xử lý": [

{

"bộ": {

"field": "category", "value": "bí mật"

},

"tệp đính kèm": {

"trường": "secret\_file\_data"

}

}

]

}

**Bộ xử lý tệp đính kèm có trường secret\_file\_data**

Khi chúng ta thực thi mã này,đường ống dẫn file pdf bí mậtđược tạo ra trên cụm.đính kèmbộ xử lý mong đợi dữ liệu được mã hóa Base64 từ một tập tin được đặt trongdữ liệu\_tệp\_bí\_mậttrường trong quá trình hút vào đường ống.

Bây giờ chúng ta đã tạo xong pipeline, hãy thử nghiệm nó. Giả sử dữ liệu tệp là “Sunday Lunch at Konda's” (có lẽ đây là mã để loại bỏ ông Konda!), và chúng ta chạy bộ mã hóa Base64 để tạo dữ liệu ở dạng được mã hóa Base64. (Tôi sẽ để bạn tự áp dụng bộ mã hóa—xem thanh bên “Mã hóa Base64” để biết thêm thông tin.)

**528 PHỤ LỤCB*Đường ống nạp***

Chúng ta có thể mô phỏng đường ống bằng cách truyền chuỗi Base64 và kiểm tra xem đầu ra có phải là thứ chúng ta mong đợi hay không (nó sẽ trả ra “Sunday Lunch at Konda's”). Mã sau đây kiểm tra đường ống khi hoạt động. Lưu ý rằng U3VuZGF5IEx1bmNoIGF0IEtvbm- RhJ3M= là tệp PDF được mã hóa Base64 với một thông điệp bí mật (“Sunday Lunch at Konda's”).

**Liệt kê B.5 Mô phỏng đường ống**

POST \_ingest/pipeline/confidential\_pdf\_files\_pipeline/\_simulate

{

"tài liệu": [

{

"\_nguồn": {

"op\_name": "Op Konda", "secret\_file\_data:"U3VuZGF5IEx1bmNoIGF0IEtvbmRhJ3M="

}

}

]

}

Giá trị trường secret\_file\_data được thiết lập thủ công bằng chuỗi được mã hóa Base64, sau đó được đưa vào đường ống.

**Mã hóa Base64**

Java có mộtjava.util.Base64lớp mã hóa; tương tự như vậy, Python có mộtcơ sở64chống lại-ule. Có khả năng cao là hỗ trợ Base64 trong ngôn ngữ lập trình/viết kịch bản mà bạn chọn. Lưu ý rằng chúng ta phải chuyển đổi đầu vào thành byte trước khi có thể áp dụng mã hóa Base64.

Chúng ta có thể sử dụng bất kỳ ngôn ngữ lập trình hoặc khuôn khổ tập lệnh nào để định vị và tải tệp, chuyển đổi chúng thành byte và đưa chúng vào bộ xử lý. Liệt kê B.5 bao gồm một đoạn mã cho ngôn ngữ Java và Python. Mã đầy đủ có sẵn trong các tệp của cuốn sách: Python (<http://mng.bz/6D46>) và Java (<http://mng.bz/o1Zv>).

Dựa trên định nghĩa đường ống (liệt kê B.4), bộ xử lý đính kèm mong đợitrường secret\_file\_data với dữ liệu được mã hóa và chúng tôi đã cung cấp nó khi mô phỏng đường ống. Sau đây là phản hồi từ thử nghiệm:

...

"tài liệu": {

"\_index": "\_index",

"\_id": "\_id",

"\_version": "-3", "\_source": {

"op\_name": "Op Konda", "category": "bí mật", "tệp đính kèm": {

"content\_type": "text/plain; charset=ISO-8859-1", "language": "et",

***B.3 Tải PDF vào Elasticsearch***

"content": "Bữa trưa Chủ Nhật tại Konda", "content\_length": 24

},

**529**

"secret\_file\_data": "U3VuZGF5IEx1bmNoIGF0IEtvbmRhJ3M="

},

"\_ingest": {

"dấu thời gian": "2022-11-04T23:19:05.772094Z"

}

}

...

Phản hồi tạo ra một đối tượng bổ sung có tên là attachment với một vài trường—content là dạng đã giải mã của tệp PDF. Siêu dữ liệu bổ sung có sẵn cho tệp đính kèm trong content\_length, language và các trường khác; dữ liệu được mã hóa gốc có sẵn trong trường secret\_file\_data. Chúng ta có thể chọn những trường nào chúng ta muốn duy trì như một phần của siêu dữ liệu. Ví dụ, đoạn mã sau chỉ đặt content, bỏ qua các giá trị siêu dữ liệu khác:

ĐẶT \_ingest/pipeline/only\_content\_pdf\_files\_pipeline

{

"description": "Đường ống để tải tài liệu PDF", "bộ xử lý": [

{

"bộ": {

"field": "category", "value": "bí mật"

},

"tệp đính kèm": {

"trường": "secret\_file\_data", "thuộc tính":["nội dung"]

}

}

]

}

**Chỉ đặt trường nội dung được lập chỉ mục**

Elasticsearch cung cấp nhiều bộ xử lý ingest và chúng phù hợp với nhiều yêu cầu. Như bạn có thể tưởng tượng, việc xem xét tất cả chúng trong phần phụ lục này sẽ không thực tế. Lời khuyên của tôi là hãy xem tài liệu tại<http://mng.bz/nWZ4> và thử nghiệm với mã.

# phụ lục C

*Khách hàng*

Một trong những điểm mạnh của Elasticsearch là cơ chế giao diện phong phú được cung cấp sẵn cho nhiều máy khách để thực hiện tìm kiếm và tổng hợp. Cho dù chúng ta sử dụng Java, Python, C#, JavaScript hay ngôn ngữ lập trình chính nào khác hỗ trợ RESTful API qua giao thức HTTP, thì luôn có sự hỗ trợ hạng nhất.

Việc đưa API lên giao diện RESTful là một quyết định kiến ​​trúc hiện đại giúp tạo ra các sản phẩm không phụ thuộc vào ngôn ngữ. Elastic đã đi theo con đường này để tạo ra một sản phẩm có thể tích hợp với nhiều ngôn ngữ.

Hiện tại, Elasticsearch có thể tích hợp với các máy khách sau: Java, Python, .NET, Ruby, Go, JavaScript, Perl và PHP, cũng như các máy khách do cộng đồng đóng góp như C++, Kotlin, Scala, Swift, Rust, Haskell, Erlang và một số máy khách khác. Elasticsearch rất vui lòng xem xét yêu cầu về máy khách tùy chỉnh nếu cần. Gửi yêu cầu kéo cho nhóm và đưa vào hoạt động.

Việc thảo luận về tất cả các khách hàng này là không thực tế (và tôi không có đủ kinh nghiệm với các ngôn ngữ khác ngoài Java, Kotlin, Scala, Python, Go, JavaScript và một số ngôn ngữ khác). Vì vậy, phần phụ lục này cung cấp cơ chế truy vấn cấp cao cho Java, Java-Script và Python.

#### Máy khách Java

Elasticsearch được viết bằng Java và như bạn có thể mong đợi, Elasticsearch cung cấp hỗ trợ gốc để gọi API Elasticsearch bằng thư viện Java Client. Nó được xây dựng bằng các mẫu xây dựng API trôi chảy và hỗ trợ cả lệnh gọi API đồng bộ và không đồng bộ (chặn và không chặn). Nó yêu cầu Java 8 ở mức tối thiểu, vì vậy hãy đảm bảo rằng ứng dụng của bạn ít nhất là trên Java 8.

Trong vài phần tiếp theo, chúng ta sẽ xem xét tích hợp Elasticsearch từ ứng dụng dựa trên Java Maven/Gradle cũng như ứng dụng Java dựa trên Spring.

**530**

***C.3 Dự án Maven/Gradlecài đặt* 531**

**Dữ liệu Spring Elasticsearch**

Nếu bạn đang làm việc trên một ứng dụng Java dựa trên Spring framework, có một cách khác để tích hợp nó với Elasticsearch: Spring Data Elasticsearch. Dự án này giúp chúng ta kết nối và truy vấn Elasticsearch bằng các mẫu Spring quen thuộc và đã được chứng minh như mẫu và kho lưu trữ. Cũng giống như chúng ta tạo một kho lưu trữ để biểu diễn một lớp cơ sở dữ liệu, Spring Data Elasticsearch cho phép chúng ta sử dụng lớp kho lưu trữ để truy vấn Elasticsearch. Tham khảo tài liệu để biết thêm thông tin về dự án Spring Data Elasticsearch:<http://mng.bz/vnyr>.

#### Lý lịch

Elasticsearch đã phát hành một máy khách Java tương thích ngược mới, Java API Client, từ phiên bản 7.17. Máy khách Java API là một máy khách hiện đại tuân theo một mô hình máy khách tính năng hợp lý với các yêu cầu và phản hồi được gõ mạnh. Phiên bản trước đó của máy khách là Java High-Level REST Client (xem ghi chú bên dưới)—đã thu hút khá nhiều lời chỉ trích cũng như trở thành một vấn đề đau đầu trong việc duy trì và quản lý đối với những người dùng elastic do bản chất vốn có của cách thiết kế bằng cách phụ thuộc vào mã chung được chia sẻ với máy chủ Elasticsearch.

Những người theo Elastic đã nhận ra nhu cầu về một máy khách hiện đại—độc lập với cơ sở mã của máy chủ, với mã API máy khách được tạo dựa trên lược đồ API của máy chủ và cung cấp một mẫu “máy khách tính năng” (chúng ta sẽ sớm thấy điều này trong hành động). Điều này dẫn đến việc tạo ra một máy khách Java API. Đây là máy khách nhẹ thế hệ tiếp theo có mã được tạo ra khá nhiều (99%!), bao gồm các mẫu trình xây dựng API trôi chảy và tự động sắp xếp và giải sắp xếp các Đối tượng Java thành JSON và ngược lại.

**Máy khách REST cấp cao Java**

Elasticsearch được thiết kế lại để thoát khỏi Java cấp caoREST client (tôi sẽ viết tắt cho phần này để dễ tham khảo hơn sau này trên JHLRC) và giới thiệu một client dựa trên API có tên là Java API client. Có một vài vấn đề với JHLRC. Cụ thể, nó được ghép nối với các phiên bản cụ thể của máy chủ Elasticsearch vì nó chia sẻ mã chung với máy chủ Elasticsearch. Tất cả các API đều được viết thủ công, nghĩa là phải bảo trì nhiều theo thời gian, dẫn đến dễ xảy ra lỗi. Sự phụ thuộc vào cơ sở mã máy chủ có nghĩa là khả năng tương thích ngược/tiến bị ảnh hưởng.

#### Thiết lập dự án Maven/Gradle

Các lớp liên quan đến ElasticsearchClient được cung cấp trong một hiện vật jar có thể được tải xuống như một phần của các phụ thuộc trong dự án của chúng tôi. Để thuận tiện, tôi đã tạo một dự án dựa trên Maven có sẵn trên GitHub (<http://mng.bz/4Dmv>) và trang web của cuốn sách.

Thông thường, cần có hai phụ thuộc để đưa vào các lớp liên quan. Chúng được khai báo trong tệp pom.

**532 PHỤ LỤCC*Khách hàng***

**Danh sách C.1 Mavensự phụ thuộc**

<phụ thuộc>

<phụ thuộc>

<groupId>co.elastic.clients</groupId>

<artifactId>elasticsearch-java</artifactId>

<phiên bản>8.5.3</phiên bản>

</phụ thuộc>

<phụ thuộc>

<groupId>com.fasterxml.jackson.core</groupId>

<artifactId>jackson-databind</artifactId>

<phiên bản>2.12.7</phiên bản>

</phụ thuộc>

...

</phụ thuộc>

TÔITôi đang sử dụng phiên bản thư viện mới nhất và tuyệt vời nhất tại thời điểm viết phần này: phiên bản 8.5.3 của sản phẩm máy khách elasticsearch-java và phiên bản 2.12.7 của thư viện lõi Jackson. Bạn có thể cần nâng cấp các thư viện này dựa trên nhu cầu của mình.

Nếu bạn sử dụng Gradle, hãy thêm các thành phần sau đây làm thành phần phụ thuộc vào tệp dựng Gradle của bạn.

**Liệt kê C.2 Các phụ thuộc của Gradle**

phụ thuộc {

triển khai 'co.elastic.clients:elasticsearch-java:8.5.3' triển khai 'com.fasterxml.jackson.core:jackson-databind:2.12.7'

}

Sau khi thiết lập dự án, bước tiếp theo là khởi tạo máy khách và đưa nó vào hoạt động.

#### Khởi tạo

Hãy xem xét việc khởi tạo máy khách và cách chúng ta có thể đưa nó vào hoạt động. Lớp máy khách làco.elastic.clients.elasticsearch.ElasticsearchClientvà được khởi tạo bằng cách cung cấp đối tượng vận chuyển (co.elastic.clients.transport.ElasticsearchTrans-port) cho hàm tạo của nó. Đối tượng vận chuyển này, đến lượt nó, cầnnghỉ ngơiKhách hàngvà các đối tượng ánh xạ JSON. Hãy cùng xem các bước sau:

**1**Tạo đối tượng RestClient đóng gói HttpHost của Apache, trỏ đến URL của máy chủ Elasticsearch.

**Liệt kê C.3 Khởi tạoKhách hàng nghỉ ngơi**

RestClient restClient = RestClient.builder(

mới HttpHost("localhost", 9200)).build();

Trình xây dựng RestClient được tạo và chấp nhận điểm cuối Elasticsearch của chúng tôi được hiển thị trên máy chủ cục bộ tại cổng 9200.

* 1. ***Không gian tênkhách hàng* 533**

**2**Tạo đối tượng vận chuyển bằng cách sử dụng mã trong danh sách C.4. Elastic-Đối tượng searchTransport được xây dựng bằng phiên bản rest-Client đã khởi tạo trước đó và trình ánh xạ JSON (chúng tôi sử dụng trình ánh xạ Jackson JSON trong phiên bản này).

**Liệt kê C.4 Xây dựng đối tượng Vận chuyển**

JacksonJsonpMapper jsonMapper = new JacksonJsonpMapper();

ElasticsearchTransport elasticsearchTransport =

RestClientTransport mới(restClient, jsonMapper);

**Tạo một JacksonJsonpMapper mới**

**sự vật**

**Tạo đối tượng vận chuyển**

Chúng tôi vượt quanghỉ ngơiKhách hàngVàjsonMapper, được khởi tạo trước khi tạo đối tượng vận chuyển.

**1** Mang đếnElasticsearchClientđể sống động hơn với đoạn mã sau.

**Liệt kê C.5 Xây dựngkhách hàng**

ElasticsearchClient elasticsearchClient =

ElasticsearchClient mới(elasticsearchTransport);

ElasticsearchClient chỉ cần đối tượng transport được tạo ra cách đây một lúc. Về cơ bản là vậy—chúng ta có một client và đã đến lúc sử dụng nó để tương tác với Elasticsearch.

#### Khách hàng không gian tên

Trước đó, chúng tôi đã đề cập rằng Java API client tuân theo một mẫu các yêu cầu, phản hồi và các client “tính năng”. Elasticsearch có khái niệm về tên gói và không gian tên cho mỗi tính năng mà nó đưa ra—ví dụ, các API liên quan đến cụm có sẵn trong

Gói \*.cluster, lập chỉ mục các API liên quan đến hoạt động trong \*.index, v.v. Trình khách API Java tuân theo cùng một mô hình: một "trình khách" được cung cấp cho mọi tính năng, như thể hiện trong hình C.1.

Ví dụ, tất cả các lớp liên quan đến chỉ mục như yêu cầu, phản hồi và máy khách lập chỉ mục đều tồn tại trongco.elastic.clients.elasticsearch.indicesgói. Khách hàng cho không gian tên này làElasticsearchIndicesClientvà được lấy từ chínhElasticsearchClient(chúng ta thấy nó hoạt động trong một khoảnh khắc). Tất cả các hoạt động chỉ mục được mong đợi sẽ được thực hiện bởi điều nàyElasticsearchIndicesClient. Tương tự như vậy, tất cả các tính năng khác đều tuân theo cùng một mô hình—một “thư mục” cho mỗi tính năng (không gian tên), với mỗi tính năng có một máy khách được đặt tênElasticsearchFEATUREClient.

Có thể bạn sẽ thấy choáng ngợp khi thấy số lượng các lớp, nhưng các ví dụ sau đây về việc sử dụng máy khách Java API để lấy máy khách tính năng/không gian tên để hỗ trợ tính năng đó làm rõ rằng phần còn lại của thông số kỹ thuật máy khách cũng tuân theo cùng một cách tiếp cận. Hãy cùng xem cách tạo chỉ mục.

**534 PHỤ LỤCC*Khách hàng***

**Không gian tên khách hàng và các lớp liên quan**



**Hình C.1 ElasticsearchMáy khách không gian tên Java**

#### Tạo một chỉ mục

Hãy tạo ra mộtchuyến baylập chỉ mục bằng cách sử dụng máy khách Java API này. Tất cả các hoạt động liên quan đến chỉ mục tạo thành một không gian tên được gọi làchỉ số, vì vậy các lớp và một máy khách liên quan đến các hoạt động trên các chỉ mục nằm trongco.elastic.clients.elasticsearch.indicesgói. Như mong đợi, máy khách hỗ trợ các hoạt động chỉ mục làElasticsearchIndicesClient. Để có được máy khách này, chúng ta phải yêu cầu máy khách Java API chính (ElasticsearchClient) để cung cấp một phiên bản của máy khách:

ElasticsearchIndicesClient elasticsearchIndicesClient = this.elasticsearchClient.indices();

* 1. ***Tạo mộtchỉ số* 535**

Gọi làchỉ số()chức năng trênelasticsearchKhách hàngtrả về ví dụElastic- ticsearchIndicesClient. Khi chúng ta có máy khách, chúng ta có thể gọitạo nên()phương pháp để tạo một chỉ mục.tạo nên()phương pháp mong đợi mộtTạoIndexRequestđối tượng. Mẫu này dẫn đến mẫu yêu cầu/phản hồi tiếp theo của chúng ta.

Tất cả các phương thức trên máy khách được mong đợi sẽ được truyền vào với một đối tượng yêu cầu. Có nhiều lớp yêu cầu và mỗi đối tượng yêu cầu được khởi tạo bằng cách sử dụng mẫu xây dựng.

Giả sử chúng ta cần tạo một chỉ mục. Điều này yêu cầu CreateIndexRequest phải được khởi tạo với các đối số cần thiết để tạo một chỉ mục. Mã sau đây tạo CreateIndexRequest bằng cách sử dụng một trình xây dựng:

CreateIndexRequest tạoIndexRequest =

mới CreateIndexRequest.Builder().index("chuyến bay").build();

Phương thức index() chấp nhận một chuỗi làm tên của chỉ mục. Khi chỉ mục được tạoSau đó, chúng ta có thể gọi phương thức create() trên máy khách bằng cách truyền yêu cầu này:

CreateIndexResponse createIndexResponse = elasticsearchIndicesClient.create(createIndexRequest);

Cuộc gọi này triệu hồitạo nên()phương pháp trênelasticsearchIndicesClient, gửi truy vấn đến máy chủ Elasticsearch để tạo chỉ mục.

Kết quả của lệnh gọi này được lưu trong đối tượng phản hồi: CreateIndex-Response, trong trường hợp này. Một lần nữa, bất kỳ lệnh gọi nào cũng sẽ trả về phản hồi theo cùng một mẫu—ví dụ, phản hồi của CreateIndexRequest là đối tượng CreateIndexResponse. Đối tượng phản hồi có tất cả thông tin cần thiết về chỉ mục mới được tạo.

Phương pháp đầy đủ được đưa ra trong danh sách C.6. Mã nguồn cho toàn bộ lớp học cùng với dự án có sẵn trong các tệp của cuốn sách.

**Liệt kê C.6 Tạo chỉ mục bằng cách sử dụngElasticsearchIndicesClient**

/\*\*

* Phương pháp tạo chỉ mục bằng ElasticsearchIndicesClient chuẩn

\*

* @param tên chỉ mục
* @throws Ngoại lệ IO

\*/

public void createIndexUsingClient(String indexName) throws IOException { ElasticsearchIndicesClient elasticsearchIndicesClient =

this.elasticsearchClient.indices();CreateIndexRequest tạoIndexRequest =

mới CreateIndexRequest.Builder().index(indexName).build();

CreateIndexResponse createIndexResponse = elasticsearchIndicesClient.create(createIndexRequest);

System.out.println("Đã tạo chỉ mục thành công: "+createIndexResponse);

}

**536 PHỤ LỤCC*Khách hàng***

Chúng ta có thể cải thiện mã này. Thay vì tạo raElasticsearchIndicesClient

riêng biệt, chúng ta có thể sử dụng trình xây dựng như thể hiện trong danh sách sau.

