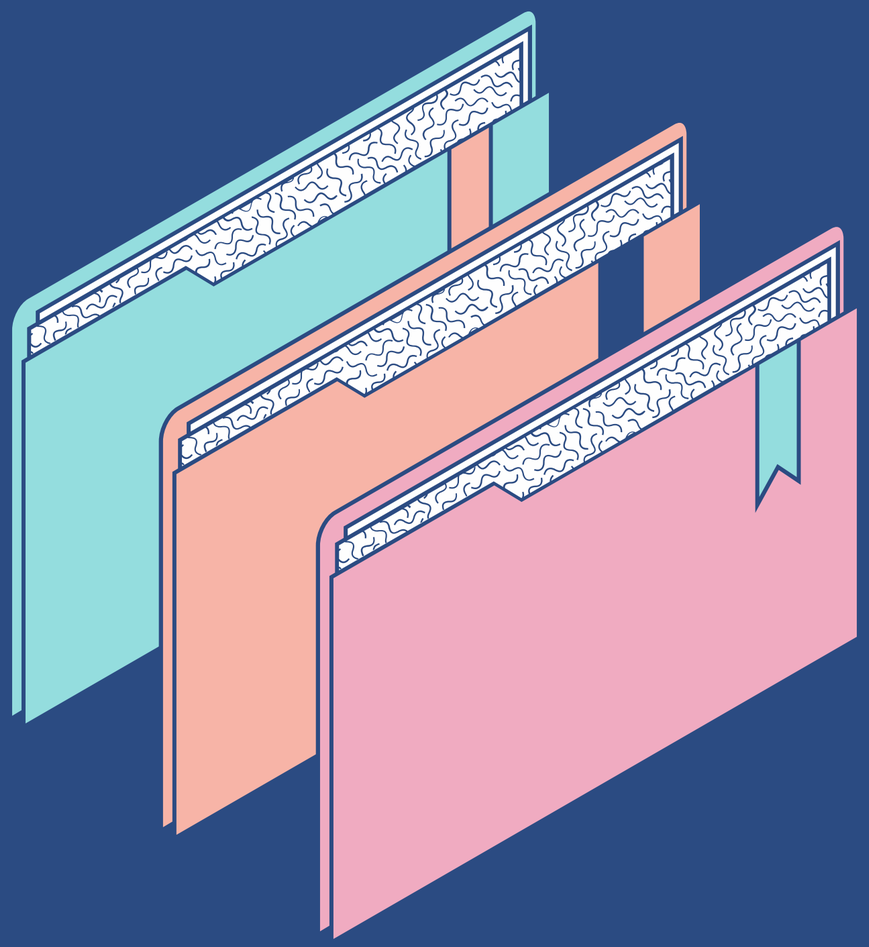
WEEK 3 PROJECT

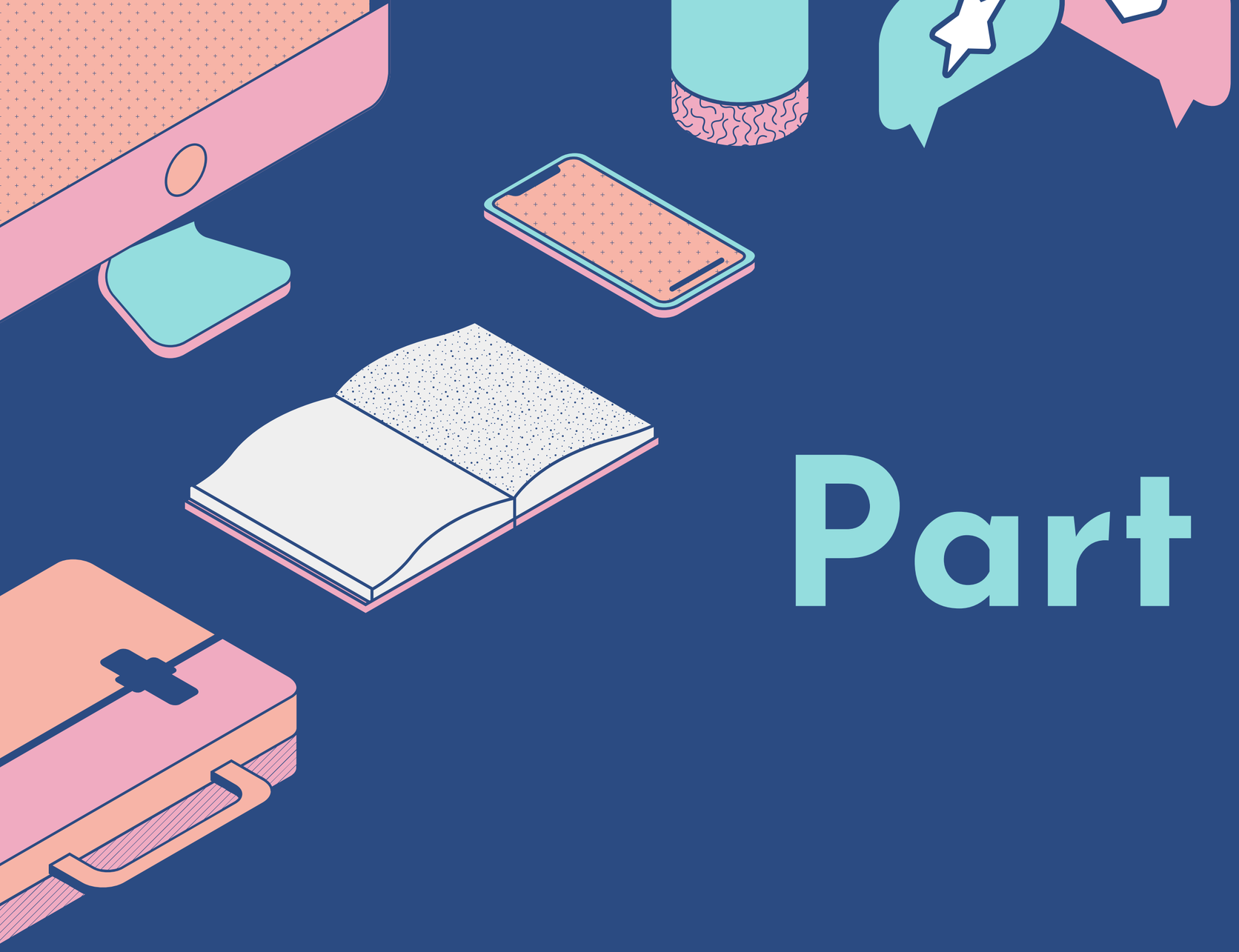
MongoDB Project

DaQuest Team



Agenda

- PART I (VOLCANOES_OF_EARTH) DATASET
- PART II (H_AND_M) DATASET



Part 1

volcanoes of earth Dataset dashbord:

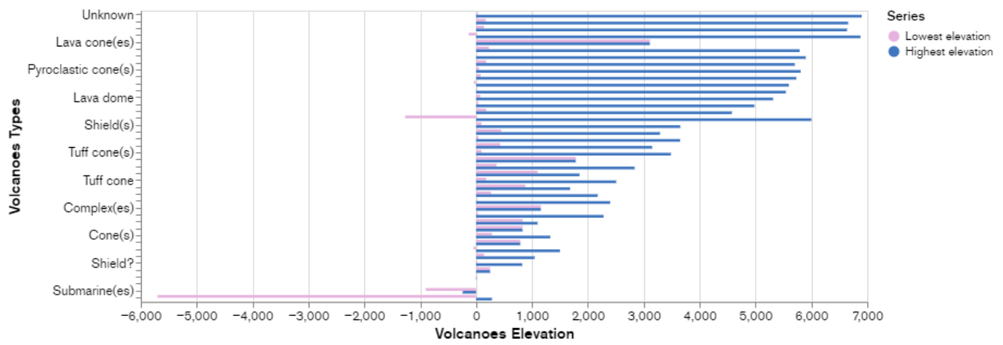
Volcanoes Dashboard

Count the volcanoes number in all countries

This map shows multi colors depend on the volcanoes numbers in each country

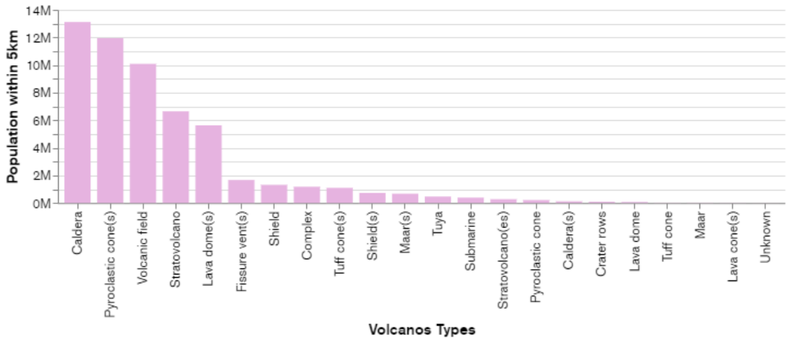
