




5교시. Vector Search 개요, 실습

- 
- 1 AI & Vector Search
 - 2 Hybrid Search
 - 3 Vector Search 실습 : RGB 모델

5-1. AI & Vector Search



Key Milestones in the History of AI | 20th Century

AI is born

Early Developments

Expert Systems

Machine Learning



Can a Machine think?

1950

Alan Turing test
- if a machine tricks a human into thinking it's a human, then it's intelligence.



AI term coined

1956

John McCarthy introduced the term 'Artificial Intelligence' at The Dartmouth Conference.



Gen. Problem Solver

1957

General Problem Solver is the first AI algorithm. It can solve formalized problems (e.g. Towers of Hanoi).



The chatbot Eliza

1965

Eliza is the first chatterbot - or "chatbot" modernly. It can simulate a human conversation.



R1, an Expert System

1980

R1 uses 2500 rules to ensure that the customer's order is complete, saving the company \$25M a year.



Deep Blue

1997

IBM's Deep Blue supercomputer defeats the world chess champion Garry Kasparov.



Kismet

1998

Cynthia Breazeal at MIT creates Kismet, a robot that can detect and respond to people's feeling.

1st AI Winter
1974-1980

2nd AI Winter
1987-1993

Key Milestones in the History of AI | 21st Century

Machine Learning

Deep Learning

Generative Models



Watson

2006

IBM's Watson question-answering system is created. In 2011, it defeats Jeopardy!'s champion.



Siri & Alexa

2011

Apple released Siri, an AI assistant that can respond to voice. 3 years later, Amazon launched Alexa.



Google AI

2014

Google AI recognizes cats from 10 million Youtube videos with almost 75% accuracy in 3 days.



AlphaGO

2016

AlphaGo defeats top Go player Lee Sedol in Seoul. Go is incredibly difficult given the vast number of positions.



AlphaFold

2020

DeepMind's AlphaFold system can predict protein structure, with implications for drug discovery and biology.



DALL·E

2021

DALL·E, and a year later Midjourney and StableDiffusion, can generate images from textual descriptions.



ChatGPT

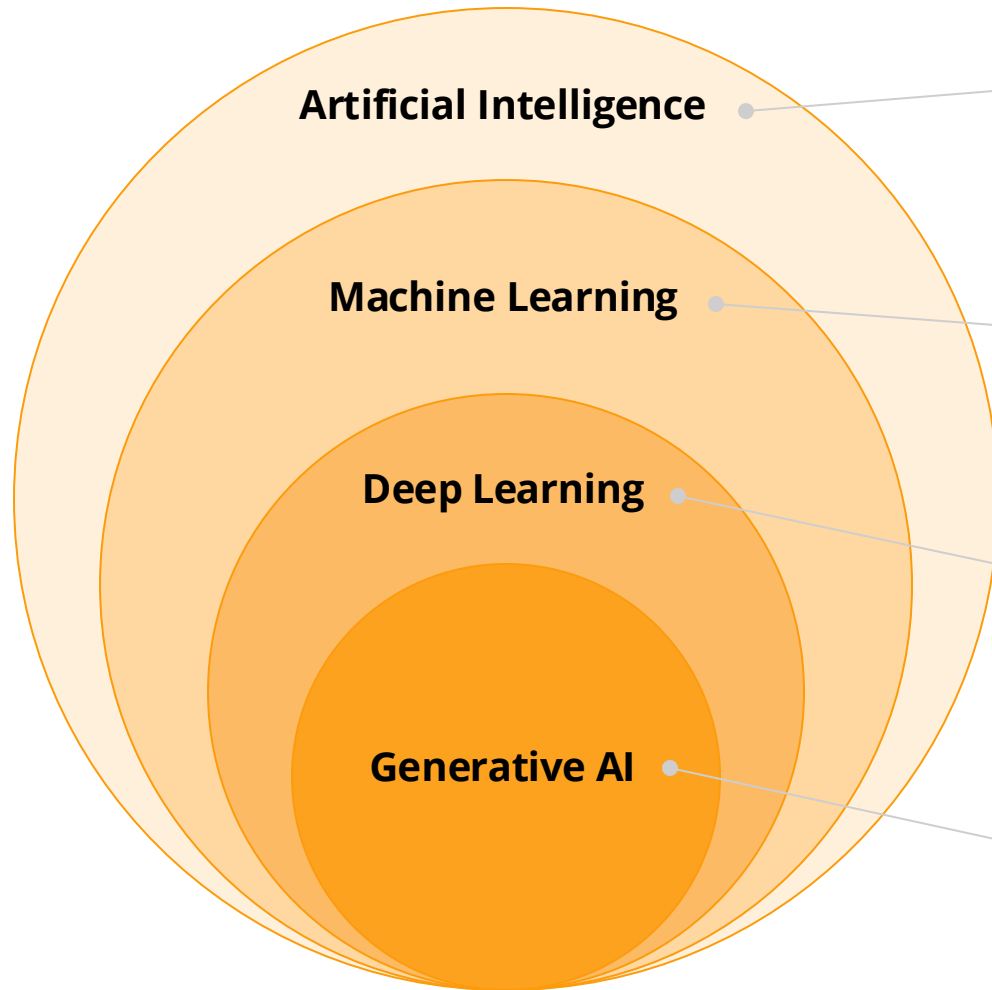
2022

OpenAI releases the AI chatbot ChatGPT. It is based on the Large Language Model GPT-3 created in 2020.

AI Boom

AI Explosion

The Technology behind AI



Artificial Intelligence (AI)

Techniques that allows computers to emulate human behavior (e.g. learn, recognize patterns, solve complex problems).

Machine Learning (ML)

A subset of AI, using advanced algorithms to detect patterns in large data sets, allowing machines to learn and adapt for prediction or content generation use cases.

Deep Learning (DL)

A subset of ML, using multiple layers of artificial neural networks that simulate human brains for in-depth data processing.

Generative AI (GenAI)

A subset of DL, using models that generate content like text, images, or code based on provided input.

Powering Apps: A Combination of Predictive & Generative AI

Predictive AI

Outcomes and Insights driven by ML



- Predict Outcomes based on historical data
- Utilize ML algorithms for pattern recognition
- Learns patterns and correlations from data
- Drives decision making and Future planning
- High ROI, trained on proprietary data

- Predictive Insights
- Dynamic Pricing
- Fraud Detection
- Inventory Optimization

+

Generative AI

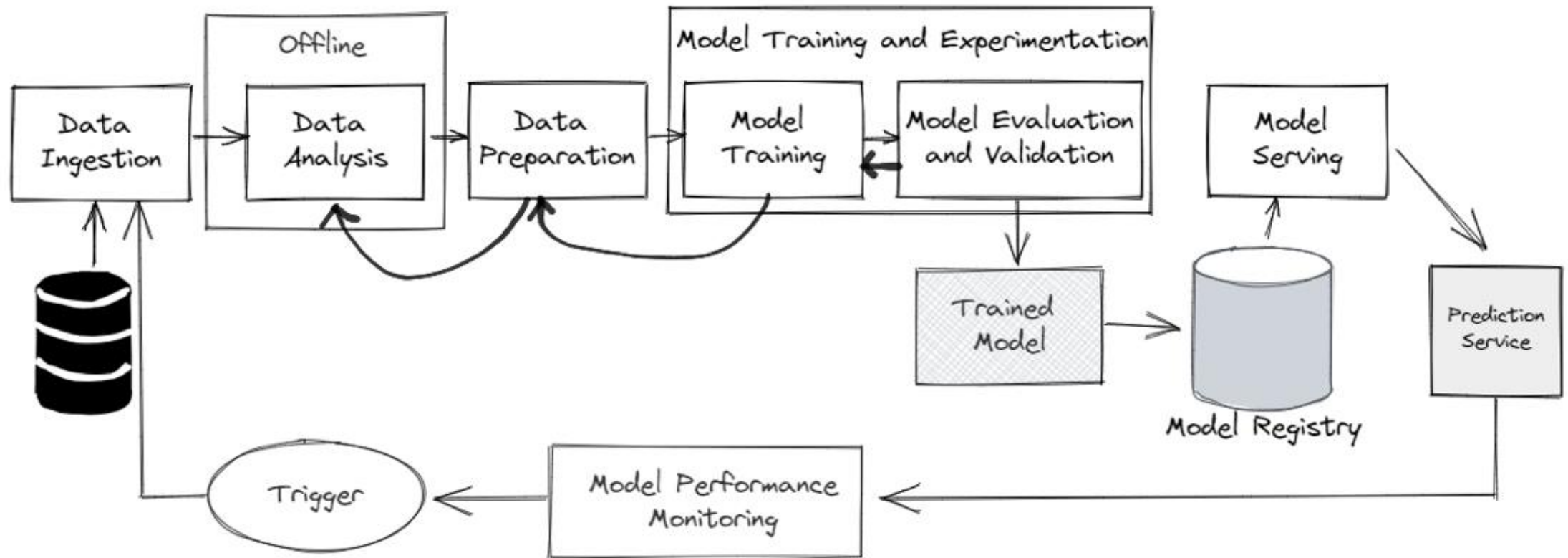
Generate Content and Experiences



- Generate or Synthesize content
- Needs large amounts of unlabeled data for training
- Generates new data probabilistically
- Fosters creativity, innovation
- Accelerates human productivity