**Liệt kê C.7 Tạo chỉ mục bằng cách sử dụng mẫu xây dựng**

/\*\*

* Một phương pháp để tạo chỉ mục bằng cách sử dụng mẫu Builder
* @param tên chỉ mục
* @throws Ngoại lệ IO

\*/

public void createIndexUsingBuilder(String indexName) ném IOException {

CreateIndexResponse createIndexResponse = this.elasticsearchClient

.indices().create(new CreateIndexRequest.Builder()

**.index(tên chỉ mục)**

**.xây dựng()**);

System.out.println("Đã tạo chỉ mục thành công bằng Builder"+createIndexResponse);

}

Chúng tôi không tạo raElasticsearchIndicesClienttrong phương pháp này. Thay vào đó, chúng ta truyền đối tượng yêu cầu (như một trình xây dựng) đếntạo nên()phương pháp (cácchỉ số()phương pháp lấyElasticsearchIndicesClienttrên đótạo nên()phương pháp này được gọi là phương pháp ẩn).

Chúng ta có thể tiến xa hơn nữa—chúng ta có thể sử dụng hàm lambda để làm cho mã ngắn gọn hơn.

**Liệt kê C.8 Tạo chỉ mục bằng Lambdabiểu thức**

/\*\*

* Một phương pháp để tạo chỉ mục bằng cách sử dụng biểu thức Lambda
* @param tên chỉ mục
* @throws Ngoại lệ IO

\*/

public void createIndexUsingLambda(String indexName) ném IOException {

CreateIndexResponse createIndexResponse = this.elasticsearchClient.indices().create(

**yêu cầu -> request.index(indexName)**

);

System.out.println("Chỉ mục đã được tạo thành công bằng Lambda"

+createIndexResponse);

}

Phần in đậm của mã là biểu thức lambda. Nó nói rằng với một yêu cầu (là mộtTạoIndexRequest.Builder()loại đối tượng), tạo một chỉ mục bằng cách sử dụngElasticsearchIndicesClientkhách hàng.

* 1. ***Lập chỉ mụctài liệu* 537**

Vì một chỉ mục có thể được tạo bằng cách sử dụng lược đồ, với các thiết lập cũng như bí danh, chúng ta có thể nối các phương thức đó trong yêu cầu:

CreateIndexRequest createIndexRequest = new CreateIndexRequest.Builder()

.index(tên chỉ mục)

**Tạo một chỉ mục với tên đã cho**

**Thêm cài đặtđến chỉ mục**

.ánh xạ(..)

.cài đặt(..)

.bí danh(..)

.xây dựng();

**Tạo một tập hợp các ánh xạ**

**Tạo bí danh cho chỉ mục**



Tiếp theo, chúng ta hãy xem cách lập chỉ mục tài liệu chuyến bay vào mục lục của chúng ta.

#### Lập chỉ mục tài liệu

Trong chương 13, chúng ta đã thảo luận về việc lập chỉ mục tài liệu bằng cách sử dụng Query DSL trong Kibana Console (hoặc cURL). Ví dụ, mã này lập chỉ mục tài liệu chuyến bay với ID ngẫu nhiên làm khóa chính.

**Liệt kê C.9 Lập chỉ mục một tài liệu bằng cách sử dụng QueryDSL**

Chuyến bay POST/\_doc

{

"tuyến đường":"London tới New York", "tên":"BA123",

"hãng hàng không":"British Airways", "duration\_hours":5

}

Ở đây, chúng ta gọi điểm cuối \_doc bằng phương thức POST với các chi tiết tài liệu được bao gồm dưới dạng đối tượng JSON. ID tài liệu được tạo tự động bởi Elastic-search. Chúng ta có thể truy xuất tài liệu này bằng cách phát hành GET flights/\_doc/\_search.

Hãy lập chỉ mục cùng một tài liệu, nhưng lần này sử dụng máy khách Java API. Hãy nhớ rằng,chúng ta đã tạo chỉ mục chuyến bay ở phần trước, vì vậy tất cả những gì chúng ta cần làm là sử dụng Fluent API để xây dựng truy vấn.

**Danh sách C.10 Lập chỉ mục tài liệu bằng Java APIkhách hàng**

public void indexDocument(String indexName, Flight flight) ném IOException

{

IndexResponse indexResponse = this.elasticsearchClient.index( i -> i.index(indexName)

.document(chuyến bay)

);

System.out.println("Đã lập chỉ mục tài liệu thành công"+indexResponse);

}

Thực hiện truy vấn này sẽ lập chỉ mục chuyến bay vào chỉ mục chuyến bay của chúng tôi. ElasticsearchCli-ent đưa ra phương thức index có thể kết hợp với các phương thức khác như id và document. Trong trường hợp này, chúng ta không sử dụng ID vì chúng ta để hệ thống tạo ra nó.

**538 PHỤ LỤCC*Khách hàng***

Phương thức tài liệu mong đợi đối tượng chuyến bay; bạn có để ý rằng chúng ta không chuyển đổi đối tượng Java chuyến bay thành JSON không? Đó là vì chúng ta đã ủy quyền trách nhiệm sắp xếp và hủy sắp xếp cho lớp JacksonJasonpMapper mà chúng ta đã liên kết với đối tượng vận chuyển trước đó.

CácelasticsearchClient.index()phương pháp mất mộtChỉ mụcYêu cầuđối tượng và nhổ raChỉ sốPhản hồi(điều này phù hợp với những gì chúng ta đã thảo luận về yêu cầu và phản hồi).

#### Tìm kiếm

Tìm kiếm dữ liệu bằng truy vấn với Java API client theo một đường dẫn tương tự: gọi phương thức search() trong lớp ElasticsearchClient bằng cách truyền truy vấn bắt buộc. Tuy nhiên, có một điểm khác biệt tinh tế—các tính năng khác hiển thị một client cho mỗi không gian tên, nhưng tính năng tìm kiếm thì không. Hãy cùng xem điều này hoạt động như thế nào.

Giả sử chúng ta muốn tìm kiếm một tuyến đường từ London đến New York. Chúng ta tạo một truy vấn khớp, cung cấp “London New York” làm tiêu chí tìm kiếm so với trường tuyến đường trong DSL được định dạng tốt khi làm việc với Kibana.

**Danh sách C.11 Tìm kiếm một tuyến đường bằng cách sử dụngcuộc thi đấutruy vấn**

GET chuyến bay/\_tìm kiếm

{

"truy vấn": {

"cuộc thi đấu": {

"tuyến đường": "London New York"

}

}

}

Truy vấn khớp lệnh đơn giản này kiểm tra tuyến đường có từ khóa “London New York”. Chúng ta có thể nhận được một hoặc hai kết quả tùy thuộc vào số lượng bản ghi chúng ta đã lập chỉ mục. Truy vấn này cũng có thể được viết bằng Java bằng cách sử dụng Java API client.

**Danh sách C.12 Tìm kiếm bằng Java APIkhách hàng**

this.elasticsearchClient.search(searchRequest -> searchRequest

.index(tên chỉ mục)

.query(Trình xây dựng truy vấn ->

queryBuilder.match(matchQBuilder-> matchQBuilder.field("tuyến đường")

.query(văn bản tìm kiếm)))

,Lớp bay

);

Cáctìm kiếm()phương pháp mong đợi một yêu cầu tìm kiếm, được cung cấp dưới dạng biểu thức lambda cho phương pháp.truy vấnđược viết bằng cách sử dụng một hàm lambda khác—cho mộtyêu cầu truy vấn (Truy vấn.Trình xây dựngđối tượng), chúng tôi gọicuộc thi đấuchức năng vớiMatch-Query.Builderđối tượng. JSON được chuyển đổi thànhChuyến bayĐối tượng Java, đó là lý do tại saoChuyến bay.lớpđược cung cấp như là đối số chotruy vấnphương pháp.

***C.8 Tìm kiếm* 539**

Phản hồi từtìm kiếm()phương pháp là mộtTìm kiếmPhản hồi—vì vậy chúng ta có thể ghi lại kết quả như hiển thị ở đây:

SearchResponse searchResponse = this.elasticsearchClient.search(..)

searchResponse bao gồm các kết quả là các lượt truy cập, chúng ta có thể lặp lại để lấy danh sách các chuyến bay được trả về. Danh sách sau đây hiển thị toàn bộ yêu cầu tìm kiếm, bao gồm các chuyến bay được trả về trong phản hồi.

**Danh sách C.13 Tìm kiếm bằng cách sử dụng máy khách Java API vớilượt truy cập**

public void search(String indexName, String field, String searchText) ném IOException {

SearchResponse searchResponse = this.elasticsearchClient.search(searchRequest -> searchRequest

.index(tên chỉ mục)

.query(Trình xây dựng truy vấn -> Trình xây dựng truy vấn

.match(matchQueryBuilder -> matchQueryBuilder

,Lớp bay

);

.field("tuyến đường")

.query(văn bản tìm kiếm)))

**Chiếm lấy**

Danh sách <Chuyến bay> chuyến bay =

**chuyến bay...**

(Danh sách<Chuyến bay>) searchResponse.hits().hits()

.stream().collect(Collectors.toList());

tìm kiếmPhản hồi.lượt truy cập().lượt truy cập()

.stream().forEach(Hệ thống.out::println);

}

**... hoặc in chúng ra bảng điều khiển.**

Đối tượng searchResponse có kết quả trong mảng hits—chúng ta chỉ cần đảm bảo chuyển đổi các hits sang các đối tượng miền thích hợp (Flight, trong trường hợp này). Mã nguồn đầy đủ của ví dụ này có sẵn trong các tệp của sách. Bạn có thể tìm hiểu về các máy khách khác trong tài liệu tại đây:<http://mng.bz/QPMQ>.

**540 PHỤ LỤCC *Khách hàng***

# chỉ số

**Biểu tượng**

\_aka API[206,](#_bookmark934)[210](#_bookmark959)

Điểm cuối \_alias[204](#_bookmark929)

API \_bí danh[20](#_bookmark933)6–207

\_tất cả điểm cuối[209](#_bookmark953)

\_phân tích API[242,](#_bookmark1104)[245,](#_bookmark1118)[266,](#_bookmark1216)[268,](#_bookmark1224)[270–271,](#_bookmark1237)

[273](#_bookmark1250)[–274,](#_bookmark1261)[278](#_bookmark1274)

\_phân tích điểm cuối[241,](#_bookmark1095)[248,](#_bookmark1129)[263](#_bookmark1195)

\_phân tích thử nghiệm[241](#_bookmark1096)

\_API số lượng lớn[284,](#_bookmark1297)[338,](#_bookmark1564)[363,](#_bookmark1698)[459,](#_bookmark2101)[503](#_bookmark2318)

lập chỉ mục hàng loạt tài liệu[186–188](#_bookmark865) yêu cầu số lượng lớnsử dụng cURL[189–190](#_bookmark876) định dạng của[185–186](#_bookmark861)

các thực thể độc lập và nhiều hành động[188](#_bookmark868)[–189](#_bookmark874)

lập chỉ mục tài liệu bằng cách sử dụng[39–41](#_bookmark161)

\_MèoAPI[77,](#_bookmark334)[51](#_bookmark2370)9–520

\_điểm cuối cụm[232](#_bookmark1056)

\_cluster/điểm cuối cài đặt[232](#_bookmark1057)

Điểm cuối \_component\_template[215](#_bookmark989)

\_đếm API[31](#_bookmark120)

\_điểm cuối đếm[31](#_bookmark123)

\_tạo API[15](#_bookmark693)6–158

\_tạo điểm cuối[157,](#_bookmark694)[178](#_bookmark812)

\_xóa\_bởi\_truy\_cập\_hành[182](#_bookmark841)

\_docAPI[178](#_bookmark813)

\_doc loại tài liệu[70](#_bookmark316)

Điểm cuối \_doc[29,](#_bookmark111)[154,](#_bookmark687)[166,](#_bookmark746)[182,](#_bookmark836)[537](#_bookmark2444)

\_thực hiện điểm cuối[487](#_bookmark2225)

\_giải thích API[304](#_bookmark1415)

\_giải thích điểm cuối[303](#_bookmark1411)

Thuộc tính \_id[154,](#_bookmark678)[165](#_bookmark732)

\_id cặp[164](#_bookmark727)

\_điểm cuối của ilm[229](#_bookmark1045)

Thuộc tính \_index[154](#_bookmark679)

\_tài liệu index [164](#_bookmark728)

Điểm cuối \_index\_template[213](#_bookmark981)

\_ingest API[525](#_bookmark2393)

\_ingest/đường ống/URL [525](#_bookmark2394)

\_điểm cuối ánh xạ[110,](#_bookmark476)[11](#_bookmark488)[2–113,](#_bookmark493)[117](#_bookmark527)

\_mget API[165](#_bookmark737)

Điểm cuối \_mget[164](#_bookmark729)

\_mget API đa tài liệu[163](#_bookmark723)

\_tên thuộc tính[383](#_bookmark1763)

\_mở API[211–212](#_bookmark970)

\_refresh điểm cuối[505](#_bookmark2321)

\_reindex API[190](#_bookmark882)

\_khôi phục điểm cuối [483](#_bookmark2210)

\_API chuyển đổi [22](#_bookmark1027)4–226

\_cuộc gọi chuyển tiếp[226](#_bookmark1032)

\_lệnh rollover[226](#_bookmark1033)

\_điểm cuối rollover[225](#_bookmark1031)

Thuộc tính \_score[41,](#_bookmark172)[291,](#_bookmark1332)[306](#_bookmark1423)

\_API tìm kiếm[32,](#_bookmark127)[36,](#_bookmark150)[304](#_bookmark1416)

\_điểm cuối tìm kiếm[33–35,](#_bookmark140)[53,](#_bookmark249)[285–286,](#_bookmark1304)[291–292,](#_bookmark1339)

[295,](#_bookmark1364)[301,](#_bookmark1402)[303,](#_bookmark1413)[341,](#_bookmark1583)[364,](#_bookmark1702)[435](#_bookmark1980)

\_cài đặt API[198,](#_bookmark912)[221](#_bookmark1015)

\_điểm cuối cài đặt[160,](#_bookmark705)[200](#_bookmark914)

Thuộc tính \_shards[291](#_bookmark1328)

\_mô phỏng cuộc gọi API[525](#_bookmark2397)

\_slm API[484](#_bookmark2216)

\_API chụp nhanh[479–](#_bookmark2192)[482](#_bookmark2209)

\_thuộc tính nguồn[162,](#_bookmark718)[166](#_bookmark745)

\_điểm cuối nguồn[166,](#_bookmark747)[168](#_bookmark754)

\_cờ nguồn[167,](#_bookmark750)[307,](#_bookmark1430)[309](#_bookmark1438)

\_đối tượng nguồn[310,](#_bookmark1441)[526](#_bookmark2398)

\_tùy chọn nguồn [3](#_bookmark1439)09–310

\_tham số nguồn[437](#_bookmark1998)

thẻ \_nguồn[32](#_bookmark131)

tham số \_source\_excludes [169,](#_bookmark756)[437](#_bookmark1999)

tham số \_source\_includes[168–169,](#_bookmark755)[437](#_bookmark2000)

**541**

**542**

\_API chia tách[221](#_bookmark1016)

\_thống kê API[217](#_bookmark998)

Điểm cuối \_stats[217](#_bookmark999)

\_thuộc tính loại[154](#_bookmark680)

\_cập nhật API

thêm các trường mới[171–172](#_bookmark770) sửa đổi các trường hiện có[1](#_bookmark773)72–173

\_cập nhật điểm cuối[171,](#_bookmark769)[173](#_bookmark780)

Điểm cuối \_update\_by\_query[180](#_bookmark827)

hoạt động \_update\_by\_query[180–181](#_bookmark826)

\_cờ phiên bản[182](#_bookmark837)

--cờ dữ liệu nhị phân[189](#_bookmark877)

\*.gói cụm[533](#_bookmark2426)

Điểm cuối /\_cat/health[520](#_bookmark2371)

**Số học**

Hình dạng 2D (hai chiều)[412](#_bookmark1890)

**MỘT**

Kế toán yêu cầu ngắt mạch[512](#_bookmark2351) thuộc tính action.auto\_create\_index[196](#_bookmark899)

cờ action.auto\_create\_index[158,](#_bookmark696)[194](#_bookmark892) thuộc tính action.destructive\_requires\_name

[209](#_bookmark954)[–211](#_bookmark965)

hành động đối tượng[230](#_bookmark1046)

mảng diễn viên[17](#_bookmark775)2–174

thêm hành động[225](#_bookmark1029)

thêm phương pháp[1](#_bookmark785)73–174Khóa ngoại ADDRESS\_ID[69](#_bookmark308) sự quản lý[467–496](#_bookmark2143)

cấu hình nâng cao[4](#_bookmark2230)88–491

Tùy chọn máy ảo Java[490–491](#_bookmark2249) tùy chọn ghi nhật ký[488–490](#_bookmark2237)

tập tin cấu hình chính[488](#_bookmark2234) nhóm chủ[491–495](#_bookmark2261)

cụm nút[491](#_bookmark2264)

trạng thái cụm[49](#_bookmark2273)2–493

các nút chính chuyên dụng[4](#_bookmark2284)94–495bầu cử tổng thống[491–492](#_bookmark2267)

đủ số lượng[493](#_bookmark2278)

vấn đề não chia đôi [493–4](#_bookmark2280)[94](#_bookmark2282)

giao tiếp nút [474](#_bookmark2166)[–475](#_bookmark2171)

cụm mở rộng[468–474](#_bookmark2145)

thêm các nút vào cụm[4](#_bookmark2147)69–470sức khỏe cụm[470–](#_bookmark2153)[473](#_bookmark2157)

tăng thông lượng đọc[473–474](#_bookmark2159) kích thước mảnh vỡ[47](#_bookmark2173)5–477

thiết lập nhiều chỉ mục[476–477](#_bookmark2178) thiết lập chỉ mục đơn[47](#_bookmark2175)5–476

ảnh chụp nhanh[477–](#_bookmark2184)[487](#_bookmark2226)

tự động hóa[48](#_bookmark2215)3–487

tạo ra[481–4](#_bookmark2201)[82](#_bookmark2206)

xóa bỏ[483](#_bookmark2212)

**MỤC LỤC**

Tổng quan[478–47](#_bookmark2186)[9](#_bookmark2188)

đăng ký kho lưu trữ ảnh chụp nhanh[479–4](#_bookmark2190)[81](#_bookmark2198) khôi phục[482](#_bookmark2208)

truy vấn tổng hợp[437](#_bookmark2001)

tổng hợp[52](#_bookmark244)[–58,](#_bookmark272)[434–46](#_bookmark1971)[6](#_bookmark2141)

xô[56–58,](#_bookmark273)[44](#_bookmark2047)5–455

tổng hợp cấp độ con[450](#_bookmark2066)[–451](#_bookmark2068) tổng hợp phạm vi tùy chỉnh[451–](#_bookmark2071)[453](#_bookmark2077) thùng biểu đồ histogram[56–57](#_bookmark267)

biểu đồ histogram[446–450](#_bookmark2050)

tổng hợp nhiều thuật ngữ[454](#_bookmark2083)[–455](#_bookmark2087)

thùng đựng phạm vi[57](#_bookmark269)–58

thuật ngữ tổng hợp[453–454](#_bookmark2079) kết hợp tìm kiếm và[436](#_bookmark1990) điểm cuối và cú pháp[435–4](#_bookmark1979)[36](#_bookmark1983) bỏ qua kết quả[437](#_bookmark1997)

mét[53–56,](#_bookmark262)[43](#_bookmark2005)7–445

số liệu trung bình[440–4](#_bookmark2020)[41](#_bookmark2022)

số liệu về số lượng [444–445](#_bookmark2042)

số liệu thống kê mở rộng[443](#_bookmark2036)

số liệu tối đa[442](#_bookmark2030)

số liệu tối thiểu[441–442](#_bookmark2026)

số liệu khác[55](#_bookmark258)–56

dữ liệu mẫu[438](#_bookmark2010)

số liệu thống kê[442–](#_bookmark2032)[443](#_bookmark2034)

tổng số mét[54–5](#_bookmark256)[5,](#_bookmark257)[441](#_bookmark2024)

số liệu value\_count[438–440](#_bookmark2012) nhiều và lồng nhau[436–43](#_bookmark1993)[7](#_bookmark1995) Tổng quan[435–43](#_bookmark1977)[7](#_bookmark2003)

cha mẹ[456](#_bookmark2090)

đường ống[458–46](#_bookmark2096)[5](#_bookmark2140)

có sẵn[461–46](#_bookmark2109)[2](#_bookmark2130)

tổng hợp cha tích lũy[46](#_bookmark2133)[2–464](#_bookmark2135) tổng hợp max\_bucket[46](#_bookmark2137)4–465

tổng hợp min\_bucket[465](#_bookmark2139)

dữ liệu mẫu[459](#_bookmark2100)

cú pháp cho[459–4](#_bookmark2103)[61](#_bookmark2107)

các loại[459](#_bookmark2098) Truy vấn DSL cho[296](#_bookmark1374) anh em ruột[457–458](#_bookmark2093)

đối tượng tổng hợp[435](#_bookmark1981)

đối tượng aggs[53](#_bookmark250)

Công cụ AI (trí tuệ nhân tạo)[2,](#_bookmark7)[16](#_bookmark77)

bí danh[202–207](#_bookmark926)