Volcanoes types with there max and min elevation

This chart present all the max and min "Underwater" elevation based on the volcanoes types



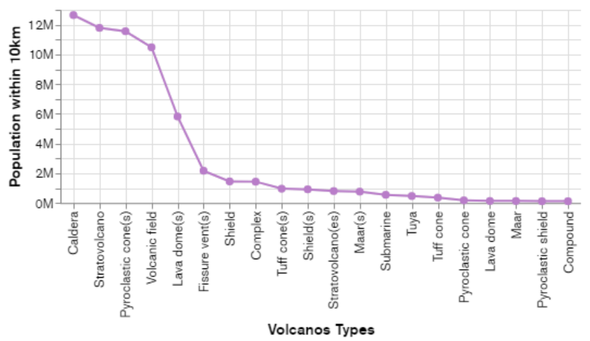
Volcanos types where population within 5km

This chart shows the volcanos types within 5km more than 5000 people



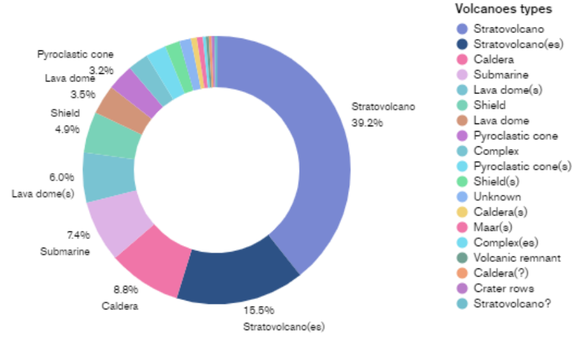
Volcanoes types within 10km

This chart present the volcanoes types where the population more than 100k people ...