- Hyper-personalized experiences
- Contextualized content
- Chatbots and CoPilots
- Synthetic data and Summarization

Model? Machine Learning Workflow



출처 : <https://www.iguazio.com/blog/ml-workflows-what-can-you-automate/>
<https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning>

What is a Vector

What is a Vector? | Basic RGB Example

This is a vector

2.6

11.3

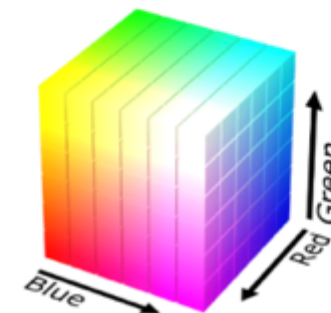
-4.2

First value

Second value

Third value

The RGB model example



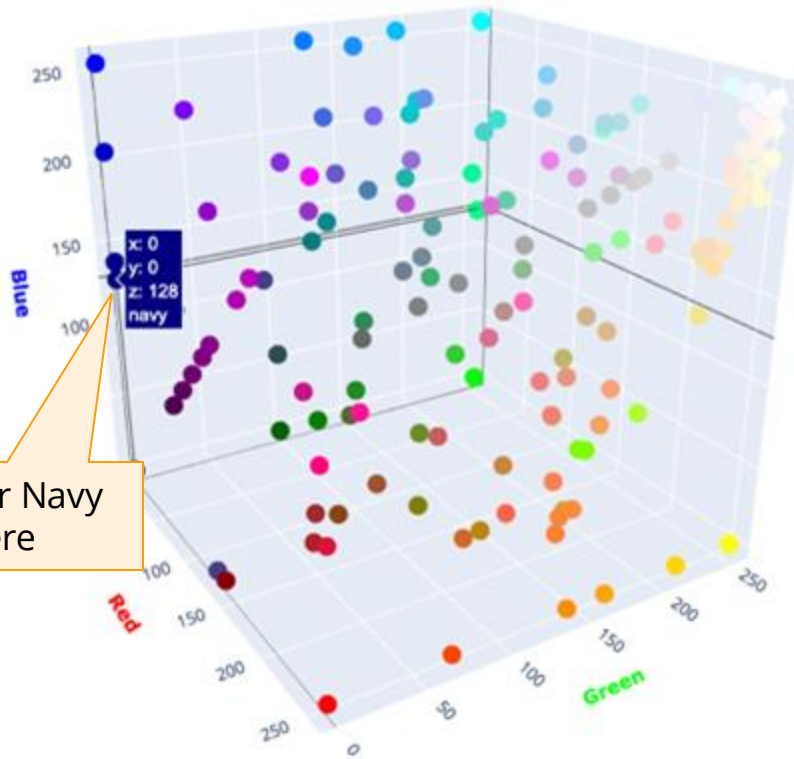
The model used to create the colors you see on TV and computer screens.
Each color is the addition of a **Red**, **Green** and **Blue** primary colors.

Here, it contains 3 values
=> its dimension is 3

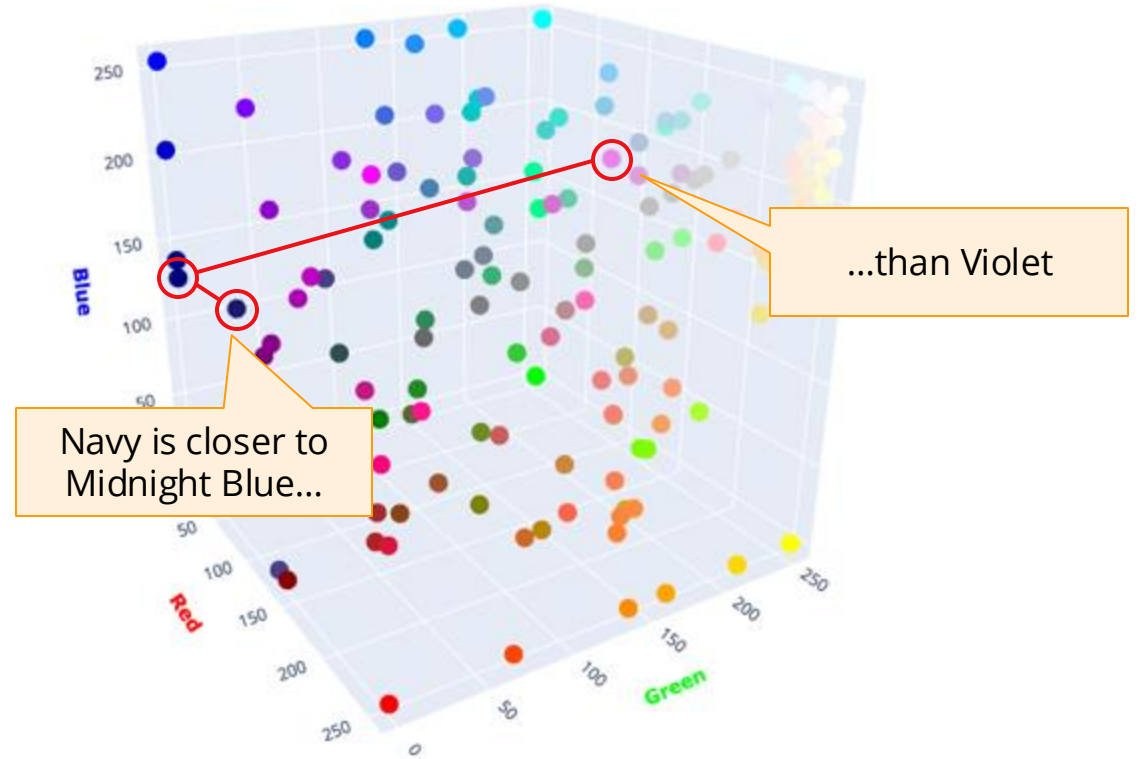
A Vector is a just an **array of numerical values**