được định nghĩa[73](#_bookmark325)

xóa bỏ[210](#_bookmark958)

di chuyển dữ liệu với thời gian chết bằng không[204](#_bookmark931)[–206](#_bookmark932) nhiều hoạt động sử dụng API \_aliases[206–](#_bookmark933)[207](#_bookmark935) hoạt động chuyển đổi[224–227](#_bookmark1026)

tạo bí danh cho[225](#_bookmark1028) phát hành[225–227](#_bookmark1030)

đối tượng bí danh[202](#_bookmark927)

thuộc tính allocate\_explanation[507](#_bookmark2335)

API phân bổ[507](#_bookmark2334)

thuộc tính allow\_expensive\_queries[327](#_bookmark1522)

Loại chữ cái[241](#_bookmark1097)

**MỤC LỤC 543**

Tìm kiếm mở của Amazon[9](#_bookmark36) Dịch vụ web của Amazon (AWS)[9](#_bookmark37) thuộc tính phân tích[246](#_bookmark1122)

máy phân tích[238–243](#_bookmark1077)

giải phẫu của[23](#_bookmark1088)9–241

được xây dựng sẵn[243–25](#_bookmark1111)[8](#_bookmark1171)

máy phân tích dấu vân tay[253–2](#_bookmark1149)[54](#_bookmark1154)

phân tích từ khóa[252–253](#_bookmark1145)

máy phân tích ngôn ngữ[256–25](#_bookmark1165)[8](#_bookmark1172)

máy phân tích mẫu[254–255](#_bookmark1156)

máy phân tích đơn giản[251](#_bookmark1138)

máy phân tích chuẩn[244–250](#_bookmark1117)

bộ phân tích khoảng trắng[251–2](#_bookmark1141)[52](#_bookmark1142) các thành phần của

bộ lọc ký tự[265–270](#_bookmark1210)

bộ lọc mã thông báo[27](#_bookmark1260)4–278

bộ phân tích[270](#_bookmark1234)[–274](#_bookmark1258) phong tục

tùy chỉnh nâng cao[2](#_bookmark1183)60–261

Tổng quan[258–2](#_bookmark1175)[61](#_bookmark1187)

chuẩn hóa[239](#_bookmark1085)

Tổng quan[238](#_bookmark1074)

chỉ định[261–](#_bookmark1189)[265](#_bookmark1207)

để lập chỉ mục[2](#_bookmark1191)61–263

để tìm kiếm[2](#_bookmark1198)63–265

thử nghiệm[241–24](#_bookmark1094)[3](#_bookmark1108)

cấu hình máy phân tích trên bay[242–243](#_bookmark1103) kiểm tra phân tích rõ ràng[242](#_bookmark1100)

phân chia[2](#_bookmark1080)38–239

ngày neo[324](#_bookmark1512)

Toán tử AND[37–39,](#_bookmark159)[293,](#_bookmark1348)[343,](#_bookmark1594)[353,](#_bookmark1655)[355](#_bookmark1666)

Toán tử AND/OR[345](#_bookmark1610)

Apache Solr[8–](#_bookmark33)[9](#_bookmark35)

Tìm kiếm ứng dụng [12](#_bookmark59) kiểu dữ liệu mảng[139](#_bookmark629)

công cụ trí tuệ nhân tạo (AI)[2,](#_bookmark7)[16](#_bookmark77)

đối tượng đính kèm[529](#_bookmark2407)

bộ xử lý đính kèm[527](#_bookmark2404)[–528](#_bookmark2406) kiểu dữ liệu đính kèm[135](#_bookmark613) tệp đính kèm đối tượng bên trong[134](#_bookmark610) đối tượng đính kèm[134–1](#_bookmark606)[35](#_bookmark614)

tài sản đính kèm[134](#_bookmark611)

đối tượng tác giả[38](#_bookmark156)

tài sản của tác giả[22,](#_bookmark89)[415](#_bookmark1908)

chìa khóa tác giả/tiêu đề[455](#_bookmark2084)

Cài đặt TỰ ĐỘNG[332,](#_bookmark1542)[344](#_bookmark1604)

Trung bình (avg\_bucket)[461](#_bookmark2120)

tổng hợp trung bình[53](#_bookmark245)

tuyên bố trung bình[440](#_bookmark2021)

hàm trung bình[435,](#_bookmark1972)[440](#_bookmark2018)

số liệu trung bình[440](#_bookmark2020)[–441](#_bookmark2022)

thời trang trung bình [395](#_bookmark1811)

avg\_rating\_per\_bucket thùng[451](#_bookmark2069) AWS (Dịch vụ web của Amazon)[9](#_bookmark37)

**B**

mô-đun base64[528](#_bookmark2405)

BBOX (truy vấn hộp giới hạn)[408](#_bookmark1877) Nhịp đập[12](#_bookmark52)

thuật toán best\_fields[349](#_bookmark1630)

loại best\_fields[348–](#_bookmark1628)[349,](#_bookmark1629)[351](#_bookmark1643)

chiến lược lĩnh vực tốt nhất[389](#_bookmark1789) kiểu dữ liệu nhị phân[115](#_bookmark507)

BKD (Khối k chiều)[87,](#_bookmark380)[102](#_bookmark435)

Bloch, Joshua[23,](#_bookmark91)[49–50](#_bookmark221)

cờ bom tấn[175](#_bookmark797)

Thuật toán BM25 (Best Match 25)[41,](#_bookmark173)[89,](#_bookmark387)[91,](#_bookmark396)

94–95[,](#_bookmark410)[390](#_bookmark1795)

truy vấn hợp chất bool[371](#_bookmark1732) khai báo bool[368](#_bookmark1718)

đối tượng bool[368](#_bookmark1719)

truy vấn bool[48–4](#_bookmark217)[9,](#_bookmark218)[365–384](#_bookmark1711) kết hợp tất cả các mệnh đề[381–3](#_bookmark1758)[82](#_bookmark1760) phải mệnh đề[367](#_bookmark1716)[–369](#_bookmark1726)

mệnh đề must\_not[370–373](#_bookmark1728)

được đặt tên[382–384](#_bookmark1762)

nên mệnh đề[37](#_bookmark1740)3–378

tăng điểm[3](#_bookmark1744)74–377

điều khoản lọc[378–](#_bookmark1752)[381](#_bookmark1756)

thiết lập minimum\_should\_match[377–3](#_bookmark1747)[78](#_bookmark1750)

cấu trúc của[366](#_bookmark1713)[–367](#_bookmark1714)

truy vấn hợp chất tìm kiếm bool[136](#_bookmark619) Truy vấn Boolean (bool)[297,](#_bookmark1377)[365](#_bookmark1704)

Kiểu dữ liệu boolean[127–128,](#_bookmark568)[439](#_bookmark2014) thuật toán tương tự boolean[94](#_bookmark408) kiểu boolean[116](#_bookmark519)

thuộc tính tăng cường[385](#_bookmark1770)

Tham số tăng cường[35,](#_bookmark142)[339](#_bookmark1570)

tham số boost\_mode[395](#_bookmark1818) Truy vấn tăng cường (boosting)[365](#_bookmark1707) tăng cường truy vấn[3](#_bookmark1779)[86–388,](#_bookmark1784)[397](#_bookmark1824)

thuộc tính bottom\_right[407–](#_bookmark1873)[408](#_bookmark1875)

toạ độ bottom\_right[40](#_bookmark1867)5–406

bottom\_right geopoint[132](#_bookmark600)

tham số bottom\_right[407](#_bookmark1871)

truy vấn bounding\_box[398–3](#_bookmark1832)[99](#_bookmark1834) truy vấn hộp giới hạn (BBOX)[408](#_bookmark1877) tổng hợp xô[56–58,](#_bookmark273)[445–455](#_bookmark2047)

tổng hợp cấp độ con[450](#_bookmark2066)[–451](#_bookmark2068) tổng hợp phạm vi tùy chỉnh[451–](#_bookmark2071)[453](#_bookmark2077) biểu đồ histogram[56–5](#_bookmark267)[7,](#_bookmark268)[446–450](#_bookmark2050)

biểu đồ ngày[4](#_bookmark2053)47–448

thiết lập khoảng thời gian cho biểu đồ ngày[448–4](#_bookmark2057)[50](#_bookmark2063) tổng hợp nhiều thuật ngữ[454](#_bookmark2083)[–455](#_bookmark2087)

thùng đựng phạm vi[57](#_bookmark269)–58

thuật ngữ tổng hợp[453–454](#_bookmark2079)

Tương quan thùng (bucket\_correlation)[461](#_bookmark2122) Số lượng thùng (bucket\_count\_ks\_test)[461](#_bookmark2121) Bộ chọn thùng (bucket\_selector) [461](#_bookmark2110) Sắp xếp theo nhóm (bucket\_sort)[461](#_bookmark2112)

**544**

xô[56,](#_bookmark266)[445](#_bookmark2048)

Tập lệnh thùng (buckets\_script)[461](#_bookmark2111) thiết lập buckets\_path[460](#_bookmark2105)

biến buckets\_path [459,](#_bookmark2104) [461](#_bookmark2106) hoạt động tài liệu số lượng lớn[185–190](#_bookmark856)

lập chỉ mục hàng loạt tài liệu[186–188](#_bookmark865) yêu cầu số lượng lớnsử dụng cURL[189–190](#_bookmark876) định dạng của \_bulk API[185–1](#_bookmark861)[86](#_bookmark863)

các thực thể độc lập và nhiều hành động[188](#_bookmark868)[–189](#_bookmark874)

tạo hành động[188](#_bookmark869)

xóa hành động[189](#_bookmark873)

cập nhật hành động [189](#_bookmark871) vấn đề về tốc độ chỉ số[503](#_bookmark2317)

hoạt động hàng loạt[39](#_bookmark162)

chỉ số trạm xe buýt[401](#_bookmark1846)

phân tích kinh doanh[13](#_bookmark63) Kiểu dữ liệu byte[117](#_bookmark526)

**C**

khoảng thời gian lịch[448](#_bookmark2058)

thuộc tính calendar\_interval[448](#_bookmark2059)

Trạng thái ĐÃ HỦY [317](#_bookmark1472)

số liệu về số lượng[444–44](#_bookmark2042)[5](#_bookmark2045)

Loại CarItems[70](#_bookmark313)

cd (lệnh)[517,](#_bookmark2361)[521](#_bookmark2377)

chỉ số điều tra dân số[123](#_bookmark555)

Thay đổi điểm (change\_point)[461](#_bookmark2123) đối tượng char\_filter[258](#_bookmark1176)

phần char\_filter[261](#_bookmark1184)

bộ lọc ký tự[265–270](#_bookmark1210)

bộ lọc hmtl\_strip [266–267](#_bookmark1215) bộ lọc ký tự ánh xạ[2](#_bookmark1222)67–268ánh xạ qua tập tin[268–269](#_bookmark1227)

bộ lọc ký tự pattern\_replace[269–2](#_bookmark1230)[70](#_bookmark1232) chỉ mục trò chuyện[46](#_bookmark2150)9–471

tổng hợp cấp độ con[45](#_bookmark2066)0–451

cầu dao điện[511–512](#_bookmark2346)

khách hàng[523,](#_bookmark2382)[530,](#_bookmark2410)[539](#_bookmark2460)

tạo chỉ mục[534–537](#_bookmark2430)

lập chỉ mục tài liệu[537–538](#_bookmark2443)

khởi tạo[5](#_bookmark2417)32–533

Máy khách Java[530](#_bookmark2411)[–531](#_bookmark2412)

Thiết lập dự án Maven/Gradle[531–532](#_bookmark2414) không gian tên khách hàng[533](#_bookmark2425)

tìm kiếm[538–](#_bookmark2454)[539](#_bookmark2461)

thuộc tính cluster\_name[51](#_bookmark2362)8–519

thuộc tính cluster.election.duration[492](#_bookmark2268)

thuộc tính cluster.election.initial\_timeout [492](#_bookmark2269)

thuộc tính cluster.follower\_lag.timeout[493](#_bookmark2275)

thuộc tính cluster.indices.close.enable[211](#_bookmark967)

thuộc tính cluster.name [81–](#_bookmark347)[82](#_bookmark349)

thuộc tính cluster.publish.timeout[492](#_bookmark2274) các cụm

sức khỏe cụm[470–](#_bookmark2153)[473,](#_bookmark2157)[50](#_bookmark2330)6–507

**MỤC LỤC**

nhóm chủ[4](#_bookmark2261)91–495

trạng thái cụm[492](#_bookmark2273)[–493](#_bookmark2276)

các nút chính chuyên dụng[494](#_bookmark2284)[–495](#_bookmark2287) bầu cử tổng thống[491–492](#_bookmark2267)

nút chính[491](#_bookmark2264)

đủ số lượng[493](#_bookmark2278)

vấn đề não chia đôi [493–494](#_bookmark2280)

đa cụm đa nút [83](#_bookmark352)–84

tỉ lệ[468–474](#_bookmark2145)

thêm các nút vào cụm[46](#_bookmark2147)9–470sức khỏe cụm[47](#_bookmark2153)0–473

tăng thông lượng đọc[473–474](#_bookmark2159) cụm nút đơn[81–82](#_bookmark346)

không ổn định[506–51](#_bookmark2328)[1](#_bookmark2342)

sức khỏe cụm[50](#_bookmark2330)6–507

ngưỡng sử dụng đĩa[509–5](#_bookmark2338)[11](#_bookmark2343)

các mảnh vỡ chưa được chỉ định[507–509](#_bookmark2333) co.elastic.clients.elasticsearch.ElasticsearchClient

lớp học[532](#_bookmark2418) co.elastic.clients.elasticsearch.indices

bưu kiện[533–53](#_bookmark2427)[4](#_bookmark2431) co.elastic.clients.transport.ElasticsearchTransport

sự vật[532](#_bookmark2419)

lệnh (cd)[517,](#_bookmark2361)[521](#_bookmark2377)

[tiện ích gọi URL dòng lệnh.](#_bookmark2365)*Nhìn thấy*kiểu dữ liệu hoàn thành cURL[116,](#_bookmark522)[120,](#_bookmark540)[131,](#_bookmark590)[14](#_bookmark648)6–147

mẫu thành phần[212](#_bookmark977) mẫu chỉ mục có thể cấu hình [212](#_bookmark978)

truy vấn hợp chất[47–52,](#_bookmark241)[2](#_bookmark1376)[96–297,](#_bookmark1379)[36](#_bookmark1686)1–396

truy vấn bool[48–4](#_bookmark217)[9,](#_bookmark218)[365–384](#_bookmark1711) kết hợp tất cả các mệnh đề[381–3](#_bookmark1758)[82](#_bookmark1760) phải mệnh đề[367](#_bookmark1716)[–369](#_bookmark1726)

mệnh đề must\_not[370–373](#_bookmark1728)

được đặt tên[382–384](#_bookmark1762)

nên mệnh đề[37](#_bookmark1740)3–378

cấu trúc của[366](#_bookmark1713)[–367](#_bookmark1714)

tăng cường truy vấn[3](#_bookmark1779)86–388

điểm số không đổi[384–](#_bookmark1768)[386](#_bookmark1776)

truy vấn dis\_max[3](#_bookmark1786)88–389

điều khoản lọc[51–5](#_bookmark238)[2](#_bookmark242)

truy vấn function\_score[389–3](#_bookmark1793)[95](#_bookmark1819) kết hợp điểm số chức năng[394–](#_bookmark1808)[395](#_bookmark1820) hàm field\_value\_factor[393–](#_bookmark1804)[394](#_bookmark1806)

hàm random\_score[391](#_bookmark1797)[–392](#_bookmark1799)

hàm script\_score[392–3](#_bookmark1801)[93](#_bookmark1802)

phải mệnh đề[49–50](#_bookmark220)

mệnh đề must\_not[50–51](#_bookmark228)

Tổng quan[364–36](#_bookmark1701)[5](#_bookmark1709)

dữ liệu mẫu[362](#_bookmark1689)[–363](#_bookmark1699)

sản phẩm lập chỉ mục[363](#_bookmark1697)

sơ đồ[362–363](#_bookmark1691)

nên mệnh đề[51](#_bookmark234)

khái niệm và nền tảng[347](#_bookmark1618) thẻ khái niệm[347](#_bookmark1620)

Con dấu BÍ MẬT[526](#_bookmark2401)

đường ống\_tệp\_bảo\_mật [525](#_bookmark2396)

**MỤC LỤC 545**

Truy vấn điểm số không đổi (constant\_score)[365](#_bookmark1705) điểm số không đổi[384–386](#_bookmark1768)

Kiểu dữ liệu constant\_keyword[123–124](#_bookmark554) hàm constant\_score[3](#_bookmark1773)85–386

truy vấn constant\_score[288,](#_bookmark1311)[362,](#_bookmark1687)[385–3](#_bookmark1771)[86](#_bookmark1775)

chứa mối quan hệ[412](#_bookmark1888)

quy ước trên mô hình cấu hình[488](#_bookmark2231) Vai trò của nút điều phối[84](#_bookmark363)

nút điều phối[283](#_bookmark1291)

vai trò điều phối viên[86,](#_bookmark374)[283](#_bookmark1292)

thuộc tính copy\_to[499–500](#_bookmark2302)

copy\_to xây dựng[501](#_bookmark2306)

chức năng copy\_to[501](#_bookmark2305) các kiểu dữ liệu cốt lõi[118](#_bookmark531) đếm tài liệu[31](#_bookmark122)

chỉ số covid[53](#_bookmark252)

tạo hành động[188](#_bookmark869) tạo chỉ mục API[110](#_bookmark475) tạo hoạt động[188](#_bookmark870)

phương thức create()[157,](#_bookmark695)[535](#_bookmark2435)[–536](#_bookmark2440)

Trạng thái ĐÃ TẠO [317](#_bookmark1473)

thuộc tính created\_at[213–2](#_bookmark982)[14](#_bookmark984)

thuộc tính created\_by[213–214,](#_bookmark985)[216](#_bookmark990)

TạoIndexRequest[535](#_bookmark2436)

Đối tượng CreateIndexResponse[535](#_bookmark2439) truy vấn nhiều trận đấu cross\_fields[349](#_bookmark1631) CRUD (tạo, đọc, cập nhật và xóa)

hoạt động[151](#_bookmark668)

biến ctx[173](#_bookmark781) Số lượng tích lũy

(số lượng tích lũy) [461](#_bookmark2113) Tổng tích lũy (cumulative\_sum)[461](#_bookmark2114) tổng hợp tích lũy[462](#_bookmark2131) tổng hợp cha tích lũy[4](#_bookmark2133)62–464

cURL (tiện ích gọi URL dòng lệnh)[518](#_bookmark2365) yêu cầu hàng loạt sử dụng[189–1](#_bookmark876)[90](#_bookmark878)

Tổng quan[24–26](#_bookmark100)

Truy vấn DSL cho[295–296](#_bookmark1370)

**D**

tham số -d[296](#_bookmark1372) Vai trò của nút dữ liệu[84](#_bookmark359) vai trò dữ liệu[85,](#_bookmark370)[524](#_bookmark2389)

luồng dữ liệu[72](#_bookmark324)

tìm kiếm trên khắp[310–312](#_bookmark1447)

dữ liệu chuỗi thời gian[73–74](#_bookmark326)

các kiểu dữ liệu[115–](#_bookmark506)[117](#_bookmark529)

trình độ cao[131–146](#_bookmark596)

kiểu dữ liệu phẳng[140–14](#_bookmark632)[2](#_bookmark633) Kiểu dữ liệu geo\_point[131–1](#_bookmark598)[33](#_bookmark603) tham gia kiểu dữ liệu[1](#_bookmark635)42–144kiểu dữ liệu lồng nhau[138–139](#_bookmark625) Kiểu dữ liệu đối tượng[134–137](#_bookmark605)

kiểu dữ liệu search\_as\_you\_type[144–14](#_bookmark645)[6](#_bookmark650) Kiểu dữ liệu boolean[127–12](#_bookmark566)[8](#_bookmark568)

kiểu dữ liệu ngày [1](#_bookmark560)24–126các trường có nhiều[146–147](#_bookmark653)

