<https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-metrics-avg-aggregation.html>

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | DSL:  <https://docs.spring.io/spring-data/elasticsearch/docs/1.1.0.M1/reference/html/elasticsearch.repositories.html>    <https://www.elastic.co/blog/found-java-clients-for-elasticsearch> |  |  |
|  |  | tech primer :-elastic search |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | Download elastic search :<https://www.elastic.co/downloads/>  **Run elastic search**  Run bin/elasticsearch (or bin\elasticsearch.bat on Windows)  **To get response**( means elastic search run on this url)  <http://localhost:9200/>    Machine generated alternative text: 2  3  Run bin/elasticsearch (or binXe1asticsearch.bat on Windows)  Run curl http://10ca1host:920/ or Invoke-RestMethod http://10ca1host:92ee With PowerShell  **Kibana**  Kibana is use to interact elastic search    Machine generated alternative text: 2  3  4  •  •  Open config/kibana.yml in an editor  Set elasticsearch. hosts to point at your Elasticsearch instance  Run bin/kibana (or binXkibana . bat on Windows)  Point your browser at http://localhost:5601      Machine generated alternative text: Dist:nbutea REST"'  c h anaMics,  Oov•nuoaa  Visuah2e your data.  Navigate the  Coae•ct parse. in  a ligrgweight fashion,  Ingest. transform.    <http://localhost:5601> ( where kibana is running)    Monitoring section will tell elasticsearch is up or not)    **Note:**  Kibana can only run if elastic search is running    Elasticsearch is document oriented    **Elasticsearch** is one such NOSQL distributed **database**. ... ES (**Elasticsearch**) is a document-oriented **database**, designed to store, retrieve and manage document oriented or semi-structured data. When you use **Elasticsearch** you store data in JSON document form. Then you query them for retrieval.    Or  **Elasticsearch** is a **search** engine based on the Lucene library. It provides a distributed, multitenant-capable full-text **search** engine with an HTTP web interface and schema-free JSON documents.**Elasticsearch** is developed in Java.    Kibana  Kibana is an open source analytics and visualization platform designed to work with Elasticsearch. You use Kibana to search, view, and interact with data stored in Elasticsearch indices. You can easily perform advanced data analysis and visualize your data in a variety of charts, tables, and maps.  Kibana makes it easy to understand large volumes of data. Its simple, browser-based interface enables you to quickly create and share dynamic dashboards that display changes to Elasticsearch queries in real time.    Elasticsearch is able to achieve fast search responses        Machine generated alternative text: Elasticsearch is Document Oriented  • Insert Documents  • Delete Documents  • Retrieve Documents  • Analyze Documents  • Search Documents      Machine generated alternative text: Inverted Index  Maps words to the actual document locations of  where they occur      Machine generated alternative text: Docunvnt 1  The br$ht blue  butterfly hangs  on the  Docunv•nt 2  It'S best to  forget the great  sky and to  retire from  every wind.  DocunE•nt 3  Under blue  sky. in bright  sunlight, one  need not  search around,  Stopword list  around  e Very  ID  3  4  11  12  13  index  1.3  Term  br$ht  butterfly  forget  hangs  retire  wind    <https://www.elastic.co/guide/en/elasticsearch/guide/master/inverted-index.html>    Note: document stored in elastic search    Document================json    **Note: a document is individual entry in elastic just like row is individual entry in database**    Machine generated alternative text: Complete Elasticsearch Masterclass with Logstash and Kibana  Elasticsearch is  Document Oriented  Employees Table in a Relational Database  "tot•tØt 6.  "doc.ents": I  "salary": 4seøe.  -hired•te-: "April 2  -&partnnt": -Research"  : "Cassandra-.  -hired•te-: — January 12.  • dep. rt•ento:  2'tr,  DEPARTMENT  Admin  Sales  Research  Admin  Sales  37eae,  "hi redate-: "August  lies.  -salary": 92eøe.  19,  EMPLOYEE ID NAME  SALARY  41000  68000  45000  37000  92000  43000  HIREDATE  October 3, 1998  Janua 12, 2001  17, 2008  August 19, 2012  March 15, 2013  January 11, 2014  1101  1102  1103  1104  1105  1106  David  Cassandra  Michelle  Brian  Jason  Robert  *'arch IS  -Sales"  redate-: • January  11. 2014".    Machine generated alternative text: Complete Elasticsearch Masterclass  with Kibana and Logstash  Indexing, Retrieving and Deleting  Documents    Machine generated alternative text: How Elasticsearch relates to a database  Relational DB  Table  Row  Column  Elasticsearch 6  Index  Document  Field    Machine generated alternative text: Inserting Indexing  Index a document means inserting a document into elastic    NOTE: in elastic search 5 and before table were reffered as TYPE    Machine generated alternative text: Index Type Document Field  • Vehicles (index)  Car (document 1)  Car (document 2)  912843.  -red" .  •purchase Date•: 22.    