Vectors Similarity

Example of 123 vectors of RGB colors



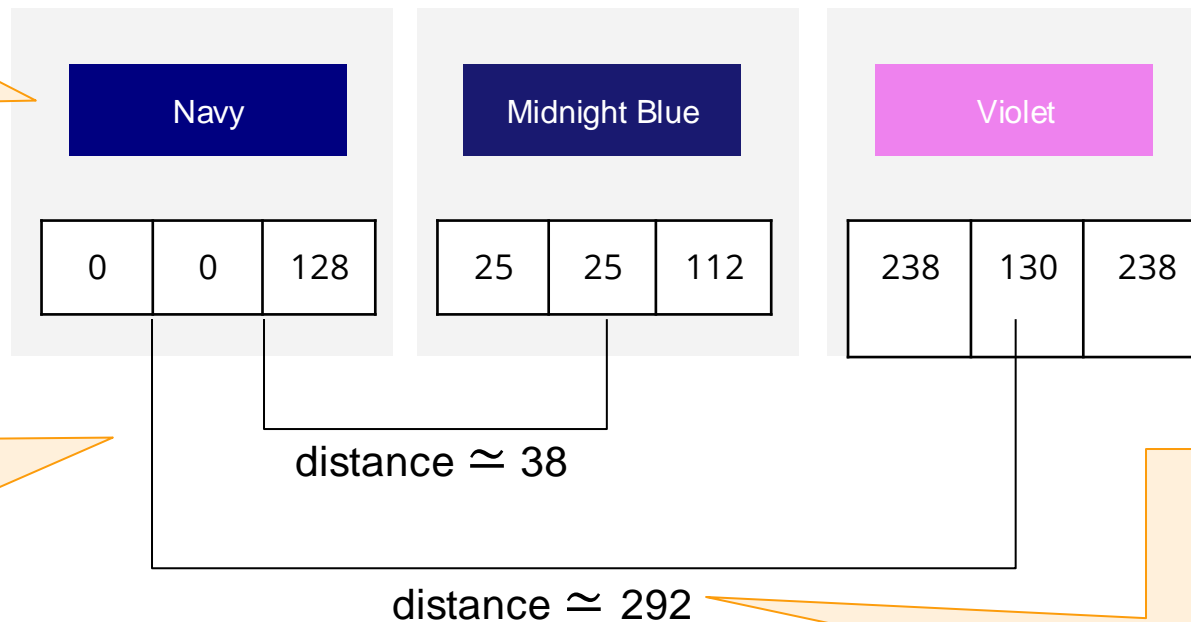
Similar colors are closer to each other



Vectors make it possible to translate **similarity** as perceived by humans to **proximity in a vector space**.

How does Similarity works

To the human eyes,
Navy is closer to Midnight
Blue than Violet



Mathematically,
we got the same result by
comparing the vectors

Vectors are compared using
a similarity distance.

Here the *euclidean distance*
 $292 \approx \sqrt{(238-0)^2 + (130-0)^2 + (238-128)^2}$

Vectors can easily be compared mathematically using a **similarity distance**

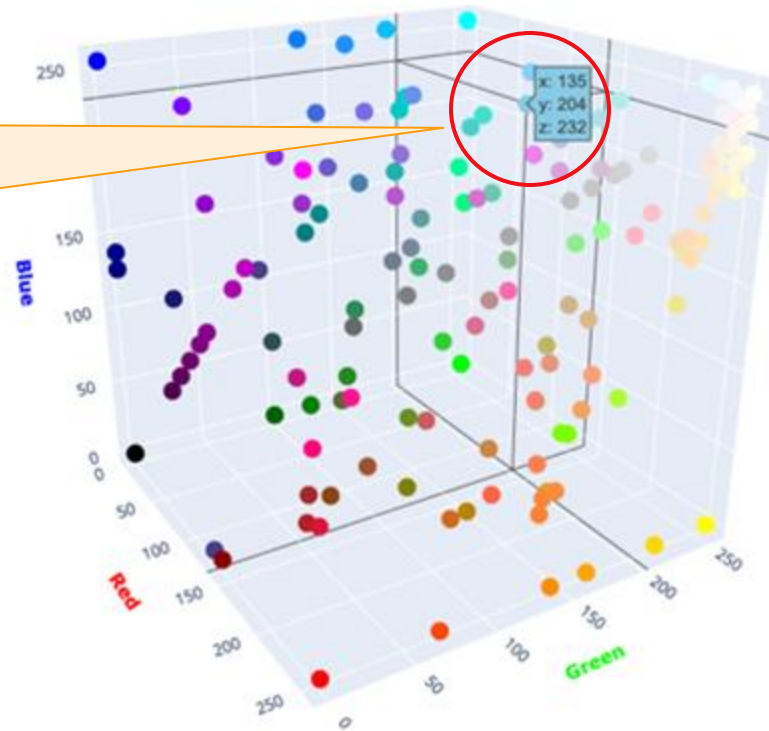
Similarity Search with K-NN (K-Nearest Neighbors)

Example of 123 vectors of RGB colors

Top k-NN results of the query

Which are the top k nearest neighbors to this color?

[135,204,232]



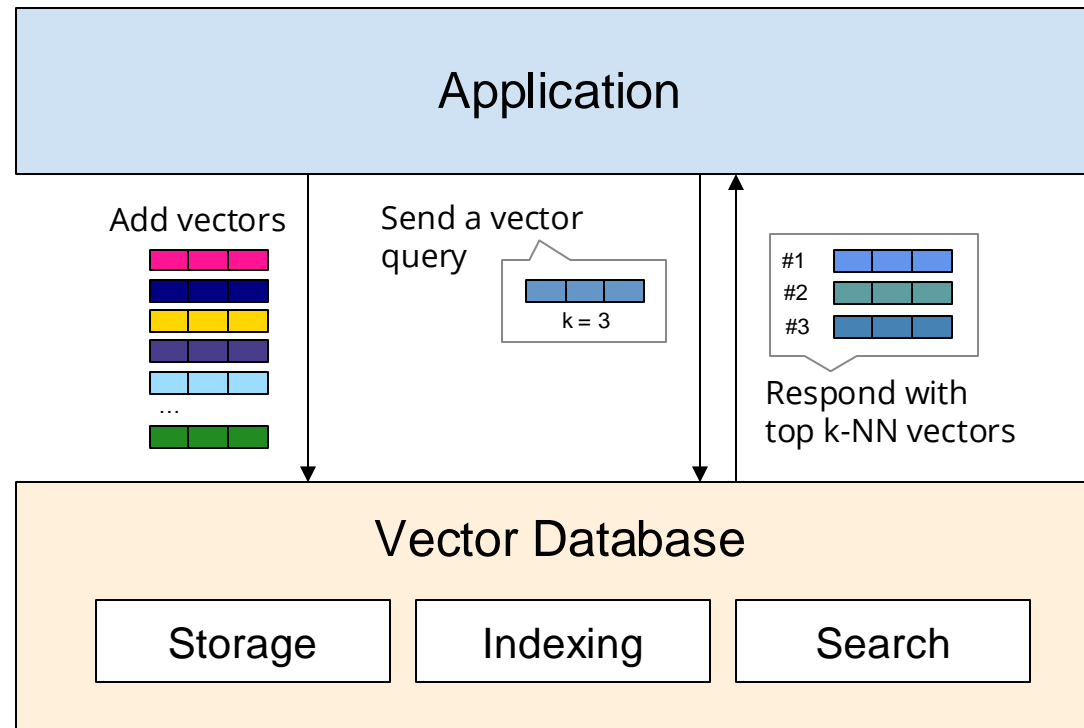
#1 sky blue
[135,206,235]

#2 light sky blue
[135,206,250]

#3 light blue
[173,216,230]

A similarity search is a query that **finds the k nearest neighbors to a vector**, as measured by a similarity metric

What is Vector Database



Vector databases provide the ability to **store, index and search vectors** using similarity search

Couchbase Vector Search

The vectors are stored as
a field in JSON documents

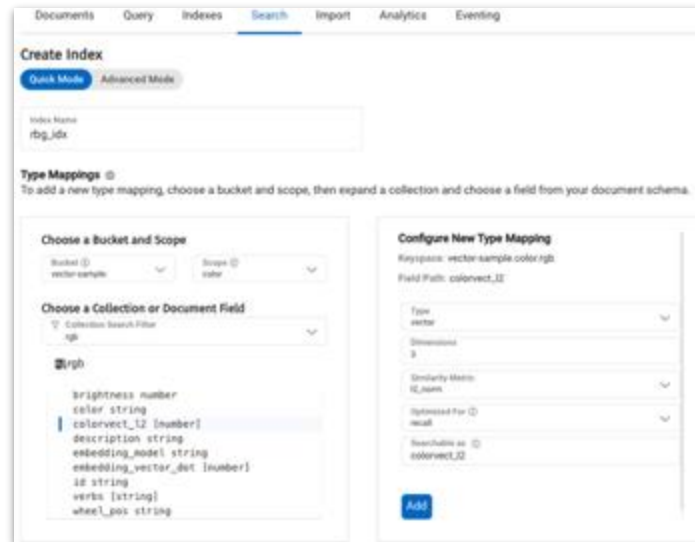
```
{
  "id": "#000080",
  "color": "navy",
  "brightness": 14.592,
  "colorvect_l2": [0, 0, 128],
  "description": "Navy is a deep, rich color that exudes sophistication. It is a dark shade of blue that is often associated with authority, stability, and elegance. Navy is a versatile color that can be both bold and understated, making it a popular choice in fashion and interior design. It is a timeless color that never goes out of style and adds a touch of sophistication to any look or space."
}
```

JSON Storage



Data Service

A **Vector Index** must be created to
allow the vectors to be searched

A screenshot of the Couchbase Search console. The 'Create Index' page is shown with the 'Quick Mode' tab selected. The index name is 'rgb_idx'. Under 'Type Mappings', a new mapping is being created for the 'colorvect_l2' field. The field is mapped to a 'vector' type with a dimension of 3. The 'Similarity Metric' is set to 'l2_norm' and 'Optimized for' is set to 'recall'. The 'Searchable as' field is set to 'colorvect_l2'. An 'Add' button is at the bottom right of the configuration panel.