Kiểu dữ liệu địa chỉ IP (ip)[13](#_bookmark587)0–131từ khóa[122–12](#_bookmark550)[4](#_bookmark558)

loại từ khóa hằng số[123–](#_bookmark554)[124](#_bookmark556) sửa chữa sắp xếp sử dụng[108–109](#_bookmark466) loại từ khóa[12](#_bookmark551)2–123

kiểu dữ liệu đại diện[124](#_bookmark557) cơ chế suy luận[10](#_bookmark455)5–107số[127](#_bookmark563)

phạm vi

ví dụ về kiểu date\_range[12](#_bookmark577)8–130Tổng quan[128–13](#_bookmark570)[0](#_bookmark581)

kiểu dữ liệu văn bản[11](#_bookmark535)8–122phân tích các trường văn bản[118–12](#_bookmark538)[0](#_bookmark541)

kiểu dữ liệu token\_count[120–122](#_bookmark545) loại ép buộc[11](#_bookmark502)4–115

vai trò data\_cold[85](#_bookmark365)

vai trò data\_frozen[85](#_bookmark366)

vai trò data\_hot[85](#_bookmark367)

vai trò data\_warm[85](#_bookmark368) cơ sở dữ liệu

phép loại suy cơ sở dữ liệu quan hệ[68–](#_bookmark307)[69](#_bookmark309) công cụ tìm kiếm so với6–7

tìm kiếm được hỗ trợ bởi5–6

kiểu dữ liệu ngày[46,](#_bookmark201)[124–126,](#_bookmark561)[194,](#_bookmark891)[421](#_bookmark1940) định dạng ngày tháng, dẫn đến sai[109](#_bookmark469) biểu đồ ngày

thiết lập khoảng thời gian cho[448–450](#_bookmark2057) Tổng quan[447–44](#_bookmark2053)[8](#_bookmark2056)

loại ngày[103–10](#_bookmark440)[6,](#_bookmark456)[109,](#_bookmark470)[112](#_bookmark489)

tổng hợp date\_histogram[447–448,](#_bookmark2055)[463](#_bookmark2134)

thùng date\_histogram[456](#_bookmark2091) kiểu dữ liệu date\_range[128](#_bookmark573) loại phạm vi ngày tháng[115](#_bookmark513) loại phạm vi ngày[128–130](#_bookmark577)

Mức độ GỠ LỖI[48](#_bookmark2241)9–490

tham số default\_field[355](#_bookmark1660)

Thuộc tính default\_operator[293](#_bookmark1351)

tham số default\_operator[293,](#_bookmark1349)[355](#_bookmark1667) XÓA \_tất cả tất cả điểm cuối [209](#_bookmark955) XÓA hành động[209](#_bookmark951)

xóa hành động[189](#_bookmark873)

XÓA API[210](#_bookmark957)

Phương pháp XÓA[29](#_bookmark107)

xóa hoạt động[40](#_bookmark163)

xóa tài liệu[181–18](#_bookmark832)[5](#_bookmark852) xóa tất cả các tài liệu[184](#_bookmark850)[–185](#_bookmark853) xóa theo truy vấn[182](#_bookmark841)

xóa bằng ID [181–182](#_bookmark834) xóa với truy vấn phạm vi[1](#_bookmark847)83–184

phần chi tiết[303](#_bookmark1408)

giám đốc tài sản[319](#_bookmark1490)

truy vấn nhiều trận đấu dis\_max[350](#_bookmark1639) truy vấn dis\_max [350,](#_bookmark1638)[388–38](#_bookmark1786)[9](#_bookmark1791) phục hồi thảm họa (DR)[97](#_bookmark423)

**546**

mối quan hệ rời rạc[412](#_bookmark1889)

Truy vấn hợp chất disjunction max (dis\_max)[365](#_bookmark1708)

ngưỡng sử dụng đĩa[509](#_bookmark2338)[–511](#_bookmark2343)

đĩa-giai-đoạn-lũ-lụthình mờ[510–511](#_bookmark2341)

hình mờ đĩa cao[510](#_bookmark2340)

hình mờ đĩa thấp[509](#_bookmark2339)

thuộc tính khoảng cách[410](#_bookmark1881)

truy vấn distance\_feature[42](#_bookmark1939)1–426tăng điểm bằng cách sử dụng ngày tháng[424–426](#_bookmark1943)

tăng điểm bằng cách sử dụng vị trí địa lý[42](#_bookmark1941)2–424đối tượng doc[171](#_bookmark771)[–172,](#_bookmark774)[179–1](#_bookmark821)[80,](#_bookmark823)[189](#_bookmark872)

cờ doc\_as\_upsert[179](#_bookmark822) Người lái tàu

cài đặt Elasticsearch[519](#_bookmark2368)

cài đặt Kibana[522](#_bookmark2379) lệnh chạy docker[519](#_bookmark2369) chìa khóa tài liệu[164](#_bookmark730)

tài liệu API

API \_bí danh[20](#_bookmark933)6–207

\_API số lượng lớn[185–](#_bookmark856)[190](#_bookmark879)

lập chỉ mục tài liệu[151–158](#_bookmark667)

\_API số lượng lớn[39–4](#_bookmark161)[1](#_bookmark166)

\_tạo API[15](#_bookmark693)6–158

ID tài liệu[151–](#_bookmark669)[156](#_bookmark692)

Tổng quan[23–24](#_bookmark97)

API tài liệu đơn[162–](#_bookmark716)[163](#_bookmark724)

tần suất tài liệu[92](#_bookmark401) ID tài liệu (mã định danh)

xóa tài liệu với[1](#_bookmark834)81–182lập chỉ mục tài liệu với[151–](#_bookmark676)[154](#_bookmark682) lập chỉ mục tài liệu mà không có[154–156](#_bookmark683) Tổng quan[151](#_bookmark669)

định danh do hệ thống tạo ra[503](#_bookmark2315)

phương pháp tài liệu[537–53](#_bookmark2446)[8](#_bookmark2449) API hoạt động tài liệu[69](#_bookmark311) tài liệu[67](#_bookmark303)[–69,](#_bookmark312)[149–19](#_bookmark662)[1](#_bookmark883)

hoạt động hàng loạt [185–190](#_bookmark856)

lập chỉ mục hàng loạt tài liệu[186–188](#_bookmark865) yêu cầu số lượng lớnsử dụng cURL[189–190](#_bookmark876) định dạng của \_bulk API[185–1](#_bookmark861)[86](#_bookmark863) độc lậpthực thể và nhiều

hành động[188–18](#_bookmark868)[9](#_bookmark874)

đếm[31](#_bookmark122)

xóa bỏ[181–1](#_bookmark832)[85](#_bookmark852)

xóa tất cả các tài liệu[18](#_bookmark850)4–185

xóa theo truy vấn (\_delete\_by\_query)[182](#_bookmark841) xóa bằng ID[181–182](#_bookmark834)

xóa với truy vấn phạm vi[183–184](#_bookmark847) API hoạt động tài liệu [69](#_bookmark311) lập chỉ mục[23–2](#_bookmark96)[6,](#_bookmark102)[150–161](#_bookmark665)

khách hàng[537–53](#_bookmark2443)[8](#_bookmark2451)

thành phần của yêu cầu[29](#_bookmark106)–30cURL[24–26](#_bookmark100)

tùy chỉnh quá trình làm mới[160–161](#_bookmark703) tài liệu API[23–24,](#_bookmark99)[15](#_bookmark667)1–158

**MỤC LỤC**

với số lượng lớn[186–188](#_bookmark865)

lập bản đồ[103–10](#_bookmark439)[4](#_bookmark448)

cơ học của[15](#_bookmark700)8–160

quá trình cho[26–3](#_bookmark104)[0](#_bookmark115)

lập chỉ mục lại[190](#_bookmark881)

sử dụng \_bulk API[39–](#_bookmark161)[41](#_bookmark166) thao túng phản ứng[166](#_bookmark742)[–170](#_bookmark757)

bao gồm và loại trừ các trường[167–170](#_bookmark752) xóa phản hồi siêu dữ liệu [166](#_bookmark744) ngăn chặn các tài liệu nguồn[167](#_bookmark749)

phân tích dữ liệu[67–68](#_bookmark304)

phép loại suy cơ sở dữ liệu quan hệ[68–](#_bookmark307)[69](#_bookmark309) đang lấy lại[31–36](#_bookmark126)[,](#_bookmark143)[161–166](#_bookmark714)

truy vấn id[165–16](#_bookmark736)[6](#_bookmark740)

đang lấy lại tất cả các tài liệu[34](#_bookmark139)–36lấy lại nhiều tài liệu[163–165](#_bookmark726)

lấy nhiều tài liệu theo ID[33](#_bookmark133)–34lấy lại các tài liệu đơn lẻ[32–33](#_bookmark129)

API tài liệu đơn[162–1](#_bookmark716)[63](#_bookmark724) ngăn chặn từ kết quả[307](#_bookmark1429) ngăn chặn nguồn từ các phản hồi[167](#_bookmark749) đang cập nhật[170–18](#_bookmark760)[1](#_bookmark830)

\_cập nhật API[171](#_bookmark768)[–173](#_bookmark777)

cơ chế cập nhật tài liệu[170–171](#_bookmark762) thay thế tài liệu[177–1](#_bookmark810)[78](#_bookmark814)

cập nhật theo kịch bản[173–177](#_bookmark779)

hoạt động update\_by\_query[180–181](#_bookmark826)

upserts[178–180](#_bookmark816)

kiểu dữ liệu kép[115,](#_bookmark508)[431](#_bookmark1961)

phạm vi đôi [130](#_bookmark582) DR (phục hồi thảm họa)[97](#_bookmark423) ánh xạ động[104–109](#_bookmark451)

những hạn chế của[10](#_bookmark461)7–109

suy ra định dạng ngày tháng không chính xác [109](#_bookmark469) suy ra các ánh xạ không chính xác [107–108](#_bookmark462) sửa lỗi sắp xếp bằng cách sử dụng các loại từ khóa[108–](#_bookmark466)[109](#_bookmark467)

cơ chế để suy ra các loại[105–10](#_bookmark455)[7](#_bookmark459) thiết lập động[197](#_bookmark904)

**VÀ**

trình phân tích edge\_ngram[273](#_bookmark1249)

chỉnh sửa khoảng cách[331,](#_bookmark1540)[357](#_bookmark1677)

*Java hiệu quả*(Bloch)[23,](#_bookmark92)[38,](#_bookmark158)[49–50,](#_bookmark230)[52,](#_bookmark239)[356](#_bookmark1670) Các lĩnh vực ứng dụng lõi đàn hồi[10–11](#_bookmark44)

Tìm kiếm doanh nghiệp đàn hồi[10](#_bookmark45) khả năng quan sát[11](#_bookmark46)

bảo vệ[11](#_bookmark47)

Ngăn xếp đàn hồi[11–](#_bookmark51)[12](#_bookmark54)

Nhịp đập[12](#_bookmark52)

Kibana[12](#_bookmark55)

Đăng nhập[12](#_bookmark53)

Tìm kiếm đàn hồi[20](#_bookmark84)–99

sự quản lý[4](#_bookmark2143)67–496

cấu hình nâng cao[48](#_bookmark2230)8–491

nhóm chủ[4](#_bookmark2261)[91–495](#_bookmark2286)

**MỤC LỤC 547**

giao tiếp nút[474](#_bookmark2166)[–475](#_bookmark2171)

cụm mở rộng[468–474](#_bookmark2145)

kích thước mảnh vỡ[47](#_bookmark2173)5–477

ảnh chụp nhanh[477–](#_bookmark2184)[487](#_bookmark2226)

tổng hợp[5](#_bookmark244)[2–58,](#_bookmark272)[434–4](#_bookmark1971)[66](#_bookmark2141)

các thành phần của[67–86](#_bookmark301)

truy vấn hợp chất[47–52,](#_bookmark241)[361–396](#_bookmark1686)

sự phát triển của[8](#_bookmark32)

tài liệu[149](#_bookmark662)[–191](#_bookmark883)

tìm kiếm toàn văn[36–45](#_bookmark147)

kiến trúc cấp cao[61](#_bookmark277)–67

chỉ số[192–2](#_bookmark885)[34](#_bookmark1061)

cài đặt[515–520](#_bookmark2357)

tải xuống nhị phân[515–5](#_bookmark2358)[16](#_bookmark2359) cài đặt qua Docker [519](#_bookmark2368) khởi động trên macOS[518](#_bookmark2364)[–519](#_bookmark2367) khởi động trên Windows[51](#_bookmark2360)6–518

máy chủ thử nghiệm với API \_cat[519–520](#_bookmark2370) lập bản đồ[100–148](#_bookmark430)

những quan niệm sai lầm liên quan đến[15–16](#_bookmark73)

Tổng quan9–10

hiệu suất và xử lý sự cố[49](#_bookmark2290)7–514cầu dao điện[511–512](#_bookmark2346)

vấn đề về tốc độ chỉ số[503](#_bookmark2313)[–506](#_bookmark2325) vấn đề tìm kiếm và tốc độ[498–502](#_bookmark2292) cụm không ổn định[506–511](#_bookmark2328)

sự áp dụng phổ biến của[16](#_bookmark76) mồi[21–30](#_bookmark86)

lập chỉ mục tài liệu [23–2](#_bookmark96)[6](#_bookmark102) ví dụ về hiệu sách trực tuyến[21–23](#_bookmark88)

sự liên quan[89–9](#_bookmark384)[0](#_bookmark389)

đang lấy dữ liệu[30–36](#_bookmark119)

thuật toán định tuyến[96–97](#_bookmark414)

tỉ lệ[97](#_bookmark420)

tìm kiếm[280,](#_bookmark1279)[312,](#_bookmark1452)[397–433](#_bookmark1823)

giải phẫu của các yêu cầu[288–2](#_bookmark1317)[90](#_bookmark1323) giải phẫu của phản ứng[290](#_bookmark1325)[–291](#_bookmark1336) đặc điểm của[297](#_bookmark1382)[–312](#_bookmark1451)

tìm kiếm toàn văn[334–360](#_bookmark1547)

chức năng của [282–283](#_bookmark1290)

những điều cơ bản của [285–288](#_bookmark1300)

tìm kiếm vị trí[398–401](#_bookmark1830)

Tổng quan[281–2](#_bookmark1281)[82](#_bookmark1288)

Truy vấn DSL[294](#_bookmark1363)[–297](#_bookmark1380)

dữ liệu mẫu[28](#_bookmark1296)3–285

truy vấn hình dạng[41](#_bookmark1894)2–414

truy vấn khoảng[4](#_bookmark1904)14–421

truy vấn chuyên biệt[421–43](#_bookmark1937)[2](#_bookmark1967)

tìm kiếm theo cấp độ thuật ngữ [313–333](#_bookmark1454) Tìm kiếm yêu cầu URI[291–](#_bookmark1338)[294](#_bookmark1361)

truy vấn cấp độ thuật ngữ[45–47](#_bookmark199)

phân tích văn bản[23](#_bookmark1063)5–279

trường hợp sử dụng[12–1](#_bookmark57)[3](#_bookmark66)

phân tích kinh doanh[13](#_bookmark63)

ghi nhật ký và giám sát ứng dụng[13](#_bookmark65) công cụ tìm kiếm[12–](#_bookmark58)[13](#_bookmark61)

phân tích bảo mật và phát hiện mối đe dọa và gian lận[13](#_bookmark64)

sử dụng không phù hợp[1](#_bookmark68)3–14

Lớp ElasticsearchClient[538](#_bookmark2455) Máy khách Java API ElasticsearchClient[534](#_bookmark2432) Các lớp liên quan đến ElasticsearchClient[531](#_bookmark2415)

phương thức elasticsearchClient.index() [538](#_bookmark2452)

ElasticsearchFEATUREClient khách hàng [533](#_bookmark2428)

Khách hàng ElasticsearchIndicesClient[533–5](#_bookmark2429)[34,](#_bookmark2433)[536](#_bookmark2441)

Đối tượng ElasticsearchTransport[533](#_bookmark2422)

trong thẻ[44](#_bookmark186)

tài sản email[122](#_bookmark552)

chỉ mục email [135](#_bookmark615)

thuộc tính kết thúc [417](#_bookmark1914)

máy phân tích tiếng anh[242,](#_bookmark1102)[256–2](#_bookmark1166)[58](#_bookmark1170)

Tìm kiếm doanh nghiệp[13](#_bookmark60)

erm truy vấn[391](#_bookmark1798)

kho lưu trữ es\_cluster\_snapshot\_repository[479,](#_bookmark2193)

[481](#_bookmark2202)

Biến ES\_PATH\_CONF[475,](#_bookmark2168)[488](#_bookmark2232)

mảng escaped\_tags[266](#_bookmark1218)

tùy chọn escaped\_tags[267](#_bookmark1219)

Công cụ ETL (trích xuất, chuyển đổi, tải)[62,](#_bookmark280)[185,](#_bookmark854)[523](#_bookmark2383)

Evan, Ben[490](#_bookmark2251)

loại trừ mảng[310](#_bookmark1442)

tồn tại truy vấn[47,](#_bookmark210)[315,](#_bookmark1463)[322–32](#_bookmark1502)[3](#_bookmark1504)

truy vấn tốn kém[327](#_bookmark1518)

thuộc tính expire\_after[485](#_bookmark2219)

giải thích API[303](#_bookmark1410)[–304](#_bookmark1417)

giải thích thuộc tính[301–303](#_bookmark1404)

giải thích cờ[301](#_bookmark1403)[–303](#_bookmark1409)

giải thích tham số[293](#_bookmark1354)

tạo chỉ mục rõ ràng[194,](#_bookmark890)[198](#_bookmark909)

lập bản đồ rõ ràng [109–115](#_bookmark474) sửa đổi các trường hiện có[11](#_bookmark496)3–114loại ép buộc[11](#_bookmark502)4–115

sử dụng API lập chỉ mục[110–112](#_bookmark478) sử dụng API lập bản đồ

cập nhật các chỉ mục trống[113](#_bookmark492) cập nhật lược đồ[112–113](#_bookmark485)

Thống kê mở rộng (extended\_stats)[461](#_bookmark2124) tổng hợp extended\_stats [438](#_bookmark2007)

hàm extended\_stats[443](#_bookmark2040)

số liệu thống kê mở rộng[443](#_bookmark2036)

**F**

thuộc tính yếu tố[393](#_bookmark1805)

thuộc tính không thành công [291](#_bookmark1329)

cờ thất bại[291](#_bookmark1330)

kết quả âm tính giả[336](#_bookmark1558)

dương tính giả[336](#_bookmark1553) bộ nhớ đệm dữ liệu trường[108](#_bookmark465)

Bộ ngắt mạch dữ liệu hiện trường[512](#_bookmark2350) hàm field\_value\_factor[393–](#_bookmark1804)[394](#_bookmark1806)

chuẩn độ dài trường[93](#_bookmark404)

**548 MỤC LỤC**

các máy phân tích cấp trường chỉ định cho các chỉ mục[262](#_bookmark1193) chỉ định cho tìm kiếm[264](#_bookmark1200)

mảng trường [358](#_bookmark1682)

đối tượng trường[147,](#_bookmark654)[308](#_bookmark1431)

tham số trường[355,](#_bookmark1661)[428](#_bookmark1953)

thuộc tính tên tập tin[134](#_bookmark607)

thuộc tính kiểu tệp[134](#_bookmark608)

thuộc tính lọc[258](#_bookmark1177)

điều khoản lọc[51](#_bookmark238)[–52,](#_bookmark242)[378–38](#_bookmark1752)[1](#_bookmark1756)

lọc ngữ cảnh[52,](#_bookmark240)[287–288](#_bookmark1310)

truy vấn lọc[286,](#_bookmark1307)[378,](#_bookmark1753)[380,](#_bookmark1754)[384–](#_bookmark1769)[385](#_bookmark1772)

máy phân tích dấu vân tay[253–2](#_bookmark1149)[54](#_bookmark1154) bộ lọc mã thông báo dấu vân tay[253](#_bookmark1150) chế độ đầu tiên[395](#_bookmark1812)

khoảng thời gian cố định[449](#_bookmark2061)

khoảng thời gian cố định[448–](#_bookmark2060)[450](#_bookmark2064) kiểu dữ liệu phẳng[140–14](#_bookmark632)[2](#_bookmark633) kiểu dữ liệu float[46,](#_bookmark202)[105,](#_bookmark453)[114](#_bookmark500) Kiểu dữ liệu float\_range[128](#_bookmark574) float\_range phạm vi [130](#_bookmark583) float, kiểu kép[116](#_bookmark518)

Kiểu số nguyên[127](#_bookmark564) hình mờ đĩa giai đoạn lũ[510–511](#_bookmark2341)

thuộc tính hình thức[289](#_bookmark1320)

tìm thấy thuộc tính[162](#_bookmark719)

tìm thấy\* thẻ[347](#_bookmark1621)

phát hiện gian lận[13](#_bookmark64)

từ thuộc tính[293,](#_bookmark1355)[299,](#_bookmark1389)[452](#_bookmark2073)

từ tham số[299](#_bookmark1390)

trạng thái ĐÃ HOÀN THÀNH [317](#_bookmark1474)

kiểu dữ liệu toàn văn[118](#_bookmark536)

tìm kiếm toàn văn[36–45,](#_bookmark195)[334](#_bookmark1547)[–360](#_bookmark1684)

thúc đẩy kết quả [42](#_bookmark177)

truy vấn mơ hồ[357](#_bookmark1676)

lập chỉ mục tài liệu bằng cách sử dụng \_bulk API[39–41](#_bookmark161) truy vấn cụm từ khớp