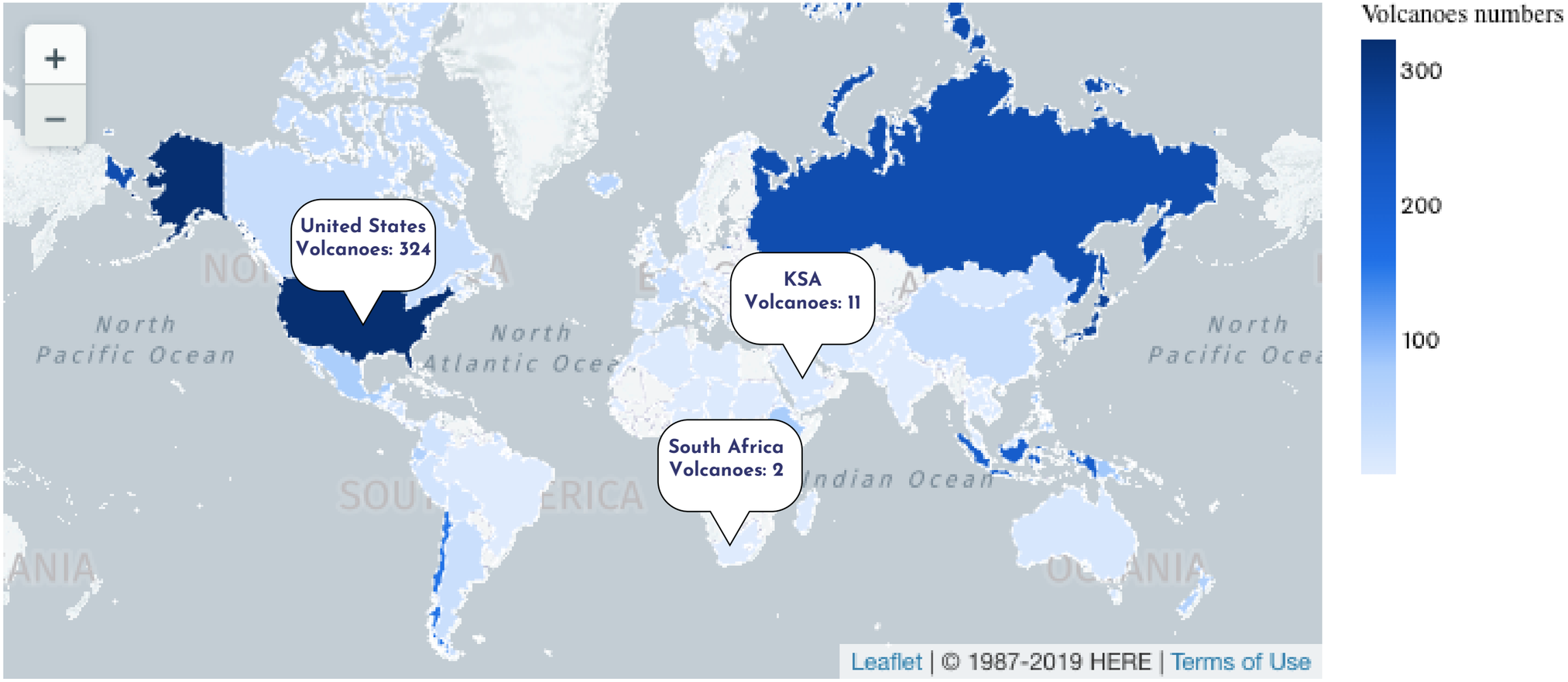
Volcanoes types in Japan

This chart presents all the Volcanoes types in Japan

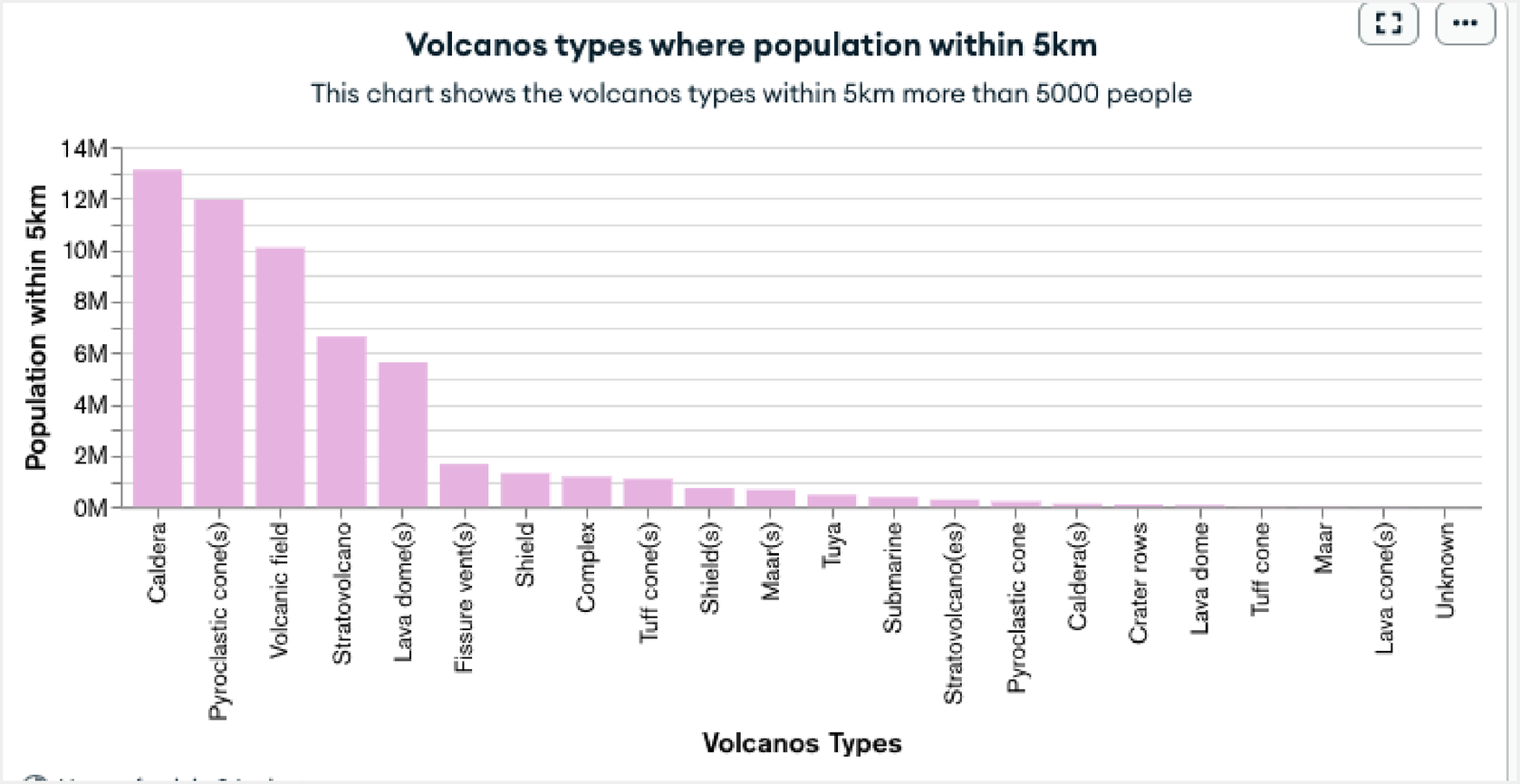