Machine generated alternative text: PUT /OndeO/(tYpO/Ud)  "fieldl": "valuel",  . "value2",  "make": "honda",  /vehicles/car/123  "milage" :  87øøø,  "color": " red  11  PUT    Concept of type is deprecated in future version of elastic search  Note:-- type is subdivision of index    Note: each document in index better have unique identifier    In es 6 onward we can have only one type of data  ===========================================================    Index a particular document  Machine generated alternative text: "make": "Honda" ,  "Color": "Black",  "HP": 25Ø,  "milage" :  24øøø,  "price":  193øø.97    Machine generated alternative text: PUT /vehic1es/car/123  "Nike": "Honda",  "Color": "Black",  25ø,  "milage": uøøø,  "price": 193øø.9?  2  3  4  5  6  8  9  lø  11 •  12  13 •  " _ index": "vehicles",  "car  "_id": "123",  —versi&l": I,  "result  "created" ,  "—shards":  "total": 2,  "successful  "failed":  "created": true    Overwrite the existing document  Anything with underscore is meta field managed by elastic search    Note: **run the same command again**    Machine generated alternative text: IPUT /vehic1es/car/123  "rmke": "Honda" ,  "Color": "Black",  "HP": 25ø,  "milage": uøøø,  "price": 193øø.97  2  3  4  5  6  8  9  lø  11 •  12  13 •  " _ index": "vehicles",  " _ type": "car",  " id": "123",  "—version":  "result": "updated" ,  "—shards":  "total": 2,  "successful  "failed":  "created": false    Version and result field will change    **To get Document**  : GET /vehicle/car/123    Machine generated alternative text: I GET  1.  2  "_index": "vehicles",  3  " _ type": "car",  4  "_id": "123",  "_verSiOn": 4,  5  6  "found": true,  7 • V —sourer:  8  "make": "Honda" ,  9  "Color": "Black",  "HP": 25ø,  lø  11  "milage": 24ØØØ,  12  "price"; 193W.97  13 -  14    Note: want to get only source  Machine generated alternative text: GET 'vehicles/car/1Z3/—sourcel  2  3  4  5  6  7.  "nuke": "Honda",  "Color  "Black",  "HP": 25ø,  "milage": 24Øøø,  "price": 193øø.97    Note: want to check whether car exist with id 123    Machine generated alternative text: HEAD "vehicles/car/1231  pøø -10K    Head command will check whether a particular document is exist or not    **V important point:---** Document are immutable  In document we cant change specific field . It will change entirely    Note:  Machine generated alternative text: POST ,'vehic1es/car/123/_  update  "_type": "car",  2  3  4  5  6  8  9  11.  12 •  "—index": "vehicles" ,  "_version": 8,  "result": "updated" ,  "total": 2,  "successful  "failed": ø  "doc":  "driver" :    It will do the same what PUT do    **Delete Document**    Machine generated alternative text: 1  DELETE /vemc1eszcaM1231  1.  • _ type": •car" ,  8.  2  3  4  5  6  7  9  lø  11  12 •  13 •  "found": true  Index": "vehicles" ,  "_id": "123",  "_version": 9,  •result": "del*e4',  "_shards":  "successful " :  "failed":  1,      **Get Vehicle:-** Will return structure of index  Machine generated alternative text: 1 | GET /vehiclesl  9.  "type": "text",  8  10  11  12  13 •  14 •  15 •  16.  17  18.  19.  21  22.  23  24  25 -  26.  27  28.  29  30  31.  32  33  34 •  WLor• :  "type": "text",  "fields":  "keyword":  "type" : "keyword" ,  "ignore_above" : 256  "type" :  "&iver" :  "long"  "fields":  "keyword" :  "type" : "keyword" ,  "ignore_above": 256  "nuke":  "type": "text",  "fields":  "keyword":  "type" : "keyword" ,  "ignore_above": 256    **Component of an index**    Machine generated alternative text: GET /business/building/lle  2.  3  33  45  46  47  " address " : "bangalore"  "floors  " •le, .  "offices  " : 21,  " loc "  "lat" : 49.2  "business"  "aliases"  "mappings "  " settings "  GET  /business    Mapping contain structure of field    Note:elastic search is able to change index very easily    **Note :** to see index we will go to discover tab    **For searching**    Machine generated alternative text: GET busi ness/ _search  • Query • :  •tem•:  •address-  16  18  •tmk••. e.  : false,  •total • ; S,  •successful :  •failed": e  •hits":  -.n_score" :  • _type" ;  •.2876821,  •_id"; -217".  ..2876821,  • —source" : (  "address • :  •nmrs": S,  •offices":  "loc":  "lat": 40.69519,  -73.98856    **Note:** term will act as field    Machine generated alternative text: •query•:  10 •  12  • fai led" :  •hits•:  •total • : 2,  •ux_score• :  •hits": C  _ type" :  —score :  • —sa•rce":  •offices":  •loc":  40.7ø7S19,  -Ion": -74.—S6  •_type":  -_id-: -217",  ress :    Note: match\_all will give all value    **Distributed execution of request**  We can have billion of document in index    Shard:-storage of document  Replica shard  Elastic search can be run on different machine having same network    Note: data is distributed in elastic search with replica node    Machine generated alternative text: ho de-I  nocl.a_    **shard** – Because **Elasticsearch** is a distributed search engine, an index is usually split into elements known as**shards** that are distributed across multiple nodes.**Elasticsearch** automatically manages the arrangement of these **shards**. It also rebalances the **shards** as necessary, so users need not worry about the details.    Note: node1 and node2 are in 2 different machine.  