Tổng quan[42–43](#_bookmark179)

cụm từ có từ bị thiếu[44](#_bookmark188) truy vấn phù hợp[340–](#_bookmark1580)[345](#_bookmark1606)

phân tích [342](#_bookmark1588) sách của tác giả[36](#_bookmark149)

sửa lỗi đánh máy bằng cách sử dụng từ khóa fuzziness[3](#_bookmark1603)[44–345](#_bookmark1607) định dạng của[340–341](#_bookmark1582)

phù hợp với ít nhất một vài từ[343–34](#_bookmark1597)[4](#_bookmark1600) tìm kiếm nhiều từ[342–343](#_bookmark1592) tìm kiếm bằng cách sử dụng[341](#_bookmark1586)

với toán tử AND[37–39](#_bookmark154) truy vấn match\_all[338–339](#_bookmark1566)

xây dựng[338–3](#_bookmark1568)[39](#_bookmark1571) dạng viết tắt của[339](#_bookmark1573)

truy vấn match\_none [339–3](#_bookmark1576)[40](#_bookmark1578)

truy vấn match\_phrase[345–346](#_bookmark1609)

truy vấn match\_phrase\_prefix[346–347](#_bookmark1617)

truy vấn multi\_match[347–3](#_bookmark1625)[52](#_bookmark1647)

những cánh đồng tốt nhất[348–](#_bookmark1627)[349](#_bookmark1636)

thúc đẩy các lĩnh vực riêng lẻ[35](#_bookmark1646)1–352truy vấn dis\_max[350](#_bookmark1638)

người phá vỡ thế giằng co[350–351](#_bookmark1641)

Tổng quan[335–33](#_bookmark1550)[8](#_bookmark1560)

độ chính xác[335–3](#_bookmark1552)[36](#_bookmark1555)

chuỗi truy vấn truy vấn[353–356](#_bookmark1654)

toán tử mặc định[355](#_bookmark1664)

các lĩnh vực trong [354–355](#_bookmark1659)

với các cụm từ[356](#_bookmark1669)

nhớ lại[336–338](#_bookmark1557)

dữ liệu mẫu[338](#_bookmark1563)

tìm kiếm trên nhiều trường[41–42](#_bookmark168) truy vấn simple\_query\_string[357–](#_bookmark1679)[359](#_bookmark1683)

lỗi chính tả[44–45](#_bookmark192)

Điểm chức năng(function\_score) truy vấn[365](#_bookmark1706) truy vấn function\_score[389–3](#_bookmark1793)[95](#_bookmark1819)

kết hợp điểm số chức năng[394–](#_bookmark1808)[395](#_bookmark1820) hàm field\_value\_factor[393–](#_bookmark1804)[394](#_bookmark1806)

hàm random\_score[391](#_bookmark1797)[–392](#_bookmark1799)

hàm script\_score[392–3](#_bookmark1801)[93](#_bookmark1802)

thuộc tính mờ[332](#_bookmark1543)

từ khóa mờ nhạt[344–345](#_bookmark1603)

tham số mờ[344](#_bookmark1605)

nguyên lý mờ[331](#_bookmark1539)

thiết lập độ mờ[44](#_bookmark194)

truy vấn mơ hồ[33](#_bookmark1538)[0–332,](#_bookmark1544)[357](#_bookmark1676)

**G**

AI tạo sinh (trí tuệ nhân tạo)[1](#_bookmark79)6–19bộ biến đổi được đào tạo trước tạo ra (GPT)[17](#_bookmark80) truy vấn geo\_bounding\_box[132,](#_bookmark601)[405–40](#_bookmark1866)[9](#_bookmark1878)

truy vấn geo\_distance[399,](#_bookmark1836)[40](#_bookmark1880)9–410

Kiểu dữ liệu geo\_point[131–13](#_bookmark598)[3,](#_bookmark603)[401–402](#_bookmark1845) Kiểu dữ liệu geo\_shape[402–40](#_bookmark1852)[4](#_bookmark1860) truy vấn geo\_shape[400–401,](#_bookmark1840)[410–](#_bookmark1884)[412](#_bookmark1891)

loại hình dạng địa lý[115,](#_bookmark514)[402](#_bookmark1850)

hình chữ nhật địa lý[399](#_bookmark1833)

các loại dữ liệu không gian địa lý[401–4](#_bookmark1842)[04](#_bookmark1861) kiểu dữ liệu geo\_point[401–4](#_bookmark1845)[02](#_bookmark1849) kiểu dữ liệu geo\_shape[402–4](#_bookmark1852)[04](#_bookmark1860)

truy vấn không gian địa lý[404–412](#_bookmark1863)

truy vấn geo\_bounding\_box[4](#_bookmark1866)05–409

truy vấn geo\_distance[409–41](#_bookmark1880)[0](#_bookmark1882)

truy vấn geo\_shape[410–412](#_bookmark1884)

bộ lọc german\_normalizer[104](#_bookmark445) Lệnh GET \_all [208](#_bookmark947)

Lệnh GET \_cat/nodes[84](#_bookmark355)

NHẬN điểm cuối \_cluster/health[76,](#_bookmark333)[78](#_bookmark335) GET \_count gọi [31](#_bookmark124)

Lệnh GET \_nodes/thread\_pool[505](#_bookmark2324) NHẬN \_phân đoạn[219](#_bookmark1009)

NHẬN \_ảnh chụp nhanhyêu cầu[479](#_bookmark2195) NHẬN API[161](#_bookmark712)

lệnh GET [32,](#_bookmark130)[103,](#_bookmark442)[20](#_bookmark941)[7–208,](#_bookmark942)[245](#_bookmark1119)

GET lệnh email/\_mapping [134](#_bookmark612)

**MỤC LỤC 549**

Phương thức GET[29,](#_bookmark108)[162,](#_bookmark717)[171,](#_bookmark764)[289](#_bookmark1318)

NHẬN yêu cầu[163,](#_bookmark721)[200,](#_bookmark916)[204](#_bookmark930)

mẫu glob (lệnh toàn cục)[212,](#_bookmark976)[482](#_bookmark2203)

Gough, Jame[490](#_bookmark2252)

GPT (bộ biến đổi được đào tạo trước tạo ra)[17](#_bookmark80) toán tử gt (lớn hơn)[128,](#_bookmark571)[323](#_bookmark1507)

toán tử gte (lớn hơn hoặc bằng) [128,](#_bookmark572)[323](#_bookmark1508)

**H**

hàm băm[96](#_bookmark415)

HDD (ổ đĩa cứng)[498](#_bookmark2295)

HDFS (Hệ thống tập tin phân tán Hadoop)[14,](#_bookmark70)[478](#_bookmark2187) Lệnh HEAD <index\_name>[208](#_bookmark944)

Yêu cầu HEAD[163](#_bookmark722)

đống[498](#_bookmark2297)

chỉ số ẩn[208](#_bookmark946)

hình mờ đĩa cao[510](#_bookmark2340)

kiến trúc cấp cao[61](#_bookmark277)–67

dữ liệu trong[62–63](#_bookmark279)

dữ liệu ra[66–67](#_bookmark297)

xử lý dữ liệu[63–66](#_bookmark283)

phân tích dữ liệu[64–](#_bookmark290)[66](#_bookmark295)

thu thập dữ liệu[63](#_bookmark284) các loại dữ liệu[64](#_bookmark288)

làm nổi bật đối tượng[43,](#_bookmark182)[300](#_bookmark1397)

làm nổi bật[43,](#_bookmark183)[299–301](#_bookmark1396)

tổng hợp biểu đồ[446,](#_bookmark2052)[452](#_bookmark2072)

thùng biểu đồ histogram[56–57,](#_bookmark268)[446](#_bookmark2051)

biểu đồ histogram[446](#_bookmark2050)[–450](#_bookmark2062)

biểu đồ ngày[447–448](#_bookmark2053)

thiết lập khoảng thời gian cho biểu đồ ngày[448](#_bookmark2057)[–450](#_bookmark2063) mảng hit[539](#_bookmark2462)

thuộc tính lượt truy cập[291](#_bookmark1331)

bộ lọc hmtl\_strip[266–267](#_bookmark1215)

tỷ lệ ngang[98,](#_bookmark426)[477](#_bookmark2180)

chính sách hot\_delete\_policy[230](#_bookmark1049)

hot\_delete\_policy\_index chỉ mục[230](#_bookmark1047)

bộ lọc ký tự html\_strip[240,](#_bookmark1089)[258,](#_bookmark1173)[266](#_bookmark1217)

bộ lọc html\_strip[260,](#_bookmark1179)[266](#_bookmark1211) Hành động XÓA HTTP[483](#_bookmark2213) Phương pháp HTTP DELETE[181](#_bookmark835) Hành động HTTP HEAD [163](#_bookmark720) Hành động HTTP POST [156](#_bookmark691) BÀI ĐĂNG HTTPphương pháp[151,](#_bookmark675)[154](#_bookmark684)

Phương pháp HTTP PUT[23,](#_bookmark98)[29,](#_bookmark112)[151,](#_bookmark674)[154](#_bookmark685)

**TÔI**

Công cụ IaC (cơ sở hạ tầng dưới dạng mã)[98](#_bookmark427) ID (mã định danh)[151](#_bookmark670)

phương pháp id[537](#_bookmark2447)

IDF (tần suất tài liệu nghịch đảo)[90,](#_bookmark394)[92,](#_bookmark399)[302](#_bookmark1406) truy vấn API ids[32](#_bookmark128)

đối tượng id[33](#_bookmark136)

truy vấn id[33,](#_bookmark134)[165](#_bookmark736)[–166,](#_bookmark740)[321–322](#_bookmark1498)

Truy vấn cấp độ thuật ngữ ID[47,](#_bookmark209)[315](#_bookmark1461)

ID, lấy nhiều tài liệu bằng[33–](#_bookmark133)[34](#_bookmark138) ILM (quản lý vòng đời chỉ mục)[227](#_bookmark1038)[–234](#_bookmark1059)

vòng đời[228–22](#_bookmark1041)[9](#_bookmark1042)

vòng đời với rollover[231–2](#_bookmark1052)[34](#_bookmark1060) quản lý thủ công[229–23](#_bookmark1044)[0](#_bookmark1050)

ngụ ý (tự động)tạo chỉ mục[194](#_bookmark894)[–198](#_bookmark906) cấu hình[1](#_bookmark903)97–198

vô hiệu hóa tự động tạo[196](#_bookmark898)[–197](#_bookmark902)

cờ in\_order[418](#_bookmark1920)

thuộc tính include\_global\_state[485](#_bookmark2218)

bao gồm mảng[310](#_bookmark1443)

hành động chỉ số[53,](#_bookmark253)[185](#_bookmark857)

phương pháp chỉ số[537](#_bookmark2448)

đối tượng chỉ mục[201](#_bookmark917)

hoạt động chỉ số[40](#_bookmark164)

gói chỉ mục[489](#_bookmark2240)

đối tượng index\_prefixes[330](#_bookmark1534)

tham số index\_prefixes[329](#_bookmark1532) máy phân tích cấp chỉ mục

chỉ định cho các chỉ mục[262–263](#_bookmark1194) chỉ định cho tìm kiếm[264–26](#_bookmark1202)[5](#_bookmark1203)

Thuộc tính index.blocks.write [223](#_bookmark1020)

index.lifecycle.name[230](#_bookmark1048)

index.lifecycle.rollover\_alias tên[232](#_bookmark1054) thiết lập thuộc tính index.max\_terms\_count[319](#_bookmark1485) Thuộc tính index.number\_of\_replicas[223](#_bookmark1021)

chỉ mục.routing.allocation.require.property [224](#_bookmark1023)

phương thức index()[535](#_bookmark2438)

cặp index/\_id[165](#_bookmark734)

chỉ số[192–234](#_bookmark885)

khách hàng[534–538](#_bookmark2430)

đóng cửa[210–212](#_bookmark962)

tránh làm mất ổn định hệ thống[211](#_bookmark966) đóng tất cả hoặc nhiều chỉ mục[211](#_bookmark964)

tạo ra[193–20](#_bookmark889)[7](#_bookmark936)

bí danh[202–207](#_bookmark926)

cài đặt tùy chỉnh[198–](#_bookmark911)[201](#_bookmark918)

rõ ràng[198](#_bookmark909)

ngầm định (tự động tạo)[194–19](#_bookmark894)[8](#_bookmark906) ánh xạ[201–2](#_bookmark920)[02](#_bookmark924)

xóa bỏ[209–21](#_bookmark950)[0](#_bookmark960)

xóa nhiều chỉ mục[2](#_bookmark952)09–210chỉ xóa các bí danh[210](#_bookmark958)

vấn đề về tốc độ chỉ số[503–506](#_bookmark2313) điều chỉnh tốc độ làm mới[503–5](#_bookmark2320)[06](#_bookmark2326) yêu cầu số lượng lớn[503](#_bookmark2317)

định danh do hệ thống tạo ra[503](#_bookmark2315)

lập chỉ mục tài liệu[23–26](#_bookmark96)[,](#_bookmark102)[150–161](#_bookmark665) thành phần của yêu cầu[29–30](#_bookmark106) cURL[24–26](#_bookmark100)

tùy chỉnh quá trình làm mới[160–161](#_bookmark703) tài liệu API[23–](#_bookmark97)[24,](#_bookmark99)[151–158](#_bookmark667)

với số lượng lớn[186–188](#_bookmark865)

lập bản đồ[103–10](#_bookmark439)[4](#_bookmark448)

**550**

chỉ số*(tiếp theo)*

cơ học của[1](#_bookmark700)58–160

quá trình cho[26–](#_bookmark104)[30](#_bookmark115)

lập chỉ mục lại[190](#_bookmark881)

sử dụng \_bulk API[39–41](#_bookmark161) đảo ngược[86–89](#_bookmark379)

quản lý vòng đời[227](#_bookmark1038)[–234](#_bookmark1059)

vòng đời[228–2](#_bookmark1041)[29](#_bookmark1042)

vòng đời với rollover[231](#_bookmark1052)[–234](#_bookmark1060) quản lý thủ công[229–2](#_bookmark1044)[30](#_bookmark1050)

giám sát và quản lý[216–220](#_bookmark995) thống kê chỉ số[217–218](#_bookmark997)

nhiều chỉ số và số liệu thống kê[218–22](#_bookmark1008)[0](#_bookmark1010) mở đầu[211–2](#_bookmark969)[12](#_bookmark973)

Tổng quan[71–72](#_bookmark318)[,](#_bookmark322)[193](#_bookmark887)

đọc[207–2](#_bookmark938)[08](#_bookmark948)

chỉ số ẩn[208](#_bookmark946)

chỉ số công khai[207–208](#_bookmark940)

lăn qua các bí danh chỉ mục[224–](#_bookmark1026)[227](#_bookmark1035) tìm kiếm trên khắp[310–312](#_bookmark1447)

kích thước mảnh vỡ

thiết lập nhiều chỉ mục[476–477](#_bookmark2178) thiết lập chỉ mục đơn[47](#_bookmark2175)5–476

co lại[222–](#_bookmark1019)[224](#_bookmark1024)

chỉ định các máy phân tích cho[26](#_bookmark1191)1–263máy phân tích cấp trường[262](#_bookmark1193)

máy phân tích cấp chỉ mục[262–263](#_bookmark1194)

chia tách[220–2](#_bookmark1013)[22](#_bookmark1017)

mẫu cho[2](#_bookmark975)12–216

tạo mẫu thành phần[215–2](#_bookmark988)[16](#_bookmark993) tạo mẫu có thể cấu hình[213–2](#_bookmark980)[15](#_bookmark986)

cập nhật các chỉ mục trống [113](#_bookmark492)

API lập chỉ mục, ánh xạ rõ ràng[110–112](#_bookmark478) cập nhật các chỉ mục trống[113](#_bookmark492)

cập nhật lược đồ[112–113](#_bookmark485)

phương pháp indexOf[173](#_bookmark787)

Đối tượng IndexRequest[538](#_bookmark2453)

thuộc tính chỉ số[482](#_bookmark2204)

không gian tên chỉ mục[534](#_bookmark2434)

đối tượng index\_boost[311](#_bookmark1448)

hàm chỉ số()[535](#_bookmark2437)

Suy luận (suy luận)[461](#_bookmark2115)

Mạch yêu cầu trên chuyến baymáy cắt[512](#_bookmark2348) Mức độ THÔNG TIN[488](#_bookmark2238)[–489](#_bookmark2239)

công cụ cơ sở hạ tầng dưới dạng mã (IaC)[98](#_bookmark427) Vai trò nút nhập[84](#_bookmark360)

đường ống hấp thụ [523–530](#_bookmark2382)

tải PDF vào Elasticsearch[527–529](#_bookmark2403) cơ học của[5](#_bookmark2388)24–526

Tổng quan[523–5](#_bookmark2384)[24](#_bookmark2387)

nhập vai trò[85,](#_bookmark371)[524](#_bookmark2390)

thuộc tính initial\_timeout[492](#_bookmark2270) đối tượng đánh bên trong[291](#_bookmark1333) Kiểu dữ liệu integer\_range[128](#_bookmark575) số nguyên\_phạm vi[130](#_bookmark584)

giao nhau (mặc định)mối quan hệ[411](#_bookmark1886)

**MỤC LỤC**

tần suất tài liệu nghịch đảo (IDF)[90,](#_bookmark394)[92,](#_bookmark399)[302](#_bookmark1406)

chỉ số đảo ngược[86–89](#_bookmark379)

IP (internetđịa chỉ giao thức)[130](#_bookmark588) Kiểu dữ liệu IP[115,](#_bookmark515)[130–131](#_bookmark587)

phạm vi ip[115,](#_bookmark516)[128,](#_bookmark576)[130,](#_bookmark585)[452](#_bookmark2075)

bản đồ iphone[424](#_bookmark1944)

thuộc tính is\_write\_index[204](#_bookmark928)

**J**

Lớp JacksonJasonpMapper[538](#_bookmark2450)

Máy khách Java[530–5](#_bookmark2411)[31](#_bookmark2412)

*Đồng thời Java trong thực tế*(Bloch)[49–50](#_bookmark223)

*Hiệu suất Java*(Cây sồi)[490](#_bookmark2253)

Máy ảo Java (JVM)[490–](#_bookmark2249)[491](#_bookmark2258) JHLRC (Máy khách REST cấp cao Java)[531](#_bookmark2413) tham gia kiểu dữ liệu[14](#_bookmark635)2–144

tham gia truy vấn[327](#_bookmark1519)

thuộc tính joined\_date[112](#_bookmark486)

JVM (máy ảo Java)[490–](#_bookmark2250)[491](#_bookmark2259)

**K**

keep\_words lọc/danh sách[261](#_bookmark1185)

đại diện chính[455](#_bookmark2085)

phân tích từ khóa[252–253](#_bookmark1145)

kiểu dữ liệu từ khóa[118,](#_bookmark532)[122](#_bookmark550)[–124,](#_bookmark558)[316,](#_bookmark1469)[328,](#_bookmark1526)[431,](#_bookmark1962)

[440](#_bookmark2016)

lựa chọn qua các loại văn bản[50](#_bookmark2309)1–502loại từ khóa hằng số[123–](#_bookmark554)[124](#_bookmark556) sửa chữa sắp xếp sử dụng[108–10](#_bookmark466)[9](#_bookmark467) loại từ khóa [12](#_bookmark551)2–123

kiểu dữ liệu đại diện[124](#_bookmark557) từ khóa tokenizer[274](#_bookmark1254)

các loại từ khóa[1](#_bookmark551)22–123

Kibana[12](#_bookmark55)

cài đặt[520–5](#_bookmark2374)[22](#_bookmark2380)

tải xuống nhị phân[521](#_bookmark2375) cài đặt qua Docker[522](#_bookmark2379) khởi động trên macOS[522](#_bookmark2378) khởi động trên Windows[521](#_bookmark2376)

ảnh chụp nhanh

tự động hóa[485–](#_bookmark2220)[487](#_bookmark2223)

đăng ký kho lưu trữ[480](#_bookmark2197)[–481](#_bookmark2199)

Lập trình Kotlin[347](#_bookmark1622)

KQL (Ngôn ngữ truy vấn Kibana)[352](#_bookmark1649)

**L**

tham số lang[177](#_bookmark807)

máy phân tích ngôn ngữ[256–258](#_bookmark1165) mô hình ngôn ngữ lớn (LLM)[17](#_bookmark81) truy vấn lá[47,](#_bookmark215)[296–](#_bookmark1376)[297](#_bookmark1379)

toán tử nhỏ hơn (lt)[128,](#_bookmark571)[323](#_bookmark1509)

toán tử nhỏ hơn hoặc bằng (lte)[323,](#_bookmark1510)[325](#_bookmark1513)

**MỤC LỤC 551**

Thuật toán khoảng cách Levenshtein[331](#_bookmark1541) Thuật toán khoảng cách chỉnh sửa Levenshtein[44](#_bookmark193) chính sách vòng đời[228](#_bookmark1039)

giống như tham số[428](#_bookmark1954) Hình dạng chuỗi dòng[404](#_bookmark1858)

LLM (mô hình ngôn ngữ lớn)[17](#_bookmark81) tìm kiếm vị trí[398–401](#_bookmark1830)

truy vấn bounding\_box[398–](#_bookmark1832)[399](#_bookmark1834)

truy vấn geo\_distance[399](#_bookmark1836)

truy vấn geo\_shape[400–401](#_bookmark1838) các loại dữ liệu không gian địa lý[401–404](#_bookmark1842)