Count the volcanoes number in all countries

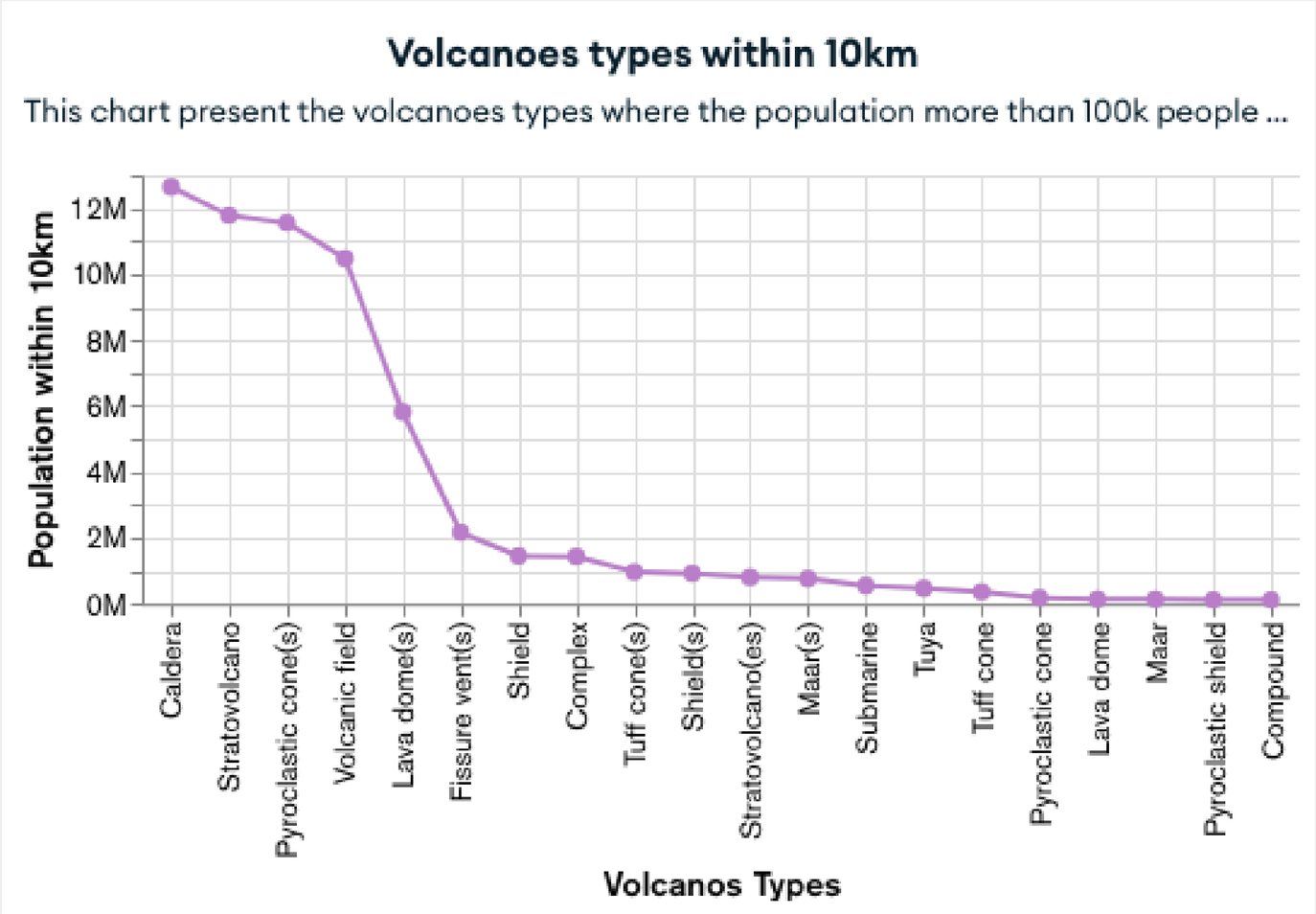
This map shows multi colors depend on the volcanoes numbers in each country