Every node know the location of document in the cluster  ==========================    Imp point:- data is managed in shards    Note: replica shards should be sync with primary shards    Machine generated alternative text: Node L  AJoAe    Machine generated alternative text: Node L  c  AJOÅe  vo:a    Shards 0 is on node 2 and shards 1 is on node 1(ie primary)    Customer send request to delete document using id to node 2 it will reroute to node 1 using hashing , first it will delete from primary shard 1then it will delete from all replica to make it sync    Get:    Machine generated alternative text: Nade  14  60  1      Request can go any node and that is called round robin    Note: each node contain part of data (shards)of index    Shards:-physical container of data  Elastic search index make of multiple shards and  Shards can have multiple segment( segment belong inside shards)  And each of the segment is inverted index    Machine generated alternative text: sÅac(  (TDD    **Analysis** is the process of converting text, like the body of any email, into tokens or terms which are added to the inverted index for searching.    the act of storing data in **Elasticsearch** is called **indexing**,  Machine generated alternative text:      Document go for ES then it will conver into inverted index process called analysis  And it will store in buffer once it is full then it will move to segment for faster searching    Immutable inverted index called segment  Note: once segment is done then this shards is searchable    Analysis: convert text into inverted index    **Text analysis for indexing and searching**    analyzer  The values of [analyzed](https://www.elastic.co/guide/en/elasticsearch/reference/current/mapping-index.html) string fields are passed through an [analyzer](https://www.elastic.co/guide/en/elasticsearch/reference/current/analysis.html) to convert the string into a stream of *tokens* or *terms*. For instance, the string "The quick Brown Foxes." may, depending on which analyzer is used, be analyzed to the tokens: quick, brown, fox. These are the actual terms that are indexed for the field, which makes it possible to search efficiently for individual words *within*big blobs of text.  This analysis process needs to happen not just at index time, but also at query time: the query string needs to be passed through the same (or a similar) analyzer so that the terms that it tries to find are in the same format as those that exist in the index.    Machine generated alternative text: d Searchin  Text Anauyls  1. The thin lifeguard was swimming in the lake  2. Swimmers race with the skinny lifeguard in lake  Remove Stop Words v/"  Lowercasing  Stemming  Synonyms  2er  Token  in  swim rn*l€  Exists in  1.2  1.2  2  2  1.2  1 and 2 are two document    Note: Analyzer does the analysis in elastic search    During indexing and searching analyzer take place.  Analyzer take place in two step:-    1)Tokenizer  2)filter    Note: how inverted index look after applying filter to above sentence that is part of analyzer    Machine generated alternative text: Searching  Token  in  lake  swim  Exists in  2  1  2  I.  2.  Text Analysis  The thin lifeguard was swimming in the lake  Swimmers race with the skinny lifeguard in lake  Remove Stop Words  Lowercasing  Stemming  Synonyms    Stop word(example-the)  Lowercase(all word)  Stemming( ex- swimmer, swimming------swim)  Synonyms: thin and skinny have same meaning    Note: **elastic search have some pre built analyzer**    **Note:** every document that is going to index will go some form of analyzer    **V imp:-Analyzer should be apply both for index time and query time**    **Note: we specify analyzer on certain field**    Machine generated alternative text: Configure the Index Structure  " settings " :  "mappings " •      **Index setting and mapping**  ES index and different document go into index    Define index structure manually    Note: **inside setting we define number of shards and number of replica** etc    Note: mapping concept is no there in future version of ES.    Note: field define in the mapping section    Analyzer is useful for splitting the text the text so that it can be tokenization    "analyzer":"standard" is us for splitting the text based on whitespace    **NOTE: as part of ES 7 mapping has been removed**    Machine generated alternative text: UT /customers  "mappings " : f  "online" : f  " properties " : f  " gender"  "type" : "text " ,  "analyzer" : "standard"  " age " : f  "type" : " integer"  "total spent" : f  "type" : "float"  "is new" : f  "type" : "boolean"  "name" : (  "type" : "text " ,  "analyzer" : "standard"  " settings" : f  "number of shards": 2,  "number of _ replicas": 1      Machine generated alternative text: dynamic :  If set to false - indexing field will be Ignored  If set to  - indexing field will throw error    **How to set a dynamic field on a given mapping**    Machine generated alternative text: GET customers/_mapping/online  "dynamic": falsE I  2  "acknowledged" :  true    Now no additional field can be add dynamically    Machine generated alternative text: GET ine  "dynamic": "strict"  2  "acknowledged": true    NOTE: if we add dynamic as strict then no extra field we can add it    Machine generated alternative text: PUT /custorers/onl tne/124  2.  "name": "Mary Cranford" ,  3.  "address": "31Ø Clark Ave" ,  "gender": "fennle" ,  5  "age": 34,  6  "total _spent": 59.75,  "is_new": false,  " retired" : true  8.  9  lø  1  11 •  12  "error":  root_cause" :  "type" :  " strict_dynami c_nuppi ng_exception" ,  "reason" :  "mapping set to strict, dynamic  introduction Of tretired) within LonlineJ is not  al lowed"  "type": ,  " reason" :  "nnpping set to strict, dynænic  introduction of Cretiredl within (online) is not  al lowed"  "status" :    **elasticsearch analyzer**    Machine generated alternative text: 1  2.  "analyzer": ,  3  4  9.  POST _analyze  "text" :  "The quick: ; brown746  fox. "  4  6  7  8  11  12  13  14  15  16.  17.  18  19  2ø  21  22  23 •  24 •  25  26  27  28  29  "tokens" :  "token" "The" ,  "start_offset": ø,  "end_offset" :  3,  "type" :  "M)rd" ,  "position": ø  -token": •quick: ,  "start_offset": 4,  "end_offset " :  11,  "type": "M)rd" ,  "position": 1  •token" : ,  "start_offset": 12,  "end_offset": 20,  "type" :  "M)rd" ,  "position": 2  "token": "fox."  "start_offset": 21,  "end_offset": 25,  "type" :  ".wrd" ,  "position": 3    NOTE: **We can define custom analyzer in setting section**  **Note:** analyzer do both indexing and searching    **Querying ES**    Everything in ES is json over http    DSL: **Domain specific language**    Specific structure of language that ES understand    **NOTE :** DSL is in form of json    Machine generated alternative text: Search DSL Components:  Context  Query  Filter Context  - Both of these can also be combined  o orm more com ex queries    1)match\_all will return everything in the courses index    Machine generated alternative text: I GET /courses/_search  "query":  "match_all " :     1. **Query to match a particular field in document**     Machine generated alternative text: " _ index": "courses",  2.  "Y  4  6.  " _ index": "courses" ,  GET  / courses/_search  "query":  "match": ("name" : "corputer  14  15  16  17  18.  19  29  21.  22  23  24  25  26 •  27  28  29  30 •  31 •  32.  33  34  35  37  " _ type": "classroan" ,  "_score": ø.8142733,  "_source":  "nme"•. "Convuter Science Iøt ,  "room": "C12",  "professor":  " nmæ": "Gregg Payne",  "department " :  "engineering" ,  "facutly_type": "full-time" ,  "emai : "payne*onuni .com"  "students_enrolled": 33,  "2013-0-27" ,  "course_description": "CS IØI is a first  year CCnputer science introduction teaching  fundamental data structures and alogirthms using  " _ type" :  "classroan" ,  "_source":  cc    **Note: if score is more document is more relevant**  Whatever field we are searching if it is more in document then that document is more relevant     1. Requirement: **to find all professor having email**     Machine generated alternative text: GET / courses/_search  ": ("field":  "query":  "exists    **NOTE: exits** will check whether particular field professor.email is present in document or not     1. **Requirement: want to search with two field in document**     **Solution :** that should be used inside must which will be written inside bool    Machine generated alternative text: GET 'courses/_search  8.  9.  3  5  6  7  lø  11  12.  13.  14  15  16  17  18.  19  2ø  21.  22  23  24  25  26  "timed_out": false,  "_shards":  "total": 5,  " successful":  s,  "failed": Ø  "hils":  "total": I,  "max_score" :  1.506144,  "hits":  " _ index": "courses",  "classroan" ,  _type" :  "_score": 1.Sø86144,  "_source":  "rune" : "Computer cc—uter 29" ,  "room":  "professor":  "nmæ": "Gregg Payne",  "department" : "engineering" ,  "facutly_type" : "part-time" ,  "email" : "payne*onuni .com"  "students_enrolled": 33,  "query":  "bool":  " rwst .  ("match":  "name" :  ("match": l"  room":     1. **Must\_not**     Machine generated alternative text: GET 'courses/ _ search  ("room":  8.  9.  2  3  5  6  lø  11  12.  13.  14  15  16  17  18.  19  2ø  21.  22  23  24  25  26 •  27  28  29  "query":  "boot":  "must" : C  ('•rratch" :  ("match" :  ("name": "accounting"))  "bill"))  "took": Ø,  "timed_out": false,  "_shards":  "total": 5,  " successful " :  5,  "failed":  "hits":  "total": p,  "max_score" .  • 1.379555,  "hits":  " _ index": "courses" ,  " _type" :  "classroan" ,  "_iC1 ,  "_score": 1.379555,  "_source":  "nane": "Accounting 101" ,  "room": "E3",  "professor" :  " nar€': "Thomas Baszo" ,  "department " :  " finance" ,  "facutly_type": "part-time" ,  "email": "baszot@onuni.com"  "students_enrolled": 27,  "course_publ ish_date": "2ø1S-ø1-19" ,  "course_description": "Act IØI is a cours  ness school on the introduction to  "must_not" : C  Æ"rwtch": I "professor.nane     1. **Should:---** it is good to have     Machine generated alternative text: GET /courses/_search  "query":  "boot":  "must" : C  ("rnatch" :  ("nntch" :  ("room":  "must_not .  ("ratch" :  "should  34 •  ("name": "accounting"))  11  12.  13.  14  15  16  17  18.  19  20  21.  22  23  24  25  26  27  28  29  31.  32 •  33 •  "hits":  "total": 1,  "max_score" :  1.379555,  "hits":  " _ index": "courses"  " _ type": "classroan" ,  "Ad":  "1",  "-score": 1.379555,  "_source":  "narne": "Accounting IØI",  "rom'": "E3",  "professor" :  "name": "Thomas Baszo" ,  "department " :  " finance" ,  "facutly_type": "part-time" ,  : "baszot@onuni.com"  "studend_enrolled": 27,  "course_publ ish_date": "201501-19" ,  "course_description": "Act løl is a cours  fran the business school on the introduction to  accounting that teaches students how to read and  conpose basic financial statements"  "bill "D  ("professor. name" :  ("match": ("name":  " computer "D    Same result will come:-  **Reason :** should will not take precedence over must and must\_not    **Note: if we want should will work then we need to use minimum\_should\_match:1 inside bool (important)**    Machine generated alternative text: GET /courses/_search  9.  "query":  "bool":  "must" : C  ("nxltch": ("name": "accounting"))  3,  11  12.  13.  14  15  16  18.  19  2ø  21.  22  23  24  25  26 •  27  28  29  31.  32 •  33 •  rat Lea .  "hits":  "total " :  1,  "max_score" :  1.8139423 ,  "hits": t  "_index": "courses" ,  "classroan" ,  _type" :  "_score": 1.8139423,  "_source":  ax Accounting 2ØØ" ,  "professor":  "nuæ": "Thomas Baszo" ,  "department " :  " fi nonce" ,  "facutly_type" : "part-time" ,  "email" : "baszot@onuni .com"  "students_enrolled": 17,  "course_publ ish_date": "2Ø16-ø6-15" ,  "course_description": "Tax Act 2øø is an  intermediate course covering various aspects of tax  law"  "must_not" :  C "rmtch": ("professor.name •  "should" : C  ("match": ("room": n  "minimun-should-match": 1  bill "h    **7)multi\_match**    **Field:** which field we want to search  **Query**: what we want to search in above field    Machine generated alternative text: GET 'courses/ _ search  "query":  "multi_match" :  8.  9.  2  3  5  6  7  10  11  12.  13 •  14  15  16  17  18.  19  2ø  21.  22  23  24  25  26 •  27  28  29  "took": Ø,  "timed_out": false,  "_shards" :  " successful":  5,  "failed":  Y,  "hits":  "total": 4,  "max_score" :  1. zø39728,  "hits":  " _ index": "courses",  "classroan" ,  " _ type" :  "_score": 1.209728,  " _ source":  : Info Systems 39" ,  "room": "E3",  "professor" .  "nææ": "Bill Cage",  "department": "accounting" ,  "facutly_type": "full-time" ,  "email" : "cageb@onuni.cm"  "students_enrolled": 19,  "course_publish_date": "2014-05-15" ,  "course_description": "Act Sys 35ø is a  e Drovlduna  " fields " :  "query" :  C"åame" ,  "profesbor• . department "J  "accounting"    **NOTE: query: accounting doesn't need to be present in both field**    **8)match\_phrase :**  Going to match a sentence or part of sentence    Machine generated alternative text: GET /courses/_search  "query":  "rntch_phrase" :  "course_description":  taken by final year "l  " _ index": "courses" ,  f rcyn  the  business  school  26 •  27  28  29  31 •  32.  33  34  35  36  37  38  39  41  42  43  45 •  46  47  48  "students_enrolled": 19,  "2Ø14-Ø5-15" ,  "course_description": "Act Sys 35ø is an  advanced course providing students a practi cal  understanding Of an accounting system in database  technology. Students will use MS Access to build a  transaction ledger system"  _type" :  "classrocn" ,  " _score" :  .98ø8292 ,  " -source" :  "nane" :  "Cost Accounting 4ØØ" ,  "professor" :  " ncn€': "Bill Cage",  "department" : "accounting" ,  "facutly_type": "full-time" ,  "emai l": "cageb@onuni .  "students_enrolled": 31,  "course_publish_date": "2ø14-12-31" ,  "course_description": "Cst Act 4Øø is an  advanced course from the business school taken by  final year accounting nnjors that covers the subjec  of husi ness incurred costs and how to record them i    **NOTE:** partial token can be match by using match\_phrase\_prefix    Machine generated alternative text: GET / courses/_search  "query":  "course_description" :  taken by firi•  " f rcyn  the  business  school  10  11  12.  13.  14  15  16  17  18.  19  21 •  22  23  24  25  26 •  27  28  29  31.  32 •  33 •  nits :  "total": I,  "max_score" :  4.6659784,  "hits":  Index" • courses" ,  _ type" :  "classroan" ,  "_id":  "_score": 4.6659784,  "-source":  "nane": "Cost Accounting 4ØØ" ,  "room": •'E7",  "professor":  "nmæ": "Bill Cage",  "department" : "accounting" ,  "facutly_type": "full-time" ,  "emai l": "cageb@onuni , cml"  "students_enrolled": 31,  "course_publ ish_date": "2Ø14-12-31" ,  "course_description": "Cst Act 4øø is an  advanced course frorn the busi ness school taken by  final year accounting nnjors that kovers the subjec  of business incurred costs and how to record them  financial statenænts"     1. Range for a particular field     Machine generated alternative text: GET courses/  search  "query"  " range '  "students enrolled":  "gte" .  "Ite": 2B  Gte-- >=    Machine generated alternative text: GET /courses/_search  "query":  "range" : I  " students_enrol led" :  "gt": 19,  "It": 271      Note: we can give range on date also    Machine generated alternative text: GET /courses/_search  "query":  "range" : I  " course_publ i sh_date" :  "gt": "2ø13-ø8-27"l    **Combine all query together**    Machine generated alternative text: 9.  5.  6  8.  9  11.  12.  14  15  16.  18.  19.  2ø  21.  22.  23 •  GET  / courses/ _ search  "query":  "must" : t  ("match" :  "must_not .  ("match .  1,  "should .  ("nane": "accounting"))  ("range" : (  "students-enrolled .  "gte": Iø,  "Ite": Zø  7  11  12.  13.  14  15  16  17  18.  19  21 •  22  23  24  25  26 •  27  28  29  "fai led" :  "hits":  "total " :  2,  "max—score" :  1.609695 ,  "hits":  e _ index": "courses" ,  "_type" :  "classroan" ,  "_id":  "8",  "_score": 1.6Ø99695,  "_source":  "nane": "Accounting Info Systems 39" ,  "room": "E3",  "professor":  "ncnæ": "Bill Cage",  "department" : "accounting" ,  "facutly_type": "full-time" ,  "emai l": "cageb@onuni.cm'"  "students_enrolled": 19,  "course_publ ish_date": "2Ø14-Ø5-15" ,  "course_description": "Act Sys 35ø is an  advanced course providing students a practi cal  understanding of an accounting system in database  1      **Search DSL Filter Context**    **Note :** We need to put filter inside query and also we need to put bool around it    **Note:** in filter scoring of document is 0    Machine generated alternative text: GET 'courses/ _ search  " _ index": "courses" ,  "query":  "bool":  "filter :  "match" : ("name": "accounting  " _ index": "courses" ,  53  54  55  56 •  57  58  59 •  61  62  63  64 •  65  66  67  68  69 •  71  72  73  74  75.  