Kiểu dữ liệu geo\_point[401–4](#_bookmark1845)[02](#_bookmark1849) hình dạng địa lýkiểu dữ liệu[402–404](#_bookmark1852)

truy vấn không gian địa lý[404–41](#_bookmark1863)[2](#_bookmark1891)

truy vấn geo\_bounding\_box[405–409](#_bookmark1866)

truy vấn geo\_distance[409–4](#_bookmark1880)[10](#_bookmark1882)

truy vấn geo\_shape[410–412](#_bookmark1884)

tình trạng khóa chặt[340](#_bookmark1577) ghi nhật ký

tùy chọn cấu hình nâng cao[488–490](#_bookmark2237) giám sát ứng dụng và[13](#_bookmark65)

Đăng nhập[12](#_bookmark53)

lon (kinh độ)[132](#_bookmark599)

kiểu dữ liệu dài[46,](#_bookmark200)[106,](#_bookmark458)[115,](#_bookmark509)[118,](#_bookmark533)[431](#_bookmark1963)

hình mờ đĩa thấp[509](#_bookmark2339)

bộ lọc chữ thường[245–246,](#_bookmark1123)[2](#_bookmark1151)[53–254,](#_bookmark1157)[342](#_bookmark1590)

bộ phân tích chữ thường[251,](#_bookmark1139)[274](#_bookmark1255)

Toán tử lt (nhỏ hơn)[128,](#_bookmark571)[323](#_bookmark1509)

toán tử lte (nhỏ hơn hoặc bằng)[323,](#_bookmark1510)[325](#_bookmark1513)

**Tôi**

Vai trò của nút học máy[84](#_bookmark361) macOS

cài đặt Elasticsearch[51](#_bookmark2364)8–519

cài đặt Kibana [522](#_bookmark2378) tập tin cấu hình chính[488](#_bookmark2234) tạo thuộc tính[201](#_bookmark921)

thao túng kết quả[307–310](#_bookmark1428)

lập bản đồ[100–148](#_bookmark430)

tạo chỉ mục[201–202](#_bookmark920)

các kiểu dữ liệu[115–](#_bookmark506)[117](#_bookmark529)

trình độ cao [131–146](#_bookmark596) Kiểu dữ liệu boolean[127–12](#_bookmark566)[8](#_bookmark568) kiểu dữ liệu ngày[124–](#_bookmark560)[126](#_bookmark561)

các trường có nhiều[146–1](#_bookmark653)[47](#_bookmark659) Kiểu dữ liệu IP[13](#_bookmark587)0–131từ khóa[122–1](#_bookmark550)[24](#_bookmark558)

số[127](#_bookmark563)

phạm vi[128–130](#_bookmark570)

kiểu dữ liệu văn bản[1](#_bookmark535)18–122năng động[104–1](#_bookmark451)[09](#_bookmark471)

những hạn chế của[1](#_bookmark461)07–109

cơ chế để suy ra các loại[105–1](#_bookmark455)[07](#_bookmark459) rõ ràng[109–11](#_bookmark474)[5](#_bookmark504)

sửa đổi các trường hiện có[1](#_bookmark496)13–114

loại ép buộc[11](#_bookmark502)4–115sử dụng API lập chỉ mục[110–113](#_bookmark478)

các tập tin chứa ánh xạ[2](#_bookmark1227)68–269lập chỉ mục tài liệu[103–1](#_bookmark439)[04](#_bookmark448)

định nghĩa lập bản đồ[102](#_bookmark434)

Tổng quan[101–10](#_bookmark432)[4](#_bookmark449)

bộ lọc ký tự ánh xạ[26](#_bookmark1222)7–268bộ lọc ánh xạ[2](#_bookmark1212)66–267

đối tượng lập bản đồ [102](#_bookmark436)

đối tượng ánh xạ[110,](#_bookmark479)[113,](#_bookmark490)[117](#_bookmark528) Vai trò nút chính[84](#_bookmark358)

vai trò chủ chốt[85,](#_bookmark372)[491,](#_bookmark2265)[494–49](#_bookmark2281)[5,](#_bookmark2285)[524](#_bookmark2391)

nút đủ điều kiện chủ [491](#_bookmark2262)

thuộc tính phù hợp[182,](#_bookmark842)[303](#_bookmark1414)

chức năng khớp[538](#_bookmark2457) Tổng quan về truy vấn cụm từ khớp[42–43](#_bookmark179)

cụm từ có từ bị thiếu[44](#_bookmark188) truy vấn phù hợp[3](#_bookmark1580)40–345

phân tích [342](#_bookmark1588) sách của tác giả[36](#_bookmark149)

sửa lỗi đánh máy bằng cách sử dụng từ khóa fuzziness [34](#_bookmark1603)4–345định dạng của[340–3](#_bookmark1582)[41](#_bookmark1584)

phù hợp với ít nhất một vài từ[343–344](#_bookmark1597) tìm kiếm nhiều từ[342–343](#_bookmark1592) tìm kiếm bằng cách sử dụng[341](#_bookmark1586)

với toán tử AND[37–39](#_bookmark154) đối tượng match\_all[338](#_bookmark1569)

truy vấn match\_all[35,](#_bookmark141)[184,](#_bookmark851)[33](#_bookmark1566)8–339

xây dựng[338–33](#_bookmark1568)[9](#_bookmark1571) dạng viết tắt của[339](#_bookmark1573)

truy vấn match\_none[339–340](#_bookmark1576)

loại match\_only\_text[116,](#_bookmark523)[120](#_bookmark542)

truy vấn match\_phrase[42,](#_bookmark180)[44,](#_bookmark189)[50,](#_bookmark225)[345–34](#_bookmark1609)[6](#_bookmark1615)

truy vấn match\_phrase\_prefix[346–347](#_bookmark1617)

truy vấn khớp/thuật ngữ[430](#_bookmark1959)

đối tượng matches\_queries[383](#_bookmark1764)

Đối tượng MatchQuery.Builder[538](#_bookmark2458) Thiết lập dự án Maven/Gradle[531–532](#_bookmark2414) tổng hợp tối đa[53,](#_bookmark246)[55,](#_bookmark259)[436](#_bookmark1984)

Thùng tối đa (max\_bucket)[462](#_bookmark2125) chức năng tối đa[435](#_bookmark1973)

chế độ tối đa[395](#_bookmark1813)

tổng hợp max\_bucket[46](#_bookmark2137)4–465

thuộc tính max\_gram [273](#_bookmark1246)

thuộc tính max\_score [291](#_bookmark1334)

thuộc tính max\_token\_length[270](#_bookmark1239)

kích thước max\_token\_length[271](#_bookmark1241)

số liệu tối đa[442](#_bookmark2030)

thiết lập maximum\_count[487](#_bookmark2221)

thuộc tính siêu dữ liệu[482](#_bookmark2205)

siêu dữ liệu, xóa khỏi phản hồi[166](#_bookmark744) tổng hợp số liệu[53–56,](#_bookmark262)[437–](#_bookmark2005)[445](#_bookmark2044)

số liệu trung bình[440–4](#_bookmark2020)[41](#_bookmark2022)

số liệu về số lượng [444–445](#_bookmark2042)

số liệu thống kê mở rộng[443](#_bookmark2036)

**552**

tổng hợp số liệu*(tiếp theo)*

số liệu tối đa[442](#_bookmark2030)

số liệu tối thiểu[441–442](#_bookmark2026)

số liệu khác [5](#_bookmark258)5–56

dữ liệu mẫu[438](#_bookmark2010)

số liệu thống kê[44](#_bookmark2032)2–443

tổng số mét [54](#_bookmark256)[–55,](#_bookmark257)[441](#_bookmark2024)

số liệu value\_count[438–4](#_bookmark2012)[40](#_bookmark2017)

tổng hợp tối thiểu[53,](#_bookmark247)[436](#_bookmark1985)

Xô của tôi (my\_bucket)[462](#_bookmark2126) hàm min[435](#_bookmark1974)

từ khóa tối thiểu [442](#_bookmark2029)

min mét[441](#_bookmark2027)

thời trang của tôi[395](#_bookmark1814)

thuộc tính min\_age[233](#_bookmark1058)

tổng hợp min\_bucket[465](#_bookmark2139)

thuộc tính min\_gram[273](#_bookmark1247)

số liệu tối thiểu[441–442](#_bookmark2026)

thiết lập minimum\_count[487](#_bookmark2222)

thuộc tính minimum\_should\_match[344,](#_bookmark1599)[377](#_bookmark1746)

thiết lập minimum\_should\_match [377](#_bookmark1747)[–378](#_bookmark1750)

tham số minumum\_should\_match [378](#_bookmark1748)

vai trò ml[524](#_bookmark2392)

thuộc tính mô hình[201](#_bookmark922)

loại mô hình[195](#_bookmark895)

truy vấn more\_like\_this[42](#_bookmark1951)7–428truy vấn nhiều trường nhất[349](#_bookmark1632) Chức năng di chuyển (moving\_function)[461](#_bookmark2116)

Di chuyển phần trăm (moving\_percentiles)[461](#_bookmark2117) truy vấn multi\_match[41,](#_bookmark170)[146,](#_bookmark646)[295,](#_bookmark1368)[347,](#_bookmark1625)[388](#_bookmark1787)

những cánh đồng tốt nhất[348](#_bookmark1627)[–349](#_bookmark1636)

thúc đẩy các lĩnh vực riêng lẻ[351–](#_bookmark1646)[352](#_bookmark1648) truy vấn dis\_max[350](#_bookmark1638)

người phá vỡ thế giằng co[350](#_bookmark1641)[–351](#_bookmark1644)

API đa tài liệu[150](#_bookmark663)

tìm kiếm đa trường[41](#_bookmark169)

truy vấn nhiều trận đấu[146,](#_bookmark647)[349](#_bookmark1633)

cụm nhiều nút[469](#_bookmark2149)

tổng hợp nhiều thuật ngữ[4](#_bookmark2083)54–455

trường đa kiểu[104](#_bookmark443) Hình dạng chuỗi nhiều dòng[404](#_bookmark1859) chế độ nhân[395](#_bookmark1815)

Hình dạng đa điểm[404](#_bookmark1855)

Hình đa giác[404](#_bookmark1857)

phải mệnh đề[49](#_bookmark220)–50

tăng cường[368](#_bookmark1723)[–369](#_bookmark1726)

Tổng quan[367–3](#_bookmark1716)[68](#_bookmark1720)

phải truy vấn[380,](#_bookmark1755)[382,](#_bookmark1759)[385](#_bookmark1774)

mệnh đề must\_not[50–51](#_bookmark228)

tăng cường[371](#_bookmark1735)[–373](#_bookmark1738)

Tổng quan[370–3](#_bookmark1728)[71](#_bookmark1733)

truy vấn must\_not[373](#_bookmark1737)

nhãn my\_fields1.kw[147](#_bookmark655)

thuộc tính myshape[412](#_bookmark1897)

thuộc tính myshape[412](#_bookmark1898)

chỉ số myshapes[412–413](#_bookmark1899) mẫu chỉ mục mysql-\*[231](#_bookmark1053)

**MỤC LỤC**

**N**

n-gram145

Thuộc tính NAME[436](#_bookmark1986)

tên thuộc tính[143](#_bookmark639)

tên tài sản[401,](#_bookmark1847)[412](#_bookmark1900)

tên:cặp giá trị[291](#_bookmark1340)

không gian tên khách hàng [533](#_bookmark2425)

truy vấn phủ định[386](#_bookmark1781)

thuộc tính negative\_boost[387](#_bookmark1782)

thiết lập negative\_boost[387](#_bookmark1783)

tổng hợp lồng nhau[436–43](#_bookmark1993)[7](#_bookmark1995)

kiểu dữ liệu lồng nhau[115–116,](#_bookmark521)[131,](#_bookmark592)[137–13](#_bookmark622)[9](#_bookmark628)

thuộc tính mạng[474](#_bookmark2167)

mạng.\* thuộc tính[488](#_bookmark2235)

thuộc tính network.host[475](#_bookmark2169)

thuộc tính network.host[475](#_bookmark2170)

chỉ số mạng[130,](#_bookmark589)[452](#_bookmark2076)

thùng tin tức[63](#_bookmark286)

mã thông báo ngram[272–273](#_bookmark1244)

thuộc tính node.roles[471](#_bookmark2156)

thiết lập node.roles[86](#_bookmark373)

các nút[81–86](#_bookmark345)

thêm vào các cụm[469–470](#_bookmark2147) nhóm chủ

các nút chính chuyên dụng[494](#_bookmark2284)[–495](#_bookmark2287) nút chính[491](#_bookmark2264)

đa cụm đa nút[83](#_bookmark352)–84

giao tiếp nút[474–4](#_bookmark2166)[75](#_bookmark2171) vai trò

cấu hình[85–86](#_bookmark369)

Tổng quan[84–85](#_bookmark357)

cụm nút đơn[81–82](#_bookmark346)

các trường không phải văn bản[46](#_bookmark203)

trình phân tích noop (không hoạt động)[252](#_bookmark1146) máy phân tích đặc biệt noop[104](#_bookmark444) chuẩn hóa[65,](#_bookmark293)[239](#_bookmark1085)

Chuẩn hóa (normalize) [461](#_bookmark2118)

dữ liệu chuẩn hóa[7](#_bookmark23)

thuộc tính chuẩn hóa[104](#_bookmark446)

Toán tử KHÔNG[353](#_bookmark1652)

từ khóa bây giờ[325](#_bookmark1514)

nteger, dài, ngắn, kiểu byte[116](#_bookmark517) Kiểu số[241](#_bookmark1098)

thuộc tính number\_of\_replicas[200,](#_bookmark915)[223](#_bookmark1022)

thiết lập number\_of\_replicas[195,](#_bookmark897)[473](#_bookmark2162)

biến number\_of\_shards[97](#_bookmark416) kiểu dữ liệu số[127](#_bookmark563)

**THE**

Kiểu dữ liệu đối tượng[112,](#_bookmark481)[115–1](#_bookmark512)[16,](#_bookmark520)[131](#_bookmark593)

những hạn chế của[13](#_bookmark618)6–137

Tổng quan[134–13](#_bookmark605)[7](#_bookmark621)

Thuật toán Okapi BM25[92–94](#_bookmark398)

Dữ liệu OLAP (Xử lý phân tích trực tuyến)[14](#_bookmark69)

**MỤC LỤC 553**

hiệu sách trực tuyếnví dụ

truy vấn khớp để tìm sách theo tác giả[36](#_bookmark149) Tổng quan[21–23](#_bookmark88)

tham số toán tử[38](#_bookmark157)

*Tối ưu hóa Java*(Evan và Gough)[490](#_bookmark2254) Toán tử Boolean HOẶC[343](#_bookmark1593)

Toán tử OR[38,](#_bookmark155)[292–293,](#_bookmark1350)[343–344,](#_bookmark1601)[353,](#_bookmark1656)[355,](#_bookmark1665)

[373](#_bookmark1741)[–374,](#_bookmark1742)[420–4](#_bookmark1929)[21](#_bookmark1930)

Loại lệnh[70](#_bookmark314)

orders\_new chỉ mục[114](#_bookmark498)

gói org.elasticsearch.http[490](#_bookmark2244)

gói org.elasticsearch.index[489–490](#_bookmark2242) đối tượng va chạm bên ngoài[291](#_bookmark1335)

**P**

phân trang[298](#_bookmark1384)[–299](#_bookmark1393)

truy vấn pan\_near[419](#_bookmark1925) loại truy vấn pan\_within[418](#_bookmark1924) thuộc tính params[176](#_bookmark799)

cờ params[177](#_bookmark804)

đối tượng params[177](#_bookmark803)

thuộc tính params [176](#_bookmark802)

tập hợp cha mẹ[456](#_bookmark2090) Cầu dao phụ huynh[512](#_bookmark2347) truy vấn tìm kiếm parent\_id[144](#_bookmark642)

trình phân tích path\_hierarchy[243,](#_bookmark1105)[274](#_bookmark1257)

tham số path.data[84](#_bookmark353)

biến path.data[63](#_bookmark285)

tham số path.logs [84](#_bookmark354)

thuộc tính path.repo [479](#_bookmark2191)

máy phân tích mẫu[236,](#_bookmark1064)[243,](#_bookmark1115)[254–255,](#_bookmark1163)[262](#_bookmark1192)

thuộc tính mẫu[255](#_bookmark1162)

bộ lọc mẫu[266](#_bookmark1213)

mẫu tokenizer[254,](#_bookmark1158)[274](#_bookmark1252)

bộ lọc ký tự pattern\_replace[240,](#_bookmark1090)[269–](#_bookmark1230)[270](#_bookmark1232)

mẫu hình[255](#_bookmark1160)

PDF, tải vào Elasticsearch[527–529](#_bookmark2403) Xô phần trăm (percentiles\_bucket)[462](#_bookmark2127) truy vấn thấm lọc[428–432](#_bookmark1957)

loại trường lọc[429](#_bookmark1958) chỉ số bình pha cà phê[430–432](#_bookmark1960)

truy vấn bình pha cà phê[398](#_bookmark1827)

loại bình pha cà phê[431](#_bookmark1964)

hiệu suất và xử lý sự cố[49](#_bookmark2290)7–514cầu dao điện[511–512](#_bookmark2346)

vấn đề về tốc độ chỉ số[503](#_bookmark2313)[–506](#_bookmark2325) điều chỉnh tốc độ làm mới[503](#_bookmark2320)[–506](#_bookmark2326) yêu cầu số lượng lớn[503](#_bookmark2317)

định danh do hệ thống tạo ra[503](#_bookmark2315) vấn đề tìm kiếm và tốc độ[498–502](#_bookmark2292)

chọn loại từ khóa thay vì loại văn bản[501–](#_bookmark2309) [502](#_bookmark2311)

mô hình hóa tài liệu[499–501](#_bookmark2301)

phần cứng hiện đại [498–49](#_bookmark2294)[9](#_bookmark2299)

cụm không ổn định[506–511](#_bookmark2328)

sức khỏe cụm[50](#_bookmark2330)6–507

ngưỡng sử dụng đĩa[509–5](#_bookmark2338)[11](#_bookmark2343)

các mảnh vỡ chưa được chỉ định[507–509](#_bookmark2333)

tính chất bền bỉ[196,](#_bookmark900)[490](#_bookmark2246)

thuộc tính phone\_number[112](#_bookmark487) cụm từ truy vấn nhiều trận đấu[349](#_bookmark1634)

truy vấn đa khớp cụm từ\_tiền tố[349](#_bookmark1635) tham số phrase\_slop[356](#_bookmark1671)

thiết lập cụm từ\_slop[356](#_bookmark1673)

truy vấn được ghim[398,](#_bookmark1828)[426–42](#_bookmark1948)[7](#_bookmark1949)

tổng hợp đường ống[458–4](#_bookmark2096)[65](#_bookmark2140)

có sẵn[461–46](#_bookmark2109)[2](#_bookmark2130)

tổng hợp cha tích lũy [46](#_bookmark2133)2–464tổng hợp max\_bucket[46](#_bookmark2137)4–465

tổng hợp min\_bucket[465](#_bookmark2139)

dữ liệu mẫu[459](#_bookmark2100)

cú pháp cho[459–4](#_bookmark2103)[61](#_bookmark2107)

các loại[459](#_bookmark2098)

đường ống[524](#_bookmark2385)

Hình dạng điểm[404](#_bookmark1854)

điểm, không gian địa lý[404](#_bookmark1864)

Hình đa giác[404](#_bookmark1856)

phần tích cực[386](#_bookmark1780)

Lệnh POST /\_refresh[161](#_bookmark706) Hành động POST[154](#_bookmark686)

POST động từ HTTP[151](#_bookmark672)

Phương pháp POST[29,](#_bookmark109)[154–156,](#_bookmark692)[171,](#_bookmark765)[185,](#_bookmark862)[289,](#_bookmark1319)[537](#_bookmark2445)

Yêu cầu POST[154,](#_bookmark688)[185](#_bookmark858)

phần trăm chính xác[282](#_bookmark1286)

truy vấn tiền tố[37,](#_bookmark152)[327–330](#_bookmark1520)

rút ngắn[328–3](#_bookmark1528)[29](#_bookmark1529)

tăng tốc[329](#_bookmark1531)[–330](#_bookmark1536) truy vấn cấp độ thuật ngữ tiền tố[47](#_bookmark211) giá đối tượng lồng nhau[23](#_bookmark93) thùng sơ cấp[217–218](#_bookmark1002)

mảnh vỡ chính[72,](#_bookmark319)[473](#_bookmark2160) mồi với dữ liệu[21–30](#_bookmark86)

lập chỉ mục tài liệu[23–26](#_bookmark96) thành phần của yêu cầu[29–30](#_bookmark106) cURL[24–26](#_bookmark100)

tài liệu API[23–](#_bookmark97)[24](#_bookmark99)

quá trình cho[26–3](#_bookmark104)[0](#_bookmark115)

ví dụ về hiệu sách trực tuyến[2](#_bookmark88)1–23phân loại riêng tư[139](#_bookmark627)

xử lý dữ liệu[63–](#_bookmark283)[66](#_bookmark294)

phân tích dữ liệu[6](#_bookmark290)4–66

thu thập dữ liệu[63](#_bookmark284) các loại dữ liệu[64](#_bookmark288)

bộ xử lý[524](#_bookmark2386)

chính sách prod\_cluster\_daily\_backups[487](#_bookmark2228)