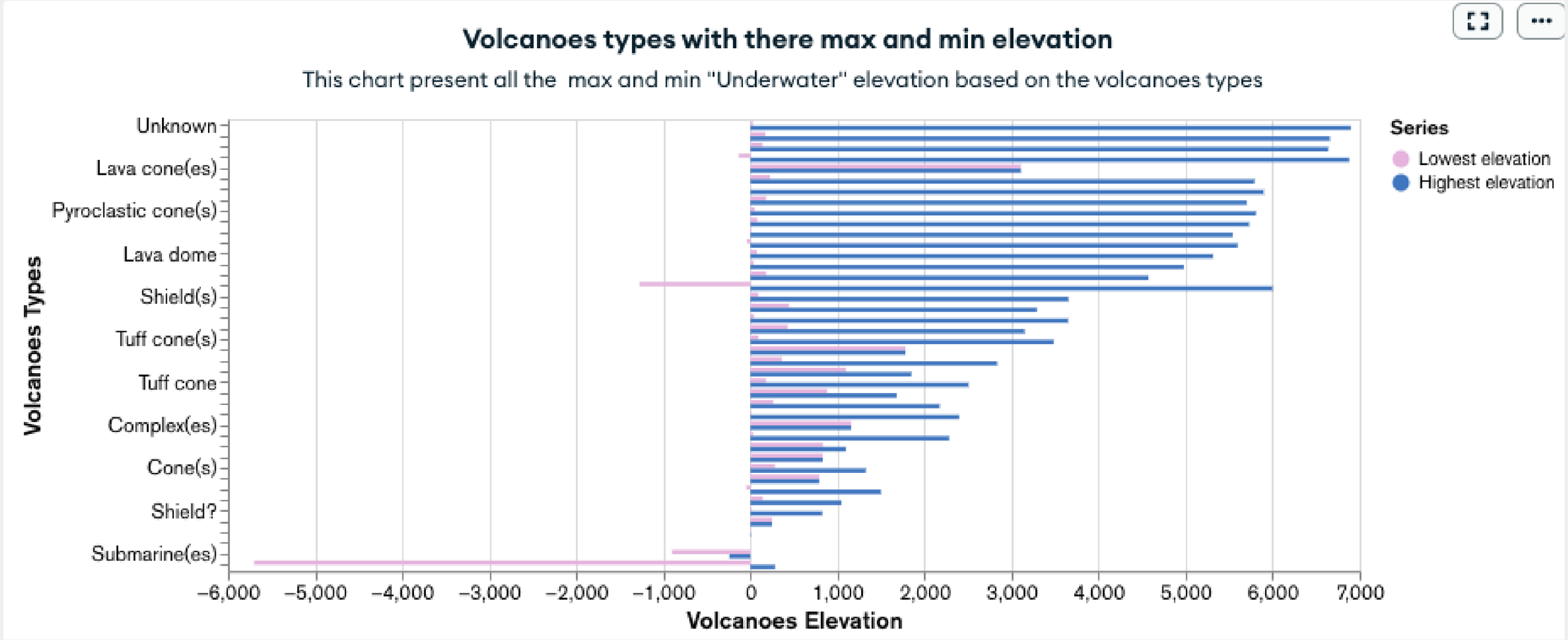
- Caldera type is the highest population with a value of 13 M.
- The lowest population is the Tuff cone and Maar type with a value of 0 M.