76  " -type  "classroan" ,  "_score": ø,  "_source":  "nane": "Cost Accounting 4øø",  "rocyn": "E7",  "professor":  "nme"•. "Bill Cage",  "department" : "accounting" ,  "facutly_type": "full-time" ,  "cageb@onuni . cm"  "students_enrolled": 31,  "course_publ ish_date": "2Ø14-12-31" ,  "course_description": "Cst Act 400 is an  advanced Course f rcyn the business school taken by  final year accounting nnjors that covers the subje  of busi ness incurred costs and hcm to record them  financial statements "  " _ type":  "classroan" ,  "_id":  "_score": q,  "_source":  "name": "Accounting IØI" ,    NOTE: filter are cached so searching is faster    Query: little bit take time to find out which document is more relevant.  Filter doesn’t take care of this    **NOTE :** if we don’t care which document is more relevant then always use filter    Machine generated alternative text: 5.  6.  8  8.  9  9.  11 •  12  " _ index": "courses" ,  14  16.  18 •  19  20 •  GET 'courses/_search  "query":  "bool":  n: bill")),  2  3  5  6  7  lø  11  12.  13.  14  15  16  17  18.  19  2ø  21.  22  23  24  25  26 •  27  28  29  "took": ø,  "timed_out": false,  "_shards":  "total": 5,  " successful " :  "failed": Ø  "hits":  "total":  5,  "must" : C  "nutch" : ("professor . name  "must—not .  ("r.atch" :  ("match" : ("name": "accounting  ( " rocyn":  "max_score": Ø,  "hits": t  "classroan" ,  " _ type" :  "Ad" :  "_score": ø,  " _source":  "nane": "Accounting Info Systems 39" ,  "rocn": "E3",  "professor":  "næne"•. "Bill Cage",  "department" :  "accounting" ,  "facutly_type": "full-time",  "cageb@onuni.cm"  "students_enrolled": 19,  "2ø14-ø5-1S" ,  "course_description": "Act Sys 35ø is an  course providing students a practical  advanced    **NOTE:** all document are equally relevant when it come to filter    **NOTE**: in query context score will calculate but in filter context score will not calculate    Machine generated alternative text: GET / courses/ _ search  "query":  "bool":  9.  n: bill")),  5  6  lø  11  12.  13.  14  15  16  18.  19  2ø  21.  22  23  24  25  26 •  28  29  "_shards":  "total " :  5,  " successful "  "failed":  "hits":  "total": I,  "hits":  " _ index" :  " _ type" •  s,  ø.6931472,  "courses" ,  "classroan" ,  "must" : C  "nutch" : ("professor . name  "must " : t  ("rnatch" : ("name": "accounting  " : ("room  "-id":  "_score": ø.6931472,  "_source":  "name": "Accounting Info Systems 350" ,  •room": "E*' ,  "professor":  "name": "Bill Cage",  "department" : "accounting" ,  " facutly_type" :  "full-time" ,  erxlil"• "cageb@onuni.  "students_enrolled": 19,  "course_publ ish_date": "2ø14-ø5-1S" ,  "course_description": "Act Sys 35Ø is an  advanced course providing students a practical  understanding of an accounting system in database  technology. Students will use MS Access to build a    **NOTE: here must is a party of query context that’s why score will calculate**    Machine generated alternative text: GET 'courses/_search  "query":  "bool":  "f i Iter"  "must .  " range" :  "students_enrolled .  "gte": 191  "must .  2  3  5  6  7  8.  9.  11  12.  13.  14  15  16  17  18.  19  2ø  21.  22  23  24  25  28  "took": 1,  "timed_out": false,  "_shards":  "total": 5,  " successful " :  "failed": Ø  5,  "hits":  rwx_score" .  • ø.6931472,  "hits": t  " _ index": "  course  " _ type":  "classroan" ,  "_id":  "_score": ø.6931472,  "_source":  "nme": "Accounting Info Systems 39" ,  "rocn": "E3",  "professor":  "næne": "Bill Cage" ,  "department " :  "accounting" ,  "facutly_type": "full-time",  "cageb@onuni.ccn"  "students—enrol led" :  14-  room":  ("n•atch" :    **NOTE:** anything in query context have relevance score but in filter context don’t have relevance score    Field boosting to get more relevance    Machine generated alternative text: "rulti_match":  "query" : "market" ,  ", "course_descriptionA2'ÄD  "fields": "name      NOTE: filter context is always faster then query context    **Reference for bulk**    [***https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-bulk.html***](https://www.elastic.co/guide/en/elasticsearch/reference/current/docs-bulk.html)  Machine generated alternative text: POST  POST / vehicles/ cars/ bulk  2B", "  20", "  IS", "  "whi te "  "whi te "  green "  "blue" ,  green "  "red",  "red",  "blue" ,  "gray"  "red",  "red",  "gray"  "gray"  "red",  "blue" ,  "gray"  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "ma ke "  "honda " ,  "honda " ,  "ford",  "toyota"  "toyota",  " sold"  " sold"  " sold"  " sold"  " sold"  . "2016  . "2016  10  11  " index" :  pr 1 ce "  " index" :  pr 1 ce "  " index" :  " index" :  pr 1 ce "  " index" :  pr 1 ce "  " index" :  pr 1 ce "  " index" :  " index" :  pr 1 ce "  " index" :  pr 1 ce "  " index" :  pr 1 ce "  " index" :  " index" :  pr 1 ce "  " index" :  pr 1 ce "  " index" :  pr 1 ce "  " index" :  " index" :  10000,  20000,  30000,  15000,  12000,  IBOOO,  BOOOO,  25000,  10000,  19000,  20000,  13000,  12500,  35000,  28000,  30000,  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  "color"  condition" :  condition" :  condition" :  condition" :  " condi t ion" :  "okay")  " new  " new  "good"  . "2016-05-18", "  . "2016-07-02", "  "2016-08-19" ,  "good"  "2016-11-05", "  . "2016-01-01", "  "2016-08-22", "  . "2016-02-12", "  "2016-02-12", "  "dodge" ,  "bmw" ,  "ford",  "dodge" ,  "dodge" ,  " sold"  " sold"  " sold"  " sold"  condition" :  condition" :  condition" :  condition" :  condition" :  "good"  " new  " new  " okay"  "good"  " sold"  "chevrolet" ,  "chevrolet " ,  " sold"  "sold" .  "2016-08-15", "  "2016-11  condition" :  condi t i on "  "good"  "okay"  "dodge" ,  "dodge" ,  " sold"  " sold"  condition" :  . "2016-03-09", "  condition" :  "2016-04-10", "  "okay"  " new  " sold"  . "2016-0B  "2016  11-20", "  " new  "chevrolet" ,  condition" :  "bmw" ,  " sold"  "good"  condition" :  **Aggregate DSL:-**    **If we didn’t provide size by default it is 10**    Machine generated alternative text: GET /vehicles/cars/_search  "size": 2ø,l  "query":  "nntch_all "  **Pagination from 0 to 5**    Machine generated alternative text: 2.  6.  9-  GET 'vehicles/cars/_search  "f ran": e,  "size":  "query":  "match_all    **Sorting:**  Machine generated alternative text: GET /vehi cles/cars/_search  51 •  "color": "blue"  "make": "ford" ,  " _ type": "cars",  "color": "gray" ,  "make": "dodge" ,  " _ index": "vehicles",  _ type" :  " cars"  "_id":  "AVyLItorqæ1E3BufUu7" ,  "_score": 1,  "frcn":  "stze":  " query" :  ø,  5,  52.  53  54  55  56  57  58  59  69  61  62  63 •  64 .  65.  66  67  68  69  79 •  71  72  73  _source" :  pri ce" :  "sold":  25øøø,  "2ø16-ø8-22",  "rutch_all  "sort":  Cprice": ("order":  1  7 people bookmarked this moment.  "condition": "  " _ index": "vehicles",  "AVyLItUrq041E3BufUvA" ,  "_score":  " _source" :  "pri ce" :  "sold •  125øø,  "2ø16-ø3-ø9",  "condition": "Okay"      **NOTE:** all documents are sorted and from, size is just to displaying document    **Count:**    Machine generated alternative text: GET /vehicles/cars/_count  2  "query":  3.  "nntch"  4  5  6  "count": p,  " _ shards":  "total": 5,  " successful " :  "failed": Ø  5,    **Aggregation query**    Machine generated alternative text: GET / vehicles/ cars/_search  "aggs":  " popu lar_cars" :  "terms " : (  "field": "make . keyword  146 •  147  148  149 •  152  153  154.  155  156  157 •  158 •  159  10  161 •  162 •  163  164  165  166 •  167  168  169 •  "popular_cars":  "doc_count_error_upper_bound" :  ø,  "buckets" :  " "dodg$',  key  "doc_count": 5  1  "key" :  "chevrotet" ,  "key  "doc _  "key" :  "key" :  "doc-count :  "bam" ,  count":  3  "ford" ,  "honda" ,  "doc_count" : 2  "tnvöta" ,    **NOTE: aggs , terms is reserved word in ES**    We need make.keyword ( because it is textual data)    Machine generated alternative text: GET / vehicles/ cars/_search  "match" "red")  "aggs":  "color": "red"  "rx*e": "dodge",  2  3  5  6  7  11  12.  13.  14  15  16  17  18.  19  21  22  23  24.  25 •  28  29  "took" : I,  "timed_out": false,  "_shards":  "total": 5,  " successful " :  "failed":  "hits":  "total":  "rmx—score" :  "hits": C  " _ index":  " _ typel:  5,  1.3862944 ,  "vehicles" ,  "cars" ,  C  popu :  "terms " : (  "field": "make.keyword"  "aggs":  "avg-price" :  "avg" : (  "field" :  "max—prtce  "max": (  "field" :  "price"  "prtce"  "_id": "AVyLItUr•qWIE3BufUv8",  "_score": 1.3862944,  _source" :  "price" :  "sold  3søøø,  "2ø16-ø4-1ø",  "mi n—price" : (  "min": (  "field": "price"  17:19/ 19:20  "condition": "new"  _ 1 n ex  ve es  "cars" ,  "_id": "AVyLItUrqO.'      NOTE: now aggregation will run based on query    **NOTE:** to find out price individually ( min,max and avg) we can use stats    Machine generated alternative text: GET 'vehicles/ cars/_search  "key": "bnsq" ,  13 •  14  15.  16  17  18.  19 •  2ø  21  22 •  23  24  25  26  27  28.  29 •  39  31  32  33.  34  35  36  37  38  39.  41 •  42  ø,  "aggregations":  "popular_cars":  " doc _count_er ror_upper _bound" :  "size"  3 • "query": (  " • ("color": "red")  "match .  "aggs":  "popular _cars":  "terms" : (  "field": "make . keyword"  "aggs":  "stats": (  "field" "price"  "buckets":  "key": "Podge" ,  "doc _ count": 3,  "stats-on-price  "count": 3,  ø,  "min" :  "max" :  "avg" :  "sum" :  18øøø ,  35øøø,  24øøø ,  72øøø  "doc_count": 1,  "count": 1,  "mtn" :  "avg" •  "sum" :  gøøøø ,  gøøøø  "key": "chevrolet" ,    What is terms in Elasticsearch?    Returns documents that contain an exact term in a provided field. You can use the term query to find documents based on a precise value such as a price, a product ID, or a username. ... By default, **Elasticsearch** changes the values of text fields as part of analysis.    Machine generated alternative text: GET / vehicles/ cars/_search  2.  3  "size"  4  5  6.  9  11.  12.  13 -  14  15.  16.  17  18 •  19.  2ø  21 •  23 •  24  25.  26.  28  "field": "price"  29.  30 •  (1  86  88  89  91.  92  93  94  95.  96  97  98  99  løø  103  :øø.