định nghĩa sản phẩm[363](#_bookmark1693)

chỉ mục sản phẩm[362–](#_bookmark1692)[363,](#_bookmark1694)[367](#_bookmark1717)

chỉ mục hồ sơ[427](#_bookmark1952)

đối tượng thuộc tính[102,](#_bookmark437)[112–113](#_bookmark482) chỉ số công khai, đọc[207](#_bookmark940)[–208](#_bookmark943) Lệnh PUT[110](#_bookmark480)

**554**

ĐẶT động từ HTTP[151](#_bookmark673)

Phương pháp PUT[29,](#_bookmark110)[151–156](#_bookmark676)

ĐẶT yêu cầu[155,](#_bookmark689)[177](#_bookmark811)

**Hỏi**

đối tượng truy vấn [389](#_bookmark1788)

ngữ cảnh truy vấn[286–](#_bookmark1308)[287](#_bookmark1309)

Truy vấn DSL[182,](#_bookmark843)[294–297](#_bookmark1363)

cho các tập hợp[296](#_bookmark1374)

cho cURL [295–296](#_bookmark1370)

truy vấn lá và hợp chất[296–297](#_bookmark1376) mẫu truy vấn[295](#_bookmark1367)

Tìm kiếm yêu cầu URI[294](#_bookmark1359)

đối tượng truy vấn[33,](#_bookmark137)[36,](#_bookmark151)[41,](#_bookmark171)[43,](#_bookmark184)[49,](#_bookmark224)[53,](#_bookmark251)[290,](#_bookmark1322)[295–297,](#_bookmark1365)

[300,](#_bookmark1398)[302,](#_bookmark1405)[311,](#_bookmark1449)[346,](#_bookmark1613)[364,](#_bookmark1703)[368,](#_bookmark1721)[435](#_bookmark1982)

tham số truy vấn[353](#_bookmark1657)

yêu cầu truy vấn[281](#_bookmark1285)

phương thức query\_string[294](#_bookmark1360)

truy vấn query\_string[335,](#_bookmark1548)[352–](#_bookmark1650)[358](#_bookmark1680)

toán tử mặc định[355](#_bookmark1664)

các lĩnh vực trong [354–3](#_bookmark1659)[55](#_bookmark1662)

với các cụm từ[356](#_bookmark1669)

thuộc tính queue\_size[505](#_bookmark2323)

đủ số lượng[493](#_bookmark2278)

trích dẫn tài sản [415](#_bookmark1909)

trích dẫn chỉ số[415](#_bookmark1907)

**R**

hàm random\_score[3](#_bookmark1797)91–392

tổng hợp phạm vi[436](#_bookmark1987)

loại thùng tổng hợp phạm vi[58](#_bookmark271) thuộc tính phạm vi[182](#_bookmark844)

tổng hợp phân loại phạm vi[57](#_bookmark270) thùng đựng phạm vi[57–](#_bookmark269)[58](#_bookmark274)

kiểu dữ liệu phạm vi kiểu date\_range[128–130](#_bookmark577)

Tổng quan[128–1](#_bookmark570)[30](#_bookmark581)

truy vấn phạm vi[327,](#_bookmark1521)[368–36](#_bookmark1724)[9,](#_bookmark1725)[373](#_bookmark1736) xóa tài liệu với[1](#_bookmark847)83–184tìm kiếm theo cấp độ thuật ngữ[47,](#_bookmark208)[323–325](#_bookmark1506)

phạm vi truy vấn cấp độ thuật ngữ[315](#_bookmark1464) phần trăm thu hồi[282](#_bookmark1287) quá trình làm mới

điều chỉnh tốc độ làm mới[503](#_bookmark2320)[–506](#_bookmark2326) tùy chỉnh[16](#_bookmark703)0–161

kiểm soát làm mới phía máy khách[161](#_bookmark708) cấu hình chu kỳ làm mới[160–](#_bookmark704)[161](#_bookmark707)

làm mới tham số truy vấn[161](#_bookmark709) thuộc tính registration\_year[201](#_bookmark923)

lập chỉ mục lại[97,](#_bookmark417)[113](#_bookmark497)

thuộc tính quan hệ[411](#_bookmark1885)

thuộc tính quan hệ [142](#_bookmark638)

đối tượng quan hệ[142](#_bookmark636)

**MỤC LỤC**

đối tượng quan hệ[143](#_bookmark640)

tài sản quan hệ[142](#_bookmark637)

sự liên quan[4,](#_bookmark15)[89–90](#_bookmark382)

thuật toán liên quan (tương tự)[90–96](#_bookmark392) cấu hình[94–96](#_bookmark406)

Thuật toán Okapi BM25[92–94](#_bookmark398) điểm liên quan[89–90,](#_bookmark389)[301](#_bookmark1401)[–304](#_bookmark1418)

\_giải thích API[30](#_bookmark1410)3–304

giải thích cờ[301](#_bookmark1403)[–303](#_bookmark1409)

sắp xếp trên[304–3](#_bookmark1422)[07](#_bookmark1425)

phương pháp loại bỏ[173](#_bookmark788)

loại bỏ hoạt động[180](#_bookmark828)

thay thế tài liệu[177–1](#_bookmark810)[78](#_bookmark814)

mảnh vỡ bản sao[72,](#_bookmark320)[473](#_bookmark2161)

phân phối của[7](#_bookmark332)6–79

Tổng quan[74–75](#_bookmark329)

bản sao[72](#_bookmark321)

nhóm sao chép[283](#_bookmark1293) Yêu cầu ngắt mạch[512](#_bookmark2349) Trình xây dựng RestClient[532](#_bookmark2421)

thể hiện restClient [533](#_bookmark2423)

đối tượng restClient[532](#_bookmark2420)

thuộc tính kết quả[1](#_bookmark677)[53–154,](#_bookmark681)[182](#_bookmark839)

kết quả thuộc tính[28](#_bookmark105)

đang lấy dữ liệu [3](#_bookmark119)0–36

đếm tài liệu[31](#_bookmark122)

đang lấy lại tài liệu[31–36](#_bookmark126)[,](#_bookmark143)[161–166](#_bookmark714)

truy vấn id[165–16](#_bookmark736)[6](#_bookmark740)

đang lấy lại tất cả các tài liệu[34](#_bookmark139)–36lấy lại nhiều tài liệu[33–34,](#_bookmark133)

[163–1](#_bookmark726)[65](#_bookmark733)

lấy lại các tài liệu đơn lẻ[32–33,](#_bookmark132)[162](#_bookmark716)[–163](#_bookmark724)

bộ lọc ngược[274](#_bookmark1262)

hành động lật ngược[232](#_bookmark1055) bí danh hoạt động chuyển đổi[224–227](#_bookmark1026)

tạo cho các hoạt động chuyển đổi[225](#_bookmark1028) phát hành các hoạt động chuyển nhượng[225–](#_bookmark1030)[227](#_bookmark1036)

vòng đời chỉ mục với[231–234](#_bookmark1052) thuật toán định tuyến[96–97](#_bookmark414)

**S**

người quyết định same\_shard [471](#_bookmark2155)

tỉ lệ[97](#_bookmark420)

nằm ngang[98,](#_bookmark426)[477](#_bookmark2180)

cụm mở rộng[4](#_bookmark2145)68–474

thêm các nút vào cụm[46](#_bookmark2147)9–470sức khỏe cụm[47](#_bookmark2153)0–473

tăng thông lượng đọc[473–474](#_bookmark2159) thẳng đứng[97–98,](#_bookmark424)[477](#_bookmark2179)

không có sơ đồ[30](#_bookmark113)

tính năng không có lược đồ[69](#_bookmark310)

tổng hợp có phạm vi[436](#_bookmark1991)

thuộc tính score\_mode[395](#_bookmark1816)

Bộ ngắt mạch biên dịch tập lệnh[512](#_bookmark2352)

**MỤC LỤC 555**

đối tượng kịch bản[173](#_bookmark782)

phần kịch bản[179](#_bookmark817)

đối tượng script\_fields[309](#_bookmark1434)

hàm script\_score[392–](#_bookmark1801)[393](#_bookmark1802)

các trường có kịch bản[308–](#_bookmark1433)[309](#_bookmark1435)

cập nhật theo kịch bản[173–177](#_bookmark779)

thêm các tập lệnh cập nhật có điều kiện[175](#_bookmark796) thêm nhiều trường[174](#_bookmark794)[–175](#_bookmark795) thêm các trường mới[174](#_bookmark791)

giải phẫu của các tập lệnh[175–17](#_bookmark798)[6](#_bookmark800) truyền dữ liệu vào các tập lệnh[176](#_bookmark801)[–177](#_bookmark805)

xóa các phần tử khỏi mảng[173–1](#_bookmark786)[74](#_bookmark790) xóa các trường[174](#_bookmark793)

ngôn ngữ kịch bản [177](#_bookmark806) cập nhật mảng bằng cách sử dụng tập lệnh[173](#_bookmark784)

API cuộn[299](#_bookmark1388)

tìm kiếm[280–31](#_bookmark1279)[2,](#_bookmark1452)[397–433](#_bookmark1823)

giải phẫu của các yêu cầu[288–2](#_bookmark1317)[90](#_bookmark1323) giải phẫu của phản ứng[290](#_bookmark1325)[–291](#_bookmark1336) như bình thường mới3–7

khách hàng[538–53](#_bookmark2454)[9](#_bookmark2461) cơ sở dữ liệu

công cụ tìm kiếm so với 6–7tìm kiếm được hỗ trợ bởi5–6

đặc điểm của[297](#_bookmark1382)[–312](#_bookmark1451)

làm nổi bật[29](#_bookmark1396)9–301

thao túng kết quả[307–310](#_bookmark1428)

phân trang[298](#_bookmark1384)[–299](#_bookmark1393)

điểm liên quan[301–304](#_bookmark1401) tìm kiếm trên các chỉ mục và dữ liệu

suối[310–31](#_bookmark1447)[2](#_bookmark1450)

phân loại[304–30](#_bookmark1420)[7](#_bookmark1426)

tìm kiếm toàn văn[334–360](#_bookmark1547)

truy vấn mơ hồ[357](#_bookmark1676)

truy vấn phù hợp[340–](#_bookmark1580)[345](#_bookmark1606)

truy vấn match\_all[338–339](#_bookmark1566)

truy vấn match\_none [339–34](#_bookmark1576)[0](#_bookmark1578)

truy vấn match\_phrase[345–346](#_bookmark1609)

truy vấn match\_phrase\_prefix[346–347](#_bookmark1617)

truy vấn multi\_match[347–3](#_bookmark1625)[52](#_bookmark1647)

Tổng quan[335–3](#_bookmark1550)[38](#_bookmark1560)

độ chính xác[335–336](#_bookmark1552)

chuỗi truy vấn truy vấn[353–35](#_bookmark1654)[6](#_bookmark1674)

nhớ lại[336–338](#_bookmark1557)

dữ liệu mẫu[338](#_bookmark1563)

truy vấn simple\_query\_string[357–359](#_bookmark1679)

chức năng của[282–283](#_bookmark1290)

những điều cơ bản của [285–288](#_bookmark1300)

\_điểm cuối tìm kiếm[285–286](#_bookmark1303)

bối cảnh thực hiện[286–28](#_bookmark1306)[8](#_bookmark1314) AI tạo ra và[16–19](#_bookmark79) tìm kiếm vị trí[398–401](#_bookmark1830)

truy vấn bounding\_box[398–](#_bookmark1832)[399](#_bookmark1834)

truy vấn geo\_distance[399](#_bookmark1836)

truy vấn geo\_shape[400–401](#_bookmark1838)

dữ liệu không gian địa lýcác loại[401–40](#_bookmark1842)[4](#_bookmark1861) truy vấn không gian địa lý[404–412](#_bookmark1863)

Tổng quan[281–28](#_bookmark1281)[2](#_bookmark1288)

hiệu suất và xử lý sự cố[498–](#_bookmark2292)[502](#_bookmark2310) chọn loại từ khóa thay vì loại văn bản

[501–5](#_bookmark2309)[02](#_bookmark2311)

mô hình hóa tài liệu[499–50](#_bookmark2301)[1](#_bookmark2307)

phần cứng hiện đại[498–499](#_bookmark2294)

Truy vấn DSL[294–297](#_bookmark1363)

cho các tập hợp[296](#_bookmark1374)

cho cURL[295–2](#_bookmark1370)[96](#_bookmark1371)

truy vấn lá và hợp chất[296–297](#_bookmark1376) mẫu truy vấn[295](#_bookmark1367)

dữ liệu mẫu[283](#_bookmark1296)[–285](#_bookmark1298)

công cụ tìm kiếm[7–](#_bookmark27)[9](#_bookmark38)

đặc điểm của2–3

cơ sở dữ liệu so với6–7

chức năng của [8](#_bookmark29)

phổ biến8-9

truy vấn hình dạng[412](#_bookmark1894)[–414](#_bookmark1902)

truy vấn khoảng[414](#_bookmark1904)[–421](#_bookmark1934)

dữ liệu mẫu[415](#_bookmark1906)

truy vấn span\_first[416–](#_bookmark1911)[417](#_bookmark1915)

truy vấn span\_near[417–418](#_bookmark1917)

truy vấn span\_or[4](#_bookmark1928)20–421

truy vấn span\_within[418–420](#_bookmark1923)

truy vấn chuyên biệt[421–432](#_bookmark1937)

truy vấn distance\_feature[421–](#_bookmark1939)[426](#_bookmark1945)

truy vấn more\_like\_this[427–4](#_bookmark1951)[28](#_bookmark1955)

truy vấn thấm lọc[428–](#_bookmark1957)[432](#_bookmark1968)

truy vấn được ghim[42](#_bookmark1948)6–427chỉ định các máy phân tích cho[263](#_bookmark1198)[–265](#_bookmark1208)

bộ phân tích trong truy vấn[263](#_bookmark1199)

trình phân tích mặc định ở cấp độ chỉ mục[264–265](#_bookmark1202) thứ tự ưu tiên[265](#_bookmark1205)

thiết lập máy phân tích ở cấp độ hiện trường[264](#_bookmark1200) dữ liệu có cấu trúc[4](#_bookmark14)

tìm kiếm theo cấp độ thuật ngữ[313–333](#_bookmark1454)

ví dụ về [314–315](#_bookmark1460)

tồn tại truy vấn[322–](#_bookmark1502)[323](#_bookmark1504)

truy vấn mơ hồ[33](#_bookmark1538)0–332

truy vấn id[321–32](#_bookmark1498)[2](#_bookmark1499)

không được phân tích[314](#_bookmark1458)

Tổng quan[314–31](#_bookmark1456)[5](#_bookmark1466)

truy vấn tiền tố [327](#_bookmark1525)[–330](#_bookmark1535)

truy vấn phạm vi[323](#_bookmark1506)[–325](#_bookmark1515)

truy vấn thuật ngữ[316–318](#_bookmark1468)

truy vấn thuật ngữ[318](#_bookmark1483)[–321](#_bookmark1495)

truy vấn ký tự đại diện[3](#_bookmark1517)25–327

dữ liệu không có cấu trúc4–5

Tìm kiếm yêu cầu URI[291–2](#_bookmark1338)[94](#_bookmark1361) các thông số bổ sung[293–294](#_bookmark1353) tìm kiếm phim theo tiêu đề[292](#_bookmark1342) tìm kiếm những bộ phim cụ thể[293](#_bookmark1347) hỗ trợ với Query DSL[294](#_bookmark1359)

**556**

công cụ tìm kiếm7–9

đặc điểm của2–3

cơ sở dữ liệu so với[6–](#_bookmark22)[7](#_bookmark24)

Elasticsearch như[12–13](#_bookmark58)

chức năng của[8](#_bookmark29)

phổ biến8-9

Tìm kiếm mở của Amazon[9](#_bookmark36)

Apache Solr[8–](#_bookmark33)[9](#_bookmark35)

Tìm kiếm đàn hồi[8](#_bookmark32)

truy vấn tìm kiếm[135,](#_bookmark616)[138,](#_bookmark626)[144,](#_bookmark643)[165,](#_bookmark738)[317,](#_bookmark1478)[352](#_bookmark1651)

thuộc tính search\_after[299](#_bookmark1394)

tính năng tìm kiếm sau[299](#_bookmark1387)

thuộc tính search\_analyzer[265](#_bookmark1206)

thuộc tính search\_analzyer[264](#_bookmark1201)

kiểu dữ liệu search\_as\_you\_type[131,](#_bookmark594)[14](#_bookmark645)4–146

tìm kiếm theo kiểu bạn[116,](#_bookmark524)[120](#_bookmark543)

phương thức search()[538–539](#_bookmark2456)

đối tượng searchResponse[539](#_bookmark2463) bảo vệ

như là lĩnh vực ứng dụng cốt lõi đàn hồi[11](#_bookmark47) phân tích bảo mật và mối đe dọa và gian lận

phát hiện[13](#_bookmark64)

công cụ quản lý sự kiện và thông tin bảo mật (SIEM)[11](#_bookmark49)

Sự khác biệt về serial (serial\_diff)[461](#_bookmark2119) Giấy phép công cộng phía máy chủ (SSPL)[8](#_bookmark34) bộ xử lý thiết lập[5](#_bookmark2395)25–526

cài đặt[195](#_bookmark896)

thành phần cài đặt[247](#_bookmark1127)

đối tượng cài đặt[199,](#_bookmark913)[255,](#_bookmark1161)[479](#_bookmark2194) kiểu dữ liệu hình dạng [412](#_bookmark1895)

truy vấn hình dạng[398,](#_bookmark1825)[412–41](#_bookmark1892)[4](#_bookmark1902)

loại hình dạng[412](#_bookmark1896)

mảnh vỡ[74–81](#_bookmark329)

phân phối của[76–79](#_bookmark332)

cân bằng lại[79](#_bookmark337)–80

kích thước[80–81,](#_bookmark343)[475–477](#_bookmark2173)

thiết lập nhiều chỉ mục[476–477](#_bookmark2178) thiết lập chỉ mục đơn[47](#_bookmark2175)5–476

chưa được giao[50](#_bookmark2333)7–509

bộ lọc ván lợp[2](#_bookmark1267)75–276

nên mệnh đề [51,](#_bookmark234)[373–378](#_bookmark1740)

tăng điểm[374–377](#_bookmark1744)

điều khoản lọc[37](#_bookmark1752)8–381

thiết lập minimum\_should\_match[377](#_bookmark1747)[–378](#_bookmark1750)

nên truy vấn[51,](#_bookmark236)[374](#_bookmark1743)

tập hợp anh chị em[457–4](#_bookmark2093)[58](#_bookmark2094)

Công cụ SIEM (quản lý thông tin và sự kiện bảo mật)[11](#_bookmark49)

thuật toán tương tự[94](#_bookmark409)

mô-đun tương tự[90–91](#_bookmark390)

máy phân tích đơn giản[236,](#_bookmark1065)[242–243,](#_bookmark1112)[251–25](#_bookmark1138)[2,](#_bookmark1143)[265](#_bookmark1204)

truy vấn simple\_query\_string[357–359](#_bookmark1679)

API tài liệu đơn [162](#_bookmark716)[–163](#_bookmark724)

cụm nút đơn [75,](#_bookmark331)[81,](#_bookmark348)[469](#_bookmark2148)

Tìm kiếm trang web[13](#_bookmark62)

**MỤC LỤC**

thuộc tính kích thước[289,](#_bookmark1321)[293,](#_bookmark1356)[298–](#_bookmark1386)[299](#_bookmark1391)

tham số kích thước[2](#_bookmark1385)[98–299,](#_bookmark1392)[437,](#_bookmark2002)[454](#_bookmark2080)

SLM (quản lý vòng đời ảnh chụp nhanh)[484](#_bookmark2217) thực hiện thủ công[487](#_bookmark2224)

sử dụng Kibana[48](#_bookmark2220)5–487

thuộc tính dốc[346,](#_bookmark1612)[418](#_bookmark1918)

từ khóa dốc[347](#_bookmark1619)

tham số dốc[44,](#_bookmark190)[345–346,](#_bookmark1614)[356](#_bookmark1672)

ảnh chụp nhanh[477–4](#_bookmark2184)[87](#_bookmark2226)

tự động hóa[483–](#_bookmark2215)[487](#_bookmark2227)

thực hiện thủ công[487](#_bookmark2224)

sử dụng Kibana[48](#_bookmark2220)5–487

tạo ra[481–48](#_bookmark2201)[2](#_bookmark2206)

xóa bỏ[483](#_bookmark2212)

Tổng quan[478–47](#_bookmark2186)[9](#_bookmark2188)

đăng ký kho lưu trữ ảnh chụp nhanh[479–4](#_bookmark2190)[81](#_bookmark2198) là Kibana[480–4](#_bookmark2197)[81](#_bookmark2199)

sử dụng API chụp nhanh[479–48](#_bookmark2192)[0](#_bookmark2196) khôi phục[482](#_bookmark2208)

Thẻ phần mềm[51](#_bookmark235)