Vector Index



Search Service

A **Vector Query** can now search
for the top k-NN of a color

```
{
  "query": { "match_none": {} },
  "knn": [
    {
      "field": "colorvect_l2",
      "vector": [135, 204, 232],
      "k": 3
    }
  ],
  "fields": ["color"]
}
```

Vector Query

Couchbase uses the **Data Service** to store vectors, and the **Search Service** to index and query vectors

5-2. Hybrid Search



Hybrid SQL++ and Vector Search with Couchbase

This is a SQL++ query

Combining Vector Search query

And standard SQL++ criteria

```
SELECT color, brightness
FROM `vector-sample`.color.rgb AS t1
WHERE
  SEARCH(t1,
  {
    "query": { "match_none": {} },
    "knn": [{
      "field": "colorvect_l2",
      "vector": [135,204,232],
      "k": 3 }]
  })
  AND
  brightness >= 180 AND brightness <= 190
)
```



SQL++ is easy and familiar to developers



You can filter vector search results with other criteria



You don't have to run 2 separate databases, one for Documents and one for Vector Search!

Couchbase can run hybrid SQL++ and Vector Search queries to **facilitate application development**

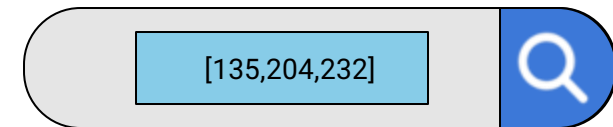
Comparison between Keyword Search and Vector Search



Keyword Search on the
description of the colors



A Keyword search looks for **terms** that match



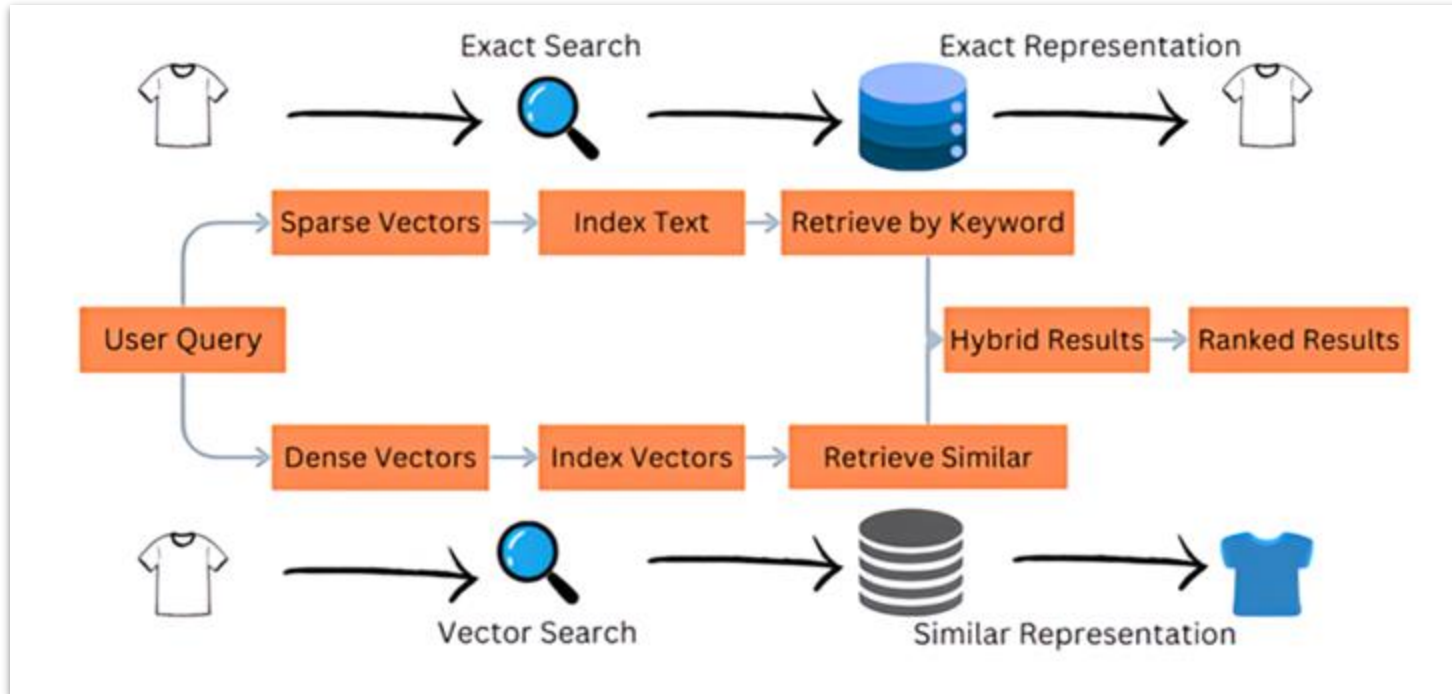
Vector Search on the
RGB vectors of the colors



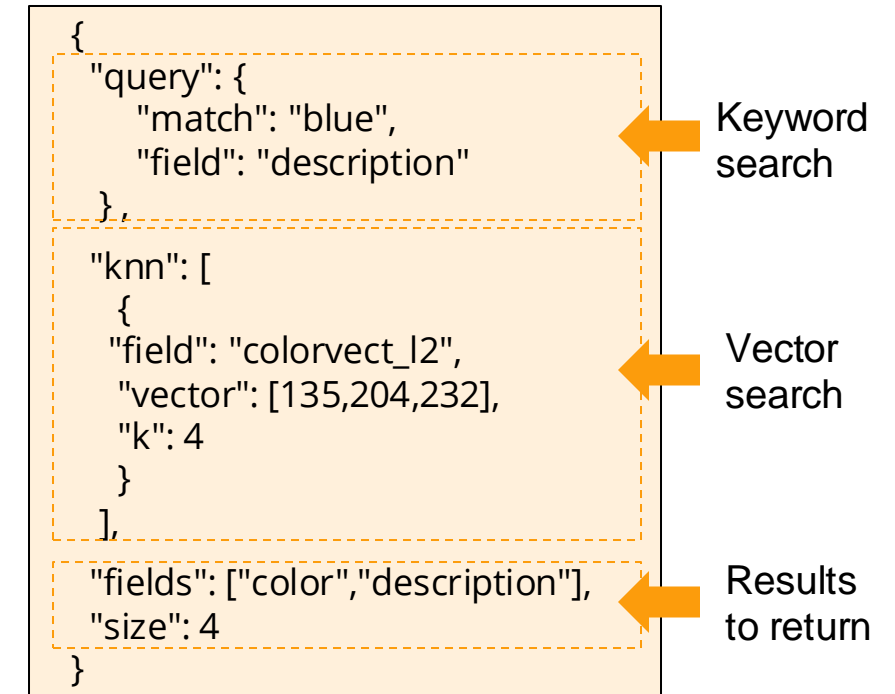
A Vector search looks for **similarity**