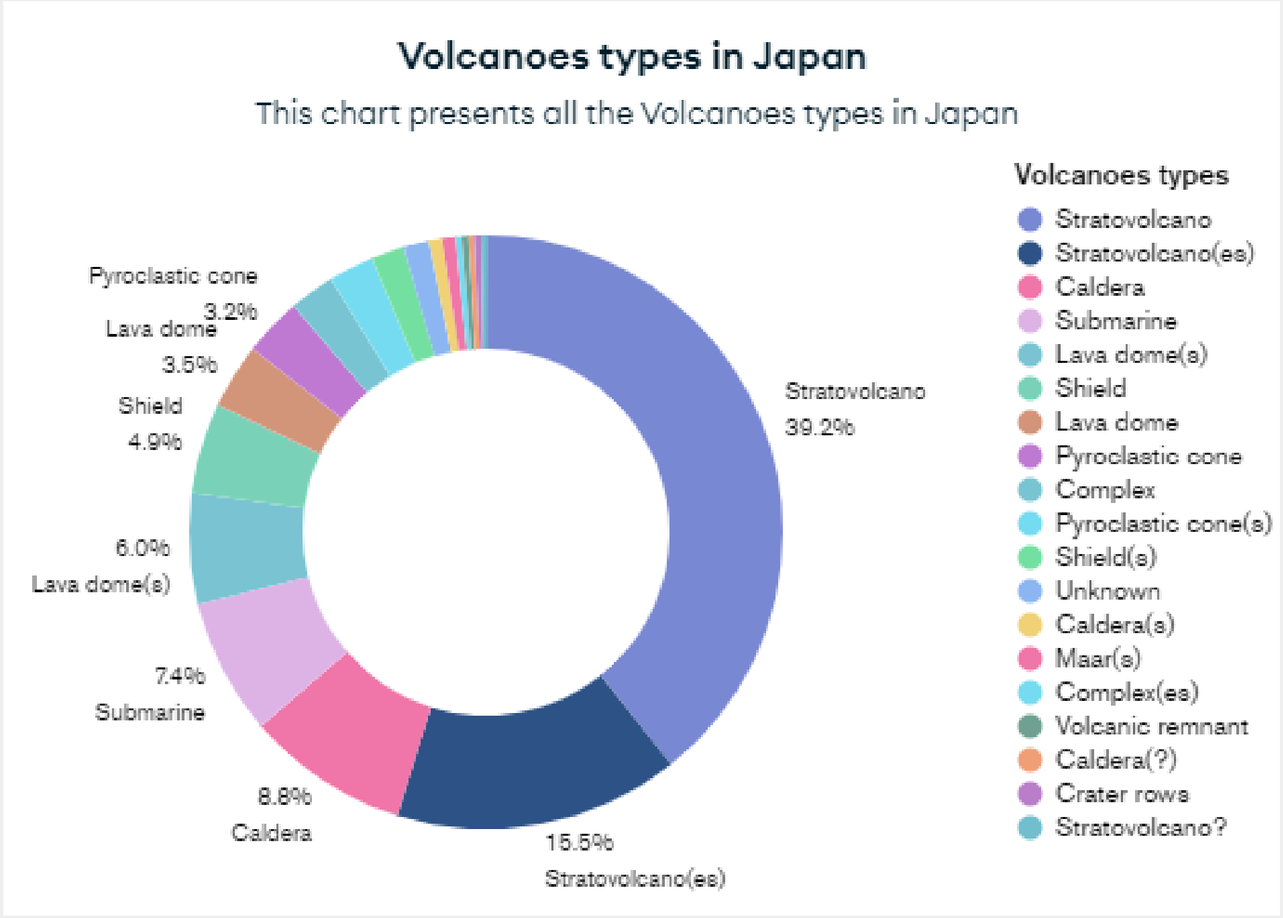
- Caldera type is the highest population with a value of 12.5 M.
- The lowest population are Lava domes, Pyroclastic shields, and Maar type with a value of 0.5 M.



- **The Unknown type has the highest length within 7880 meters.**
- **The lowest elevation of 5700 meters under the water in the Submarine type**



- **The 3 most common volcanic types in Japan were Stratovolcano (39.2%), Stratovolcano(es) (15.5%), and Caldera (8.8%).**



Part 2

H&M

function that returns all the red or orange t-shirts

$match → $project

Filter Product Type to T-Shirt and Color to red or orange using "$in"

function that returns all the red or orange t-shirts

$match → $project

to get the name
of the product
type and color

After calling the function in Jupyter notebook length: 316

```
[{'product_type_name': 'T-shirt', 'colour_group_name': 'Orange'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Orange'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Orange'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Orange'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Orange'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
 {'product_type_name': 'T-shirt', 'colour_group_name': 'Red'},
```

Function return specific color and name using parametar

$match

Filter
index_group_name
and
colour_group_name
to arguments
value

used $and and $eq operation to return exact results that contain the argument that assigns to index_group_name and, colour_group_name