ổ đĩa thể rắn (SSD)[498](#_bookmark2296) thuộc tính sắp xếp[324](#_bookmark1511)

sắp xếp đối tượng[304](#_bookmark1421)

phần tử nguồn[309](#_bookmark1436)

lọc nguồn[309–](#_bookmark1437)[310](#_bookmark1445)

khóa nguồn[173](#_bookmark783)

truy vấn khoảng[398,](#_bookmark1826)[414–421](#_bookmark1904)

dữ liệu mẫu[415](#_bookmark1906)

truy vấn span\_first[416–417](#_bookmark1911)

truy vấn span\_near[417–418](#_bookmark1917)

truy vấn span\_or[4](#_bookmark1928)20–421

truy vấn span\_within[418–420](#_bookmark1923) truy vấn span\_container span[421](#_bookmark1932) truy vấn span\_first [416–417](#_bookmark1911) truy vấn span\_multi\_term[421](#_bookmark1933) truy vấn span\_near[417–418](#_bookmark1917) span\_not truy vấn span [421](#_bookmark1931) truy vấn span\_or[4](#_bookmark1928)20–421

truy vấn span\_term[418](#_bookmark1919)

truy vấn span\_term[416](#_bookmark1912)

truy vấn span\_within[418–42](#_bookmark1923)[0](#_bookmark1926)

lỗi chính tả[44–45](#_bookmark192)

API chia tách[220](#_bookmark1014)

vấn đề não chia đôi[493–49](#_bookmark2280)[4](#_bookmark2282) SSD (ổ đĩa thể rắn)[498](#_bookmark2296)

SSPL (Giấy phép công cộng phía máy chủ)[8](#_bookmark34)

máy phân tích chuẩn[93,](#_bookmark402)[118,](#_bookmark539)[236,](#_bookmark1066)[241–25](#_bookmark1099)[0,](#_bookmark1136)[270,](#_bookmark1238)[342](#_bookmark1589)

cấu hình[247](#_bookmark1126)

cấu hình các từ dừng[247](#_bookmark1128)[–249](#_bookmark1131) cấu hình độ dài mã thông báo[250](#_bookmark1135) từ dừng dựa trên tệp[249–2](#_bookmark1132)[50](#_bookmark1134) thử nghiệm[246–247](#_bookmark1121)

bộ mã hóa chuẩn[240,](#_bookmark1091)[253,](#_bookmark1152)[259–260,](#_bookmark1180)[270–](#_bookmark1236)[271](#_bookmark1242)

độ lệch chuẩn thống kê[443](#_bookmark2037)

máy phân tích tiếng Hindi standard\_with\_stopwords[248](#_bookmark1130)

thiết lập tĩnh[197](#_bookmark905)

**MỤC LỤC 557**

tổng hợp số liệu thống kê[437,](#_bookmark2006)[442](#_bookmark2033) Thùng thống kê (stats\_bucket)[462](#_bookmark2128) số liệu thống kê[55,](#_bookmark260)[442–443](#_bookmark2032)

truy vấn thống kê[56](#_bookmark261)

thuộc tính stem\_exclusion[257](#_bookmark1168)

tham số stem\_exclusion[257](#_bookmark1169)

bộ lọc phiếu bầu[275](#_bookmark1265)

nhét đầy[239,](#_bookmark1086)[257](#_bookmark1167)

dừng phân tích[236,](#_bookmark1067)[243](#_bookmark1113)

dừng bộ lọc[253–](#_bookmark1153)[254](#_bookmark1159) dừng bộ lọc mã thông báo[246](#_bookmark1124) dừng từ

cấu hình[24](#_bookmark1128)7–249

dựa trên tập tin[249–250](#_bookmark1132)

tham số stopwords[93](#_bookmark403)

thuộc tính stopwords\_path[249](#_bookmark1133)

định dạng strict\_date\_optional\_time[106](#_bookmark457)

dữ liệu có cấu trúc[4](#_bookmark14)

tìm kiếm có cấu trúc[281](#_bookmark1282)

tài liệu sinh viên[68,](#_bookmark305)[107](#_bookmark463)

tổng hợp phụ[450](#_bookmark2067)

Định dạng chủ đề.kw[147](#_bookmark658)

thuộc tính thành công[217](#_bookmark1000)

tổng hợp tổng hợp[53,](#_bookmark248)[436,](#_bookmark1988)[438](#_bookmark2008) Thùng tổng (sum\_bucket)[462](#_bookmark2129) hàm tổng[435](#_bookmark1975)

tổng số mét[54](#_bookmark256)[–55,](#_bookmark257)[436,](#_bookmark1994)[441](#_bookmark2024)

chế độ tổng hợp[395](#_bookmark1817)

thống kê tổng\_bình\_bình\_thượng[443](#_bookmark2038)

bộ lọc đồng nghĩa[276–](#_bookmark1270)[278](#_bookmark1276)

loại từ đồng nghĩa[277](#_bookmark1271)

danh sách từ đồng nghĩa[277](#_bookmark1272)

biến synonyms\_path[277](#_bookmark1273)

tóm tắt đối tượng[43](#_bookmark185)

**T**

mẫu, cho các chỉ mục[212](#_bookmark975)[–216](#_bookmark992)

tạo mẫu thành phần[215–2](#_bookmark988)[16](#_bookmark993) tạo mẫu có thể cấu hình[213–2](#_bookmark980)[15](#_bookmark986)

thuộc tính thuật ngữ[182](#_bookmark845)

truy vấn thuật ngữ[46–](#_bookmark205)[47,](#_bookmark206)[122,](#_bookmark546)[128,](#_bookmark567)[135–136,](#_bookmark620)[316–318,](#_bookmark1468)

[371,](#_bookmark1731)[390,](#_bookmark1794)[395,](#_bookmark1810)[416](#_bookmark1913)

ví dụ về[317](#_bookmark1477)

trên các trường văn bản[3](#_bookmark1471)16–317rút ngắn[318](#_bookmark1480)

truy vấn cấp độ thuật ngữ[45,](#_bookmark197)[281,](#_bookmark1284)[315](#_bookmark1462)

tìm kiếm theo cấp độ thuật ngữ[45–47,](#_bookmark212)[313–333](#_bookmark1454)

ví dụ về [31](#_bookmark1460)4–315

tồn tại truy vấn[32](#_bookmark1502)2–323

truy vấn mơ hồ[3](#_bookmark1538)30–332

truy vấn id[321–322](#_bookmark1498)

không được phân tích[314](#_bookmark1458)

Tổng quan[314–3](#_bookmark1456)[15](#_bookmark1466)

truy vấn tiền tố [32](#_bookmark1525)7–330

rút ngắn[328](#_bookmark1528)[–329](#_bookmark1529)

tăng tốc[329](#_bookmark1531)[–330](#_bookmark1536)

truy vấn phạm vi[47,](#_bookmark208)[323–](#_bookmark1506)[325](#_bookmark1515)

truy vấn thuật ngữ[46–4](#_bookmark205)[7,](#_bookmark206)[316–318](#_bookmark1468)

ví dụ về[317](#_bookmark1477)

trên các trường văn bản[31](#_bookmark1471)6–317rút ngắn[318](#_bookmark1480)

truy vấn thuật ngữ[318](#_bookmark1483)[–321](#_bookmark1495)

ví dụ về[319–320](#_bookmark1489)

truy vấn tra cứu thuật ngữ[320–32](#_bookmark1494)[1](#_bookmark1496) truy vấn ký tự đại diện[3](#_bookmark1517)25–327

thuật ngữ tổng hợp [453–454](#_bookmark2079)

mảng thuật ngữ[319](#_bookmark1486)

truy vấn tra cứu thuật ngữ[319](#_bookmark1487) điều khoản đối tượng[319](#_bookmark1484)

truy vấn thuật ngữ[318](#_bookmark1483)[–322,](#_bookmark1500)[370–3](#_bookmark1729)[71](#_bookmark1730)

ví dụ về[319–320](#_bookmark1489)

truy vấn tra cứu thuật ngữ[320–32](#_bookmark1494)[1](#_bookmark1496) phân tích văn bản[64,](#_bookmark289)[235–](#_bookmark1063)[279](#_bookmark1277)

máy phân tích[238–24](#_bookmark1077)[3](#_bookmark1107)

giải phẫu của[239–241](#_bookmark1088)

được xây dựng sẵn [243–258](#_bookmark1111)

phong tục[258–261](#_bookmark1175)

chuẩn hóa[239](#_bookmark1085)

Tổng quan[238](#_bookmark1074)

chỉ định[261–2](#_bookmark1189)[65](#_bookmark1207)

thử nghiệm[241–243](#_bookmark1094)

phân chia[238](#_bookmark1080)[–239](#_bookmark1083)

bộ lọc ký tự[265–](#_bookmark1210)[270](#_bookmark1231)

bộ lọc hmtl\_strip [266–](#_bookmark1215)[267](#_bookmark1220) bộ lọc ký tự ánh xạ[26](#_bookmark1222)7–268ánh xạ qua tập tin[268–269](#_bookmark1227)

bộ lọc ký tự pattern\_replace[269–27](#_bookmark1230)[0](#_bookmark1232) Tổng quan[236–23](#_bookmark1070)[8](#_bookmark1075)

truy vấn dữ liệu phi cấu trúc[237](#_bookmark1072) bộ lọc mã thông báo[274](#_bookmark1260)[–278](#_bookmark1275)

bộ lọc ván lợp[275](#_bookmark1267)[–276](#_bookmark1268)

bộ lọc phiếu bầu[275](#_bookmark1265)

bộ lọc đồng nghĩa[2](#_bookmark1270)76–278

bộ phân tích[270–2](#_bookmark1234)[74](#_bookmark1258)

trình phân tích edge\_ngram[273](#_bookmark1249)

mã thông báo ngram[272–273](#_bookmark1245)

bộ mã hóa chuẩn[270–271](#_bookmark1236)

kiểu dữ liệu văn bản[70,](#_bookmark315)[103–105,](#_bookmark452)[109,](#_bookmark468)[115–1](#_bookmark510)[22,](#_bookmark547)[129,](#_bookmark578)

[134,](#_bookmark609)[140,](#_bookmark630)[146–14](#_bookmark651)[7,](#_bookmark656)[401,](#_bookmark1843)[431,](#_bookmark1965)[500](#_bookmark2304)

phân tích các trường văn bản [118–12](#_bookmark538)[0](#_bookmark541) lựa chọn các loại từ khóa qua[501–502](#_bookmark2309) kiểu dữ liệu token\_count[120–122](#_bookmark545)

TF (tần suất thuật ngữ)[90,](#_bookmark395)[92,](#_bookmark400)[302](#_bookmark1407)

TF-IDF (Tần suất thuật ngữ-Tần suất tài liệu nghịch đảo)[41,](#_bookmark174)[8](#_bookmark388)9–90

thuộc tính thread\_pool [505](#_bookmark2322)

phát hiện mối đe dọa[13](#_bookmark64)

chiến lược ba mảnh và 15 bản sao cho mỗi mảnh [83](#_bookmark351)

thuộc tính tie\_breaker[389](#_bookmark1790)

tham số tie\_breaker [350](#_bookmark1642)

dữ liệu chuỗi thời gian[73–](#_bookmark326)[74](#_bookmark327)

**558**

thuộc tính timed\_out[291](#_bookmark1326)

thuộc tính tiêu đề[292](#_bookmark1344)

tiền tố tiêu đề[308](#_bookmark1432)

quyền sở hữu tiêu đề[22,](#_bookmark90)[319,](#_bookmark1491)[330](#_bookmark1533)

để gán cho[452](#_bookmark2074)

bộ lọc mã thông báo[27](#_bookmark1260)4–278

bộ lọc ván lợp[2](#_bookmark1267)75–276

bộ lọc phiếu bầu[275](#_bookmark1265)

bộ lọc đồng nghĩa[276–](#_bookmark1270)[278](#_bookmark1276)

kiểu dữ liệu token\_count[116,](#_bookmark525)[120–122](#_bookmark544)

phân chia[64,](#_bookmark291)[238–239](#_bookmark1078)

bộ phân tích[270](#_bookmark1234)[–274](#_bookmark1258)

trình phân tích edge\_ngram[273](#_bookmark1249)

mã thông báo ngram[272–273](#_bookmark1245)

bộ mã hóa chuẩn[270–27](#_bookmark1236)[1](#_bookmark1242)

các mã thông báo[64,](#_bookmark292)[238](#_bookmark1081)

đã lấy thuộc tính [291](#_bookmark1327)

thuộc tính top\_left[407–408](#_bookmark1874)

toạ độ top\_left[405–4](#_bookmark1868)[06](#_bookmark1870)

điểm địa lý trên cùng bên trái[132](#_bookmark602)

tham số top\_left[407](#_bookmark1872)

chỉ mục top\_secret\_files[323](#_bookmark1503)

tài liệu top100\_movies[165](#_bookmark731)

tổng thuộc tính[217](#_bookmark1001)

tổng số xô[21](#_bookmark1003)7–218Mức ghi nhật ký TRACE[490](#_bookmark2243) thuộc tính track\_scores[306](#_bookmark1424)

chỉ số giao dịch[63](#_bookmark287)

chỉ số đào tạo[129](#_bookmark579) Chuyển đổi vai trò nút[84](#_bookmark362) tài sản tạm thời[196](#_bookmark901)

tiêu cực thực sự[337](#_bookmark1559)

những điều tích cực thực sự[336](#_bookmark1554)

loại thuộc tính[403](#_bookmark1853)

loại ép buộc[114–](#_bookmark502)[115](#_bookmark503)

**MỤC LỤC**

thêm nhiều trường[174–175](#_bookmark794) thêm các trường mới[174](#_bookmark791)

giải phẫu của các tập lệnh[175–176](#_bookmark798) truyền dữ liệu vào các tập lệnh[176–177](#_bookmark801)

xóa các phần tử khỏi mảng[173–17](#_bookmark786)[4](#_bookmark790) xóa các trường[174](#_bookmark793)

ngôn ngữ kịch bản [177](#_bookmark806) cập nhật mảng bằng cách sử dụng tập lệnh[173](#_bookmark784)

hoạt động update\_by\_query[180–181](#_bookmark826) upserts

Tổng quan[178–17](#_bookmark816)[9](#_bookmark818)

cập nhật như[179–1](#_bookmark820)[80](#_bookmark824)

bộ lọc chữ hoa[104,](#_bookmark447)[243,](#_bookmark1106)[274](#_bookmark1263)

bộ xử lý chữ hoa[526](#_bookmark2400) bộ lọc mã thông báo chữ hoa[260](#_bookmark1181)

Tổng quan về hoạt động Upsert (cập nhật và chèn)[178–17](#_bookmark816)[9](#_bookmark818)

cập nhật như[179–1](#_bookmark820)[80](#_bookmark824)

Tìm kiếm yêu cầu URI[291–2](#_bookmark1338)[94](#_bookmark1361) các thông số bổ sung[293–294](#_bookmark1353) tìm kiếm phim theo tiêu đề[292](#_bookmark1342) tìm kiếm những bộ phim cụ thể[293](#_bookmark1347) hỗ trợ với Query DSL[294](#_bookmark1359)

UUID (mã định danh duy nhất toàn cầu)[151](#_bookmark671)

**V.**

tổng hợp value\_count[439](#_bookmark2013)

số liệu value\_count[438–44](#_bookmark2012)[0](#_bookmark2017) tổng hợp số liệu value\_count[439](#_bookmark2015) thống kê phương sai[443](#_bookmark2039)

thuộc tính phiên bản[216](#_bookmark991)

tỷ lệ theo chiều dọc[9](#_bookmark422)[7–98,](#_bookmark424)[477](#_bookmark2179)

**TRONG**

mã thông báo uax\_url\_email[274](#_bookmark1253)

tổng hợp unique\_tvs[444](#_bookmark2043) dữ liệu không có cấu trúc (toàn văn)

Tổng quan4–5

truy vấn [237](#_bookmark1072)

tìm kiếm không có cấu trúc[281](#_bookmark1283)

văn bản không có cấu trúc[118](#_bookmark537)

cập nhật hành động[185,](#_bookmark859)[189](#_bookmark871)

Phương pháp CẬP NHẬT[171](#_bookmark766)

hoạt động cập nhật[40](#_bookmark165)

đang cập nhật tài liệu[170–181](#_bookmark760)

\_cập nhật API [1](#_bookmark768)71–173thêm các trường mới[171–172](#_bookmark770)

sửa đổi các trường hiện có[1](#_bookmark773)72–173cơ học của[1](#_bookmark762)70–171

thay thế tài liệu[177–](#_bookmark810)[178](#_bookmark814)

cập nhật theo kịch bản[173–177](#_bookmark779)

thêm các tập lệnh cập nhật có điều kiện [175](#_bookmark796)

**TRONG**

chức năng trọng lượng[394](#_bookmark1809)

bộ phân tích khoảng trắng[236,](#_bookmark1068)[243,](#_bookmark1114)[251–252](#_bookmark1141)

bộ phân tích khoảng trắng[238,](#_bookmark1082)[274](#_bookmark1256) kiểu dữ liệu đại diện[124](#_bookmark557) truy vấn ký tự đại diện[3](#_bookmark1517)25–327Cửa sổ

cài đặt Elasticsearch[516–](#_bookmark2360)[518](#_bookmark2363)

cài đặt Kibana[521](#_bookmark2376)

trong mối quan hệ [411](#_bookmark1887)

Điểm WKT (văn bản nổi tiếng)[402](#_bookmark1848)

**X**

Thuộc tính Xms [80](#_bookmark340)

-Cài đặt Xms [491](#_bookmark2255)

Thuộc tính Xmx[80](#_bookmark341)

-Cài đặt Xmx[491,](#_bookmark2256)[499](#_bookmark2298)

TÌM KIẾM / KỸ THUẬT PHẦN MỀM

Tìm kiếm đàn hồiTRONGHÀNH ĐỘNG Phiên bản thứ hai

Madhusudhan Konda●Lời nói đầu củaShay Banon

###### reate tìm kiếm hoàn toàn chuyên nghiệpcông cụ tìm kiếm Elasticsearch và Kibana! Được viết lại cho phiên bản mới nhất của Elasticsearch, cuốn sách thực tế này khám phá Elasticsearch

C

kiến trúc cấp cao, tiết lộ các mẫu cơ sở hạ tầng và hướng dẫn các khả năng tìm kiếm và phân tích củanhiều API Elasticsearch.

###### Elasticsearch trong hành động, Phiên bản thứ haihướng dẫn bạn cách thêm các tính năng tìm kiếm hiện đại vào các trang web và ứng dụng bằng Elasticsearch 8. Trong đó, bạn sẽ nhanh chóng tiến triển từ những điều cơ bản về cài đặt và cấu hình cụm, đến lập chỉ mục tài liệu, tổng hợp nâng cao và đưa máy chủ của bạn vào sản xuất. Bạn sẽ đặc biệt đánh giá cao sự kết hợp giữa chi tiết kỹ thuật với các kỹ thuật để thiết kế trải nghiệm tìm kiếm tuyệt vời.

Bên trong có gì

###### Hiểu về tìm kiếmngành kiến ​​​​trúc

* Truy vấn tìm kiếm toàn văn và cấp thuật ngữ

###### Phân tích và tổng hợp

* Hình ảnh trực quan cấp cao trong Kibana

###### Cấu hình, mở rộng và điều chỉnh cụm

Dành cho các nhà phát triển ứng dụng thoải mái với việc viết kịch bản vàứng dụng dòng lệnh.

###### Madhusudhan Kondalà một kỹ sư trưởng, kiến ​​trúc sư, cố vấn và diễn giả hội nghị toàn diện. Ông cung cấp các buổi đào tạo trực tuyếnTìm hiểu về Elasticsearch và Elastic Stack.

Đối với chủ sở hữu sách in, tất cả các định dạng sách điện tử đều miễn phí: https:/[/www.manning.com/freebook](http://www.manning.com/freebook)

Niềm đam mê của Madhuđếntrong chiều sâu và chiều rộng của cuốn sách này, giọng điệu nhiệt tình và các ví dụ thực tế. Tôi hy vọng bạn sẽ tiếp thu những gì đã đọc và đưa vào 'hành động'.

—Trích từ Lời nói đầu củaShay Banon, Nhà sáng lập Elasticsearch

“

“

### Thực tế và viết hay.

“

Một điểm khởi đầu tuyệt vời cho người mới bắt đầu và

### hướng dẫn toàn diện để biết thêmchuyên gia giàu kinh nghiệm.

“

—Simona Russo, Sự tình cờ

### Sự phấn khích của tác giả thể hiện rõ ngay từ những đoạn văn đầu tiên. Kết hợp điều đó với

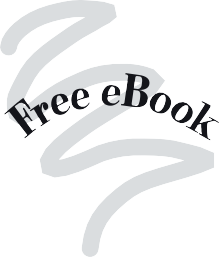
“

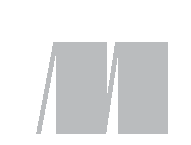
kinh nghiệm sâu rộng và trình độ kỹ thuật, và bạncó ngay một tác phẩm kinh điển.

“

—Herodotos Koukkides vàSemi Koen, Nhật Bản toàn cầuTổ chức tài chính

Xem trang đầu tiên



**NGƯỜI ĐÀN ÔNG**

**Mã số ISBN-13: 978-1-61729-985-8**