Hybrid Search to get the best of both worlds

Hybrid Search Architecture

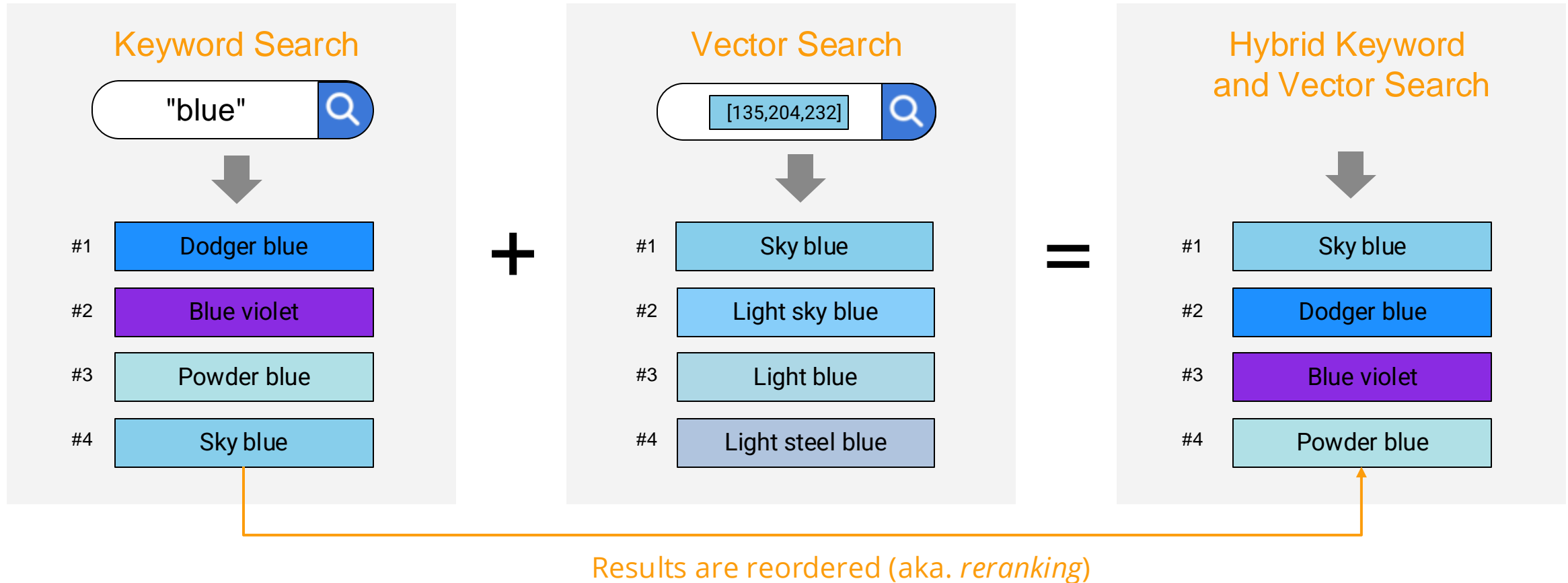


Hybrid Search with Couchbase



Vector search in conjunction with traditional Keyword search delivers the most complete and relevant results

Hybrid Keyword and Vector Search Example

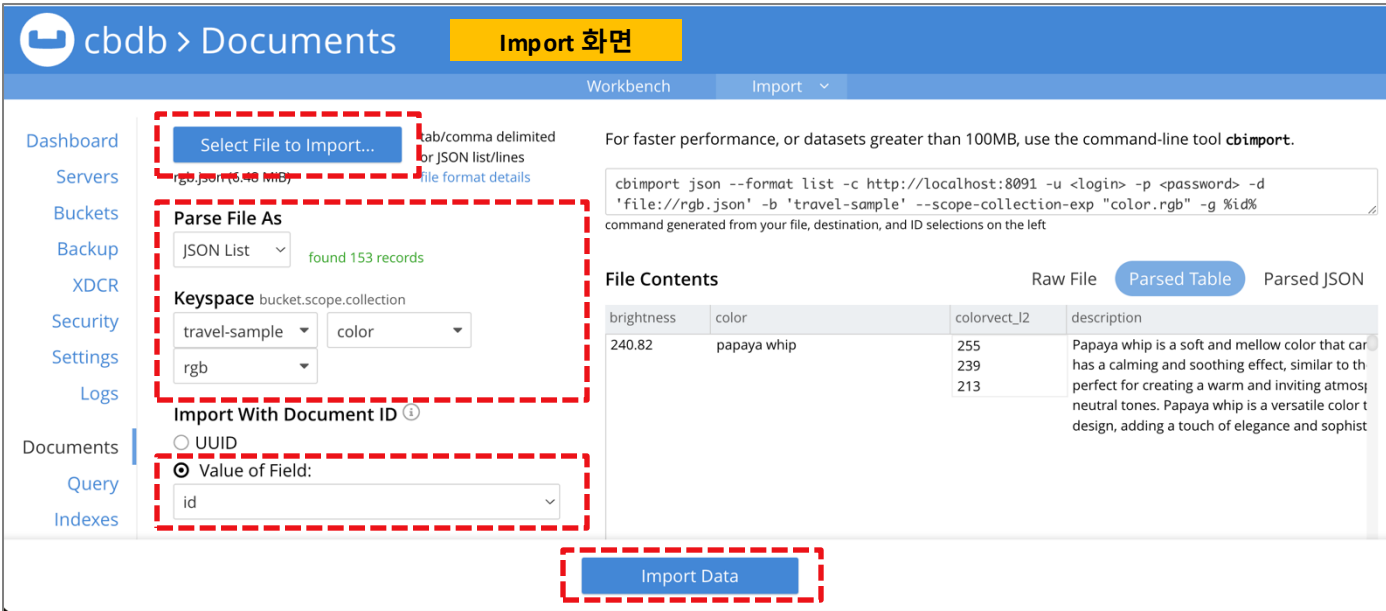


Results from the Keyword search are **boosted** if they appear in the Vector Search results