After calling the function in Jupyter notebook arguments ->Black, Ladieswear

Length -> 10188

```
colour_group_name': 'Black',
perceived_colour_value_id': 4,
perceived_colour_value_name': 'Dark',
perceived_colour_master_id': 5,
perceived_colour_master_name': 'Black',
department_no': 1676,
department_name': 'Jersey Basic',
index_code': 'A',
index_name': 'Ladieswear',
index_group_no': 1,
index_group_name': 'Ladieswear',
```

retrive Specific Product Info

$project **$match**

To get product_type_name,
color_group_name,
price,
department_name,
discount_%

retrive Specific Product Info

$project

$match

to filter the product_type_name,
to match with arguments 1
and
color_group_name,to match with
arguments 2
and
price, to match with arguments 3

$eq and $in to compare
filed with argument

After calling the function in Jupyter notebook with arguments:

Vest top, White, [54.4, 160.67]

```
[{'product_type_name': 'Vest top',
  'colour_group_name': 'White',
  'department_name': 'Jersey Basic',
  'price': 54.4,
  'discount_%': 10},
 {'product_type_name': 'Vest top',
  'colour_group_name': 'White',
  'department_name': 'Jersey Basic',
  'price': 160.67,
  'discount_%': 10}]
```

calculates_the_discount

$match → $project → $sort → $limit

Filter Product Type to argument

calculates_the_discount

$match → **$project** → $sort → $limit

to get the prod_name, price, discount_%, detail_desc and calculate the new_price using "$price", "$multiply" and "$divide"

price - ((discount/100)*price)

calculates_the_discount

$match → $project → **$sort** → $limit

Sort the outputs in ascending order using the new price, from cheapest to most expensive

calculates_the_discount

$match → $project → $sort → $limit

Limit to the first 50 results

After calling the function in Jupyter notebook with T-shirt argument: length: 50


```
{'prod_name': '2P SS T-shirt mixed',
 'detail_desc': 'T-shirts in soft, organic cotton jersey.',
 'price': 5.76,
 'discount_%': 25,
 'new_price': 4.32},
{'prod_name': 'Price TEE TVP',
 'detail_desc': 'T-shirt in soft cotton jersey with ribbing around the neckline.',
 'price': 5.41,
 'discount_%': 20,
 'new_price': 4.328},
```

Function return specific color and name using parametar

$match **$search**

Filter all document in collection to match it with argument that we pass it as parameter

Function return specific color and name using parametar

$match

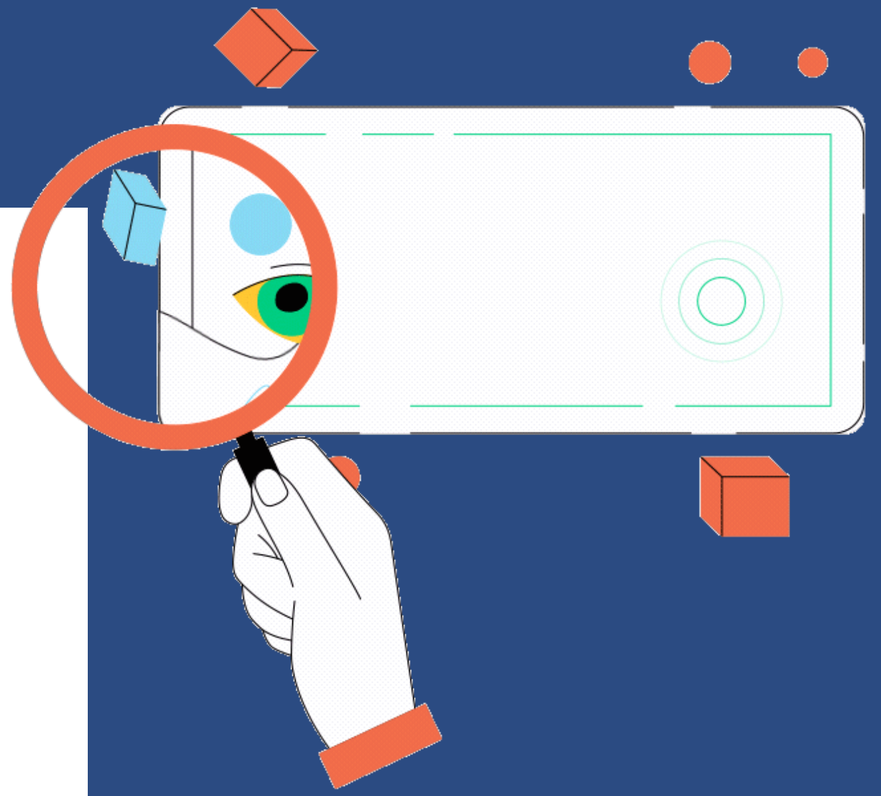
$Search

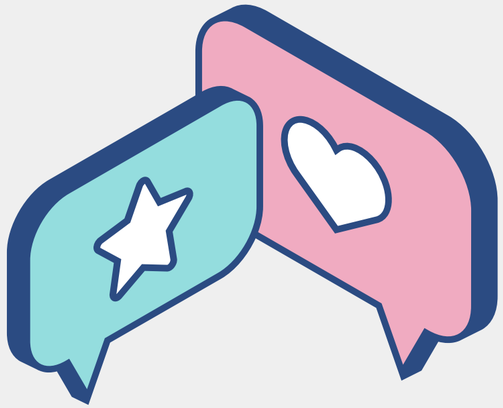
Search in all documents to find match string for argument

used $text index to search on all documents

After calling the function in Jupyter notebook argument ->Blanket

```
[{'_id': ObjectId('634a98755129ecd14727aa66'),
  'article_id': 615576001,
  'product_code': 615576,
  'prod_name': 'WAFFLES 2p swaddle',
  'product_type_no': 349,
  'product_type_name': 'Blanket',
  'product_group_name': 'Interior textile',
  'graphical_appearance_no': 1010016,
  'graphical_appearance_name': 'Solid',
```