øøez",  ø,  :øø. øøøz" ,  ø,  "from": 14516ø64øø  øøØZ" ,  "to": 14635296øØøø  "to_as_string" :  "doc_count": 1,  "avg_pri ce":  "value": Bøøøø  "key": "2ø16-øS  "from": 14635296øø  "to": 14832288øøøø  "to_as_string" :  "doc_count": 1,  "avg—pri ce":  "aggs":  "popular_cars" :  "terms " : (  "field": "make . keyword"  "aggs":  "sold_date_ranges":  "range" :  "field": "sold",  "ranges": C  "from": "2ø16-ø1-ø1",  "to": "  zø16-ø5-18"  "from": "2ø16-øs-18",  "to": "  2ø17-ø1-Ø1"  "aggs" : I  "avg—prtce" :  "avg" : (        Pouplar\_cars:---one bucket will create and solid\_date\_ranges another bucket will create means that is nested bucket    **Define bucket:**    **Bucket** aggregations in **Elasticsearch** create**buckets** or sets of documents based on certain criteria. Depending on the aggregation type, you can create filtering **buckets**, that is, **buckets** representing different value ranges and intervals for numeric values, dates, IP ranges, and more      NOTE: see above carefully aggs will apply on terms    Machine generated alternative text: GET / vehicles/cars/  search  32  33  34  35  36  37  38  39  41  42  43  45  46  47  48  49  index"  "vehicles"  _ type'  "cars"  id"  " svzTeWsB4x1m2yUc Fw2x" ,  score"  null,  source'  " price"  35eee,  "color"  "red" ,  "make"  "dodge" ,  "sold"  "2e16-e4-1B",  "from" :  'size"  " query"  e,  "match all" .  " sort" .  "price"  "order" :  GET  " desc "  "new"  count  " condition'  "sort"  35eee  / vehicles/cars/    Understand carefully    Condition: good,new and ok  Make-bmw    Machine generated alternative text: GET / vehicles/ cars/_search  2.  "size"  4  5  6.  "aggs":  "car_condi tions":  "terms " : (  8.  "field":  "condi t ion. keyword"  9  .2.  .4  6.  "field": "make.  9  16  18.  19.  2ø  21  22.  23  24.  25.  26  28.  29.  31  32 •  33.  34  35  36.  38  39  41 •  42  43  45 •  "doc-count-error-upper-bound" :  "buckets" :  "key  ": "good",  "doc _ count" 6,  "avg_price":  "value": 19øøØ  "make":  e,  ø,  "aggs"  "avg—price" :  "avg":  "field" :  "make" :  "terms" : (  "price"  keyword"l  " _bound :  "buckets":  "key": "dodge",  "doc_count": 2  "key": "toyota" ,  "doc_count" : 2  "key": ,  "doc_count": 1  1  "key": "chevrolet" ,  "doc_count": 1    Machine generated alternative text: GET /vehicles/cars/_search  "size"  "key" : "good" ,  "aggs":  " car_condi tions" :  "terms" : (  " •l"max": ("field":  16  18.  19.  2ø  21  22.  23  24.  25.  26  27  28.  29.  31  32 •  33.  34  35  36.  38  39  41 •  42  43  44 •  "buckets" :  "doc _ count" : 6,  "avg_price":  "value": 19øøø  "make":  "doc_count_error_upper_bound  "buckets": C  "key": "dodge",  "doc_count": 2  "key" : ,  "doc_count": 2  "key": ,  "doc_count" :  1  "key": "chevrotet" ,  "doc_count": 1  ø,  "field":  "condi t ion. keyword"  "aggs" : (  "avg—price" :  "avg":  "field":  "make" :  "terms" : (  "price"  "price"))  "price"))  "field": "make. keyword"  "nn n_prt ce"  : ("min":  ("field":      **NOTE:** aggregation occur in real time which is more faster than relational database      **Configure logstash**    **Logstash** helps **you** to collect data from multiple systems into a central system wherein data **can** be parsed and processed as required. Also, **Logstash** helps **you**to gather the data from multiple systems and store the data in a common format, which **is** easily **used** by Elasticsearch and Kibana.      Machine generated alternative text: Andy sis    It is used to insert data from multi source simultaneously.  The source can be anything  **To test logstash**      Machine generated alternative text: Data  Inputs  Outputs  Destinations  Logstash Pipeline  Filters    Machine generated alternative text: input  filter  output      Machine generated alternative text: 2  3  Download and unzip Logstash  Logstash can also be installed from our package repositories using apt or yum. See Repositories in the Guide.  Prepare a logstash.conf config file  Run  bin/logstash  -f logstash.conf          Machine generated alternative text: Last sat Aug 18 ttys—  S cd elasticsearch_cm'rse/  t r Ch—6.3.2/  •tetasticsearch_ccwrse: S Cd  cmTRIMJToRS  txt  -e stdin      **Very imp**    logstash -e 'input{stdin{}}output{stdout{}}'    Important    Machine generated alternative text: C httpw"m•utEco'gu•eV.nogstasnJ82,Ougins•in"Jts•t•IentmI  L' CTS • iAn  On non.W•ndcxs systems yt•u otnain the inale Of a e. ts -u  File rotation  is detected this input. regard•ss of •s via a  a TO thx Mite to for  rotation has taken plxe, both the orig.nal filename and rotated (e.g.  nog'sy•siog, t) •n '*name patterns to watch (the path ofl•on). Note that  the rotated will treated as a new file So d is Set to  file  WRh the default vahR Of start_nsit ion messages Mitten to the end of the file  tY-* to rotation and under the (M  deter'vn«' the discover _ interval will not get picked up.  File Input Configurationoptions  ms supp«ts ccmfwwation the Options later.  typ    Machine generated alternative text: Secure httpw/'m.easte„co'øw•e'e••xnogstasnJ62,'ndewrtm'  CTS • •F  • Res.uency  tkployvng Scaling Logstash  • perfo«narxe  • Morvtorng Logstash  • Mor.torng AP's  • Input plugins  • Output  • '*logins  • Codec  • to Logstash  • Re•ease Notes  • X•Pack Release Notes |  |  |
|  |  | Machine generated alternative text: PUT  / cour ses / classroom/ 9  "Tax Accounting 200"  "room" •  professor "  "Thomas Baszo"  " finance "  "department " •  part—time",  "facutly type" •  an intermediate  cour se  cover Ing various  a spe c t s  of tax law"  "email":  "baszot@onuni . com"  "students enrolled": 17,  " course_publish date"  "course description" •  "2016-06-15",  "Tax Act 200 |
|  |  |  |