5-3. Vector Search 실습 : RGB



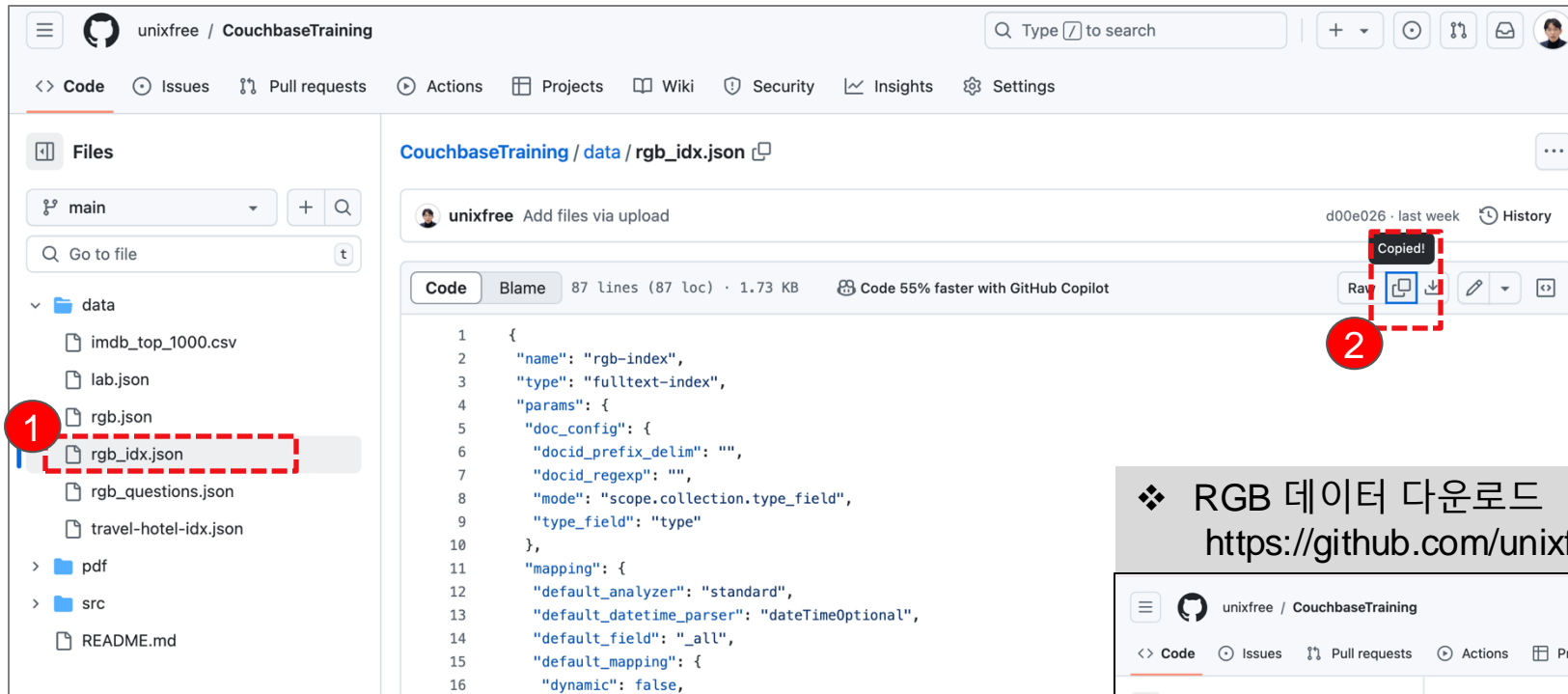
실습

| | 실습 항목 | 상세 실습 내용 | 기타 |
|---|----------------------|---|--|
| 1 | Scope, Collection 생성 | 1. travel-sample 버킷에 2. color Scope 생성 3. rgb Collection 생성 | travel-sample > color > rgb |
| 2 | Data Import | 1. Github에 있는 json 파일을 내 노트 북으로 다운로드 https://github.com/unixfree/CouchbaseTraining/blob/main/data/rgb.json 2. Couchbase Document 페이지 이동. 3. Import 선택 4. Import 수행, Document ID 는 id 로 지정 |  |
| 3 | 검색 Index 생성 | <ul style="list-style-type: none"> 21페이지에서 24페이지 참고하여 인덱스 생성 https://github.com/unixfree/CouchbaseTraining/blob/main/data/rgb_idx.json | |
| 4 | Vector Search 수행 | <ul style="list-style-type: none"> 26페이지에서 31페이지 참고하여 벡터 검색 수행 | |
| 5 | 검색 Index 수정 | <ul style="list-style-type: none"> 32페이지에서 33페이지 참고하여 벡터 검색 수행 | |
| 6 | Vector Search 수행 | <ul style="list-style-type: none"> 34페이지에서 38페이지 참고하여 벡터 검색 수행 | |

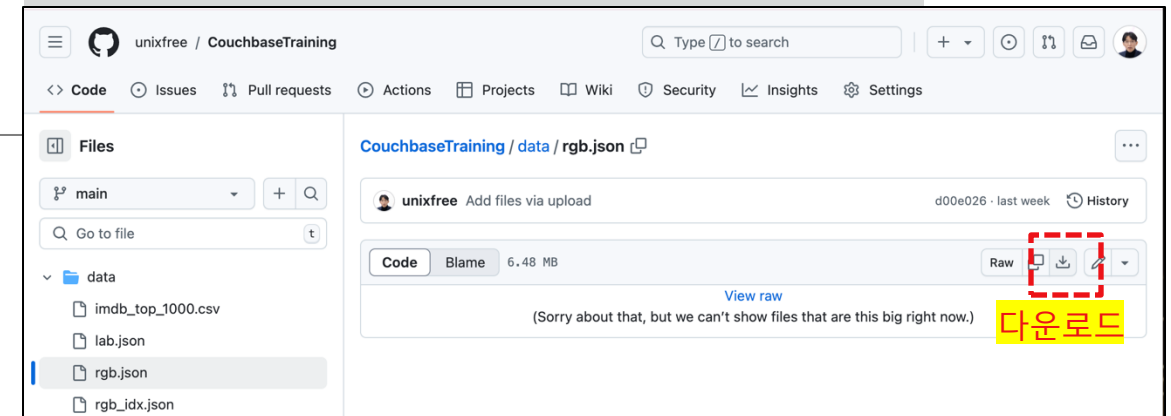
실습 > Inverted Index 생성

1. 인덱스 설정 파일 복사

<https://github.com/unixfree/CouchbaseTraining>



❖ RGB 데이터 다운로드
<https://github.com/unixfree/CouchbaseTraining>



실습 > Inverted Index 생성 (계속)

2. 인덱스 설정 파일 Import로 인덱스 생성

1) Search > 2) ADD INDEX > 3) Import > 4) 붙여넣기(control-v) > 5) Import

The screenshot illustrates the process of creating an inverted index in the Couchbase Search console. It is divided into three main sections with numbered callouts:

- Top Section (Full Text Search):** The header bar contains the 'ADD INDEX' button, highlighted with a red dashed box and labeled '2'.
- Middle Section (Add Index):** This section shows the 'Add Index' form. On the left sidebar, the 'Search' menu item is highlighted with a red dashed box and labeled '1'. The form includes fields for 'Index Name' and 'Bucket'. Below these fields, the 'Import' button is highlighted with a red dashed box and labeled '3'.
- Bottom Section (Import Index Modal):** A modal window titled 'Import Index' is open. It contains a text area where a JSON configuration file is pasted, indicated by a red dashed box and labeled '4' with the text '붙여넣기(control-v)'. At the bottom right of the modal, the 'Import' button is highlighted with a red dashed box and labeled '5'.

The JSON configuration file pasted into the modal is as follows:

```
{
  "store": {
    "indexType": "scorch",
    "segmentVersion": 15
  },
  "sourceType": "gocbcore",
  "sourceName": "travel-sample",
  "sourceParams": {},
  "planParams": {
    "maxPartitionsPerIndex": 64,
    "indexPartitions": 1,
    "numReplicas": 0
  }
}
```

실습 > Inverted Index 생성 (계속)

2. 인덱스 설정 파일 Import로 인덱스 생성

1) 아래와 같이 각 항목이 채워짐 > 2) 화면 스크롤 다운 후, 3) Create Index 클릭

cbdb > Full Text Search > Add Index

← BACK

Dashboard
Servers
Buckets
Backup
XDCR
Security
Settings
Logs
Documents
Query
Indexes
Search
Analytics
Eventing

Index Name

rgb-index

Bucket

travel-sample

Import

Data ingest from:

• Scope: color

• Collections: ["rgb"]

Index Definition Preview

copy to clipboard

{
 "name": "rgb-index",
}

Customize Index

Use non-default scope/collection

Scope

color

Type Identifier

Mappings

color.rgb | dynamic

description | text | en | index | store

colorvect_l2 | vector | index | dims: 3 | metric: l2_norm

color | text | en | index | store

default | disabled

Index Replicas

0

Index Partitions

1

Create Index

Cancel

Indexes
Search
Analytics
Eventing
Views

Mappings

+ Add Type Mapping

color.rgb | dynamic

description | text | en | index | store

colorvect_l2 | vector | index | dims: 3 | metric: l2_norm

color | text | en | index | store

default | disabled

Analyzers

Custom Filters

Date/Time Parsers

Advanced

"enabled": true,
"dynamic": false,
"fields": [
 {
 "name": "description",
 "type": "text",
 "analyzer": "en",
 "store": true,
 "index": true,
 "include_term_vectors": false,
 "include_in_all": false,
 "docvalues": false
 }
],
},
"colorvect_l2": {
 "enabled": true,
}

Confidential and Proprietary. Do not distribute without Couchbase consent. © Couchbase 2024. All rights reserved.

23

실습 > Inverted Index 생성 (계속)

2. 인덱스 설정이 완료.
인덱스가 만들어 지면 아래와 같이 화면(UI)가 보임.

The screenshot shows the 'Full Text Search' interface in the Couchbase web console. The left sidebar contains navigation links: Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, and Eventing. The main content area is titled 'Full Text Indexes' and contains a table with the following data:

| scoped access | index name | bucket | scope | collections | docs processed ⓘ | ingest ⓘ |
|---------------|------------------|---------------|-----------|-------------|------------------|----------|
| ✓ | rgb-index | travel-sample | color | rgb | 153 | idle |
| ✓ | travel-hotel-idx | travel-sample | inventory | hotel | 917 | idle |

Below the table, there are links for 'query syntax help', 'Show index definition JSON', and 'Search Index Stats'. At the bottom, there is a 'Full Text Aliases' section with a '+ Add Alias' button. The 'rgb-index' row is highlighted, and a red dashed box is drawn around the search input field 'search this index...' and the search button. Red arrows point from this box to a yellow callout box on the left and to a yellow callout box on the right. The callout box on the right has two arrows pointing to it, one from the 'ingest' status and one from the 'delete' icon.