1st Index

Create compound index for : the product type name and color

```
[160] # examined all documents and retrieved 558, time taken 81ms
      collection.find({"product_type_name": "Vest top", "colour_group_name": "White"}).explain()['executionStats']
```

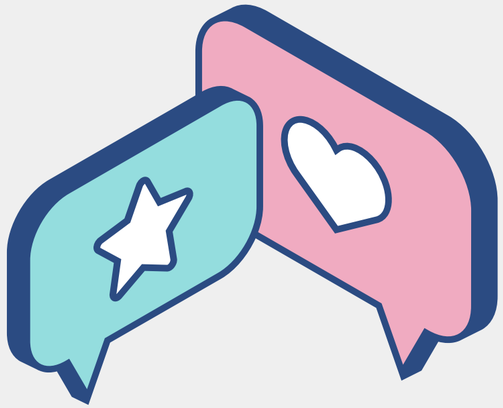
```
{'executionSuccess': True,
 'nReturned': 558,
 'executionTimeMillis': 81,
 'totalKeysExamined': 0,
 'totalDocsExamined': 105542,
```

```
# create compound index
collection.create_index([("product_type_name", pymongo.ASCENDING), ("colour_group_name", pymongo.DESCENDING)])
```

```
'product_type_name_1_colour_group_name_-1'
```

```
# query with the index examines only documents that correspond to our query and takes only 2 ms
collection.find({"product_type_name": "Vest top", "colour_group_name": "White"}).explain()['executionStats']
```

```
{'executionSuccess': True,
 'nReturned': 558,
 'executionTimeMillis': 2,
 'totalKeysExamined': 558,
 'totalDocsExamined': 558,
```



2nd Index

Create compound index for : product type, color, and price

```
[165] # examined all documents and retrieved 466, time taken 2ms
      collection.find({"product_type_name": "Vest top", "colour_group_name": "White", "price": {"$lte": 165}}).explain()['executionStats']
```

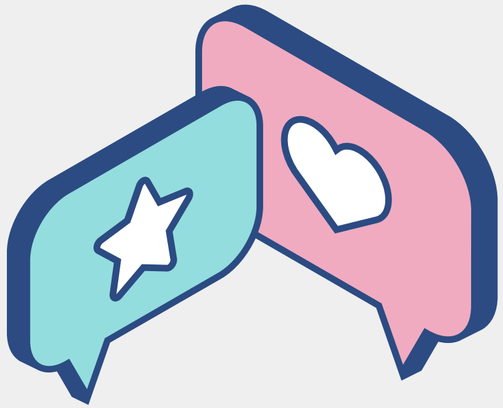
```
{'executionSuccess': True,
 'nReturned': 466,
 'executionTimeMillis': 2,
 'totalKeysExamined': 558,
 'totalDocsExamined': 558,
```

```
# create compound index
collection.create_index([("product_type_name", pymongo.ASCENDING), ("colour_group_name", pymongo.DESCENDING), ("price", pymongo.ASCENDING)])
```

```
'product_type_name_1_colour_group_name_-1_price_1'
```

```
# query with the index examines only documents that correspond to our query and takes only 2 ms
collection.find({"product_type_name": "Vest top", "colour_group_name": "White", "price": {"$lte": 165}}).explain()['executionStats']
```

```
{'executionSuccess': True,
 'nReturned': 466,
 'executionTimeMillis': 2,
 'totalKeysExamined': 466,
 'totalDocsExamined': 466,
```



3rd Index

Create a Text Index based on all text fields in the collection.

```
[181] # examined all documents and retrieved 26001, time taken 130ms
      collection.find({"index_name": "Ladieswear"}).explain()['executionStats']
```

```
{'executionSuccess': True,
 'nReturned': 26001,
 'executionTimeMillis': 130,
 'totalKeysExamined': 0,
 'totalDocsExamined': 105542,
```

```
# Create a Text Index based on all text fields in the collection
collection.create_index([("$**", pymongo.TEXT)])
```

```
'$**_text'
```

```
# examined all documents and retrieved 39737, time taken 74ms
collection.find({"$text": {"$search": "Ladieswear"}}).explain()['executionStats']
```

```
{'executionSuccess': True,
 'nReturned': 39737,
 'executionTimeMillis': 74,
 'totalKeysExamined': 39737,
 'totalDocsExamined': 39737,
```

Thank you

Do you have any questions?

Send it to us! We hope you learned something new.