검색어나, 검색
Query 입력창

인덱스 확인/수정

인덱스 삭제

실습 > color.rgb 데이터 설명

```
{
  "id": "#000080",
  "color": "navy",
  "brightness": 14.592,
  "verbs": ["deep", "rich", "sophisticated"],
  "colorvect_l2": [0, 0, 128],
  "description": "Navy is a deep, rich color that exudes sophistication. It is
    a dark shade of blue that is often associated with authority, stability,
    and elegance. Navy is a versatile color that can be both bold and
    understated, making it a popular choice in fashion and interior design. It
    is a timeless color that never goes out of style and adds a touch of
    sophistication to any look or space.",
  "embedding_model": "text-embedding-ada-002-v2",
  "embedding_vector_dot": [
    0.0021118249278515577,
    -0.005944395903497934,
    ... 1533 omitted for space
    -0.018224267289042473
  ],
  "wheel_pos": "other"
}
```

- **id**: the hex code of the color
- **color**: the name of the color
- **brightness**: a calculation of the brightness to the human eye
- **colorvect_l2**: vector based on the RGB color
- **description**: a text describing the color
- **embedding_model**: the model used to encode the embedding_vector_dot vector
- **embedding_vector_dot** : vector based on the field “description” encoded via the text-embedding-ada-002 OpenAI model
- **verbs**: list of qualifiers

실습 > Vector Search (RGB)

```
{
  "query": { "match_none": {} },
  "knn": [
    { "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 }
  ],
  "fields": ["color"]
}
```

cbdb > Full Text Search > Search Results

query syntax help

Timeout (msecs) 0

Consistency Level

Consistency Vectors {}

JSON for Query Request ☐ show command-line curl example

```
{
  "query": {
    "match_none": {}
  },
  "knn": [
    {
      "field": "colorvect_l2",
      "vector": [
        0,
        0,
        128
      ],
      "k": 3
    }
  ],
  "fields": [
    "color"
  ],
  "explain": true,
  "size": 10,
  "from": 0
}
```

Edit

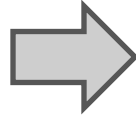
Copy to clipboard

완전한 검색 Query 확인

완전한 검색 Query 수정

실습 > Vector Search (RGB)

```
{
  "query": { "match_none": {} },
  "knn": [
    { "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 }
  ]
}
```



{ "query": { "match_none": {} }, "knn": [{ "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 }] } ☐ show advanced query settings

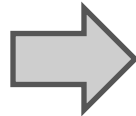
[query syntax help](#)

Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring 3 results (2ms server-side)

| | |
|------------|--------------------------|
| 1. #000080 | [3.4028234663852886e+38] |
| 2. #00008B | [0.008264462809917356] |
| 3. #191970 | [0.0006640106241699867] |

```
{
  "query": { "match_none": {} },
  "knn": [
    { "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 }
  ],
  "fields": ["color"]
}
```



{ "query": { "match_none": {} }, "knn": [{ "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 }] } ☐ show advanced query settings

[query syntax help](#)

Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

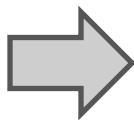
☐ Explain Scoring 3 results (<1ms server-side)

| | |
|-----------------|--------------------------|
| 1. #000080 | [3.4028234663852886e+38] |
| color | |
| ◦ navy | |
| 2. #00008B | [0.008264462809917356] |
| color | |
| ◦ dark blue | |
| 3. #191970 | [0.0006640106241699867] |
| color | |
| ◦ midnight blue | |

실습 > Vector Search (RGB)

- 유사도에 따른 결과 수 지정 : 3

```
{
  "query": {"match_none": {}},
  "knn": [ {"field": "colorvect_l2", "vector": [0,0,64], "k": 3}],
  "fields": ["color"]
}
```



{ "query": {"match_none": {}}, "knn": [{"field": "colorvect_l2", "vector": [0,0,64], "k": 3}], "fields": ["color"] } ☐ show advanced query settings
[query syntax help](#)

Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring 3 results (2ms server-side)

1. #191970 [0.0002813731007315701]

color
◦ midnight blue

2. #000000 [0.000244140625]

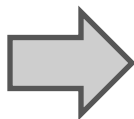
color
◦ black

3. #000080 [0.000244140625]

color
◦ navy

- 유사도에 따른 결과 수 지정 : 153

```
{
  "query": {"match_none": {}},
  "knn": [ {"field": "colorvect_l2", "vector": [0,0,64], "k": 153}],
  "fields": ["color"]
}
```



{ "query": {"match_none": {}}, "knn": [{"field": "colorvect_l2", "vector": [0,0,64], "k": 153}], "fields": ["color"] } ☐ show advanced query settings
[query syntax help](#)

Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring 153 results (4ms server-side)

1. #191970 [0.0002813731007315701]

color
◦ midnight blue

2. #000080 [0.000244140625]

color
◦ navy

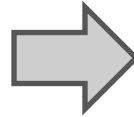
3. #000000 [0.000244140625]

color
◦ black

실습 > Vector Search (RGB)

- 없는 벡터 값

```
{
  "query": {"match_none": {}},
  "knn": [
    {"field": "colorvect_l2", "vector": [1.241999836687228,-
    8.105597595174153,-9.030840590925482],
    "k": 3} ],
  "fields": ["color"]
}
```



{ "query": {"match_none": {}}, "knn": [{"field": "colorvect_l2", "vector": [1.241999836687228,-8.105597595174153,-9.030840590925482], "k": 3}], "fields": ["color"] } ☐ show advanced query settings

[query syntax help](#)

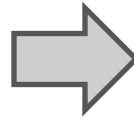
Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring 3 results (2ms server-side)

1. #000000 [0.006720458914149972]
color
◦ black
2. #006400 [0.00008496235230220179]
color
◦ dark green
3. #500050 [0.0000704472727617959]
color
◦ dark purple

- 벡터 차원(dimension)이 다를 때

```
{
  "query": {"match_none": {}},
  "knn": [
    {"field": "colorvect_l2",
    "vector": [0,64],
    "k": 3} ],
  "fields": ["color"]
}
```



{ "query": {"match_none": {}}, "knn": [{"field": "colorvect_l2", "vector": [0,64], "k": 3}], "fields": ["color"] } ☐ show advanced query settings

[query syntax help](#)

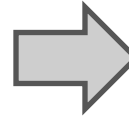
Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

No results found for your query.
Please check your search term(s) and/or use the syntax help link under the search field.

실습 > Vector Search (RGB)

• OR 조건

```
{
  "query": {"match_none": {}},
  "knn": [
    { "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 },
    { "field": "colorvect_l2", "vector": [0, 0, 64], "k": 3 }
  ],
  "fields": ["color"]
}
```



{ "query": {"match_none": {}}, "knn": [{ "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 }, { "field": "colorvect_l2", "vector": [0, 0, 64], "k": 3 }] } ☐ show advanced query settings

[query syntax help](#)

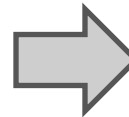
Results from: **rgb-index**
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring 4 results (2ms server-side)

1. #000080 [3.4028234663852886e+38]
color
◦ navy
2. #00008B [0.00584385769575659]
color
◦ dark blue
3. #191970 [0.0006684872427012883]
color
◦ midnight blue
4. #000000 [0.00017263349150062194]
color
◦ black

• AND 조건

```
{
  "query": {"match_none": {}},
  "knn": [
    { "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 },
    { "field": "colorvect_l2", "vector": [0, 0, 64], "k": 3 }
  ],
  "knn_operator": "and",
  "fields": ["color"]
}
```



{ "query": {"match_none": {}}, "knn": [{ "field": "colorvect_l2", "vector": [0, 0, 128], "k": 3 }, { "field": "colorvect_l2", "vector": [0, 0, 64], "k": 3 }] } ☐ show advanced query settings

[query syntax help](#)

Results from: **rgb-index**
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring 2 results (<1ms server-side)

1. #000080 [3.4028234663852886e+38]
color
◦ navy
2. #191970 [0.0006684872427012883]
color
◦ midnight blue

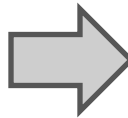
10 ▾

실습 > Vector Search (RGB)

• Boost 조건

❖ 1보다 작으면, 가중치 감소시키고, 1보다 크면 가중치 증가시킴

```
{
  "query": {"match_none": {}},
  "knn": [
    { "field": "colorvect_l2", "vector": [0, 0, 127], "k": 3, "boost": 0.1},
    { "field": "colorvect_l2", "vector": [0, 99, 0], "k": 3, "boost": 4.0}
  ],
  "fields": ["color"]
}
```



Query: {"query": {"match_none": {}}, "knn": [{"field": "colorvect_l2", "vector": [0, 0, 127], "k": 3, "boost": 0.1}, {"field": "colorvect_l2", "vector": [0, 99, 0], "k": 3, "boost": 4.0}], "fields": ["color"]}

Results from: rgb-index

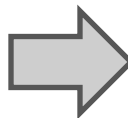
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring 6 results (<1ms server-side)

1. #006400 [0.9996876464081226]
color
 ◦ dark green
2. #000080 [0.024992191160203066]
color
 ◦ navy
3. #008000 [0.001188689234730229]
color
 ◦ green
4. #228B22 [0.0002555438768936919]
color
 ◦ forest green
5. #00008B [0.00017355688305696574]
color
 ◦ dark blue
6. #191970 [0.000016943858413696994]
color
 ◦ midnight blue

• Hybrid Search 조건

```
{
  "query": {
    "field": "brightness", "min": 70, "max": 80,
    "inclusive_min": false, "inclusive_max": true },
  "knn": [ {"field": "colorvect_l2", "vector": [0.0, 0.0, 108.0], "k": 5} ],
  "fields": ["color", "brightness"],
  "size": 5
}
```



Query: {"query": {"field": "brightness", "min": 70, "max": 80, "inclusive_min": false, "inclusive_max": true }, "knn": [{"field": "colorvect_l2", "vector": [0.0, 0.0, 108.0], "k": 5}], "fields": ["color", "brightness"], "size": 5}

Results from: rgb-index

[scoped access - bucket:travel-sample; scope:color]

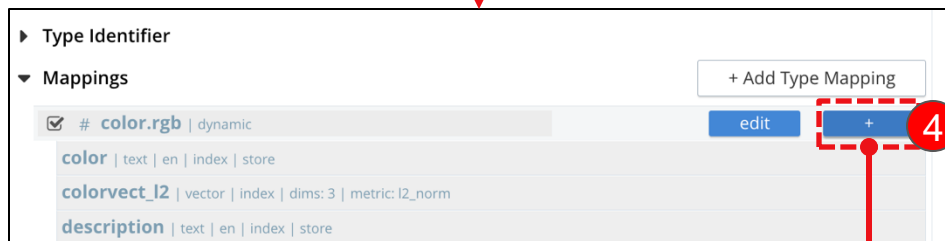
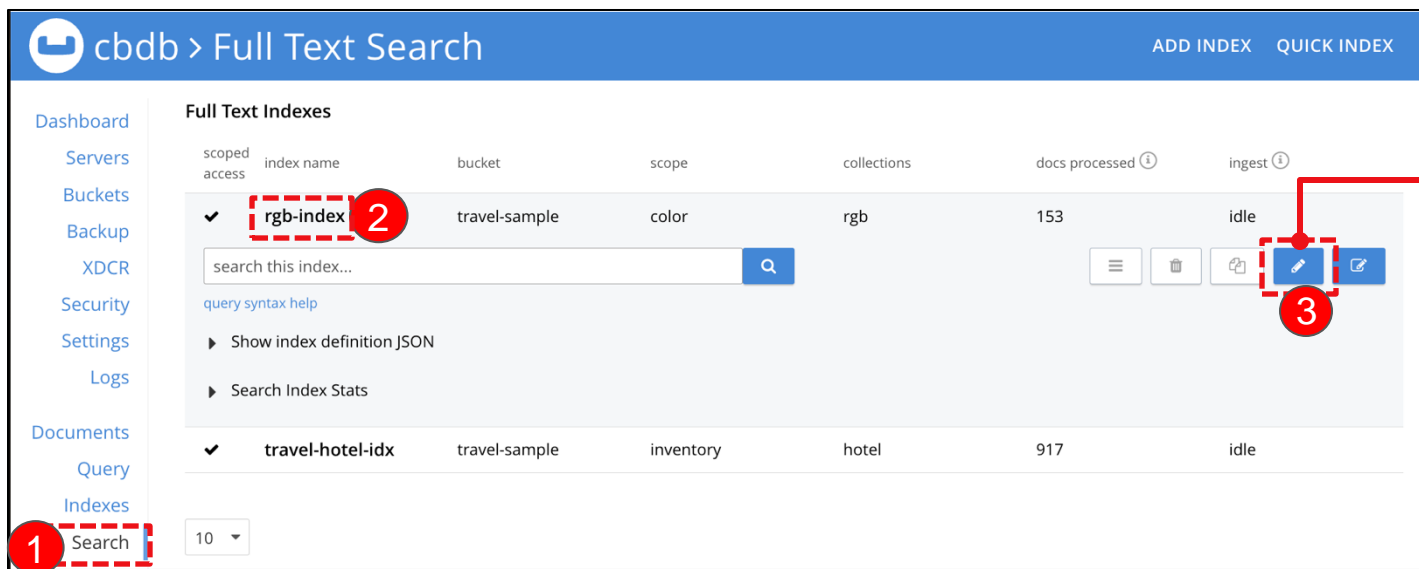
☐ Explain Scoring 5 results (<1ms server-side)

1. #000080 [0.0025]
color
 ◦ navy
2. #00008B [0.001040582726326743]
color
 ◦ dark blue
3. #191970 [0.0007898894154818325]
color
 ◦ midnight blue
4. #4B0082 [0.0001636929120969062]
color
 ◦ indigo
5. #500050 [0.00013919821826280623]
color
 ◦ dark purple

실습 > 인덱스 변경

1. 인덱스 설정 파일 Import로 인덱스 생성

1) Search > 2) rgb-index > 3) 수정(연필) 클릭 > 마우스를 color.rgb로 이동하면 edit 와 + 가 나타남, 4) + 를 선택, 5) insert child field 선택



실습 > 인덱스 변경

2. 새로운 필드에 대한 인덱스 추가

1) field 에 **brightness** 입력 > 2) type을 *number* 선택 3) *index*, *store* 선택, 4) **ok** 를 선택, 5) **Update index** 를 클릭

▶ Type Identifier

▼ Mappings

+ Add Type Mapping

☒ # **color.rgb** | dynamic

1 field **brightness**

2 type **number**

3 ☒ index ☒ store ☐ include in_all field ☐ docvalues

searchable as **brightness**

color | text | en | index | store

colorvect_l2 | vector | index | dims: 3 | metric: l2_norm

description | text | en | index | store

☐ # **default** | disabled

ok
cancel
delete

▶ Type Identifier

▼ Mappings

+ Add Type Mapping

☒ # **color.rgb** | dynamic

brightness | number | index | store

color | text | en | index | store

colorvect_l2 | vector | index | dims: 3 | metric: l2_norm

description | text | en | index | store

☐ # **default** | disabled

▶ Analyzers

▶ Custom Filters

▶ Date/Time Parsers

▶ Advanced

Index Replicas ⓘ
0

Index Partitions ⓘ
1

See documentation on creating indexes [here](#).

5 Update Index Cancel

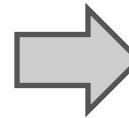
실습 > Vector Search (RGB)

• Hybrid Search, 결과 출력 변경

```
{
  "query": {
    "field": "brightness", "min": 70, "max": 80,
    "inclusive_min": false, "inclusive_max": true },
  "knn": [ {"field": "colorvect_l2", "vector": [0, 0, 108], "k": 5} ],
  "fields": ["color", "brightness"],
  "size": 5
}
```

• Vector Search, 결과 출력 변경

```
{
  "query": {"match_none": {}},
  "knn": [ {"field": "colorvect_l2", "vector": [0,0,108], "k": 5} ],
  "fields": ["color", "brightness"]
}
```



colorvect_l2, vector: [0, 0, 108], k: 5, fields: [color, brightness], size: 5

Results from: rgb-index

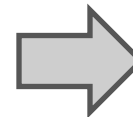
[scoped access - bucket:travel-sample; scope:color]

12 results (1ms server-side)

1. #B000B0

brightness: 72.688

color: dark lavender



{ "query": { "match_none": {} }, "knn": [{ "field": "colorvect_l2", "vector": [0,0,108], "k": 5 }] }

Results from: rgb-index

[scoped access - bucket:travel-sample; scope:color]

5 results (2ms server-side)

1. #000080

brightness: 14.592

color: navy

2. #00008B

brightness: 15.846

color: dark blue

3. #191970

brightness: 34.918

color: midnight blue

4. #4B0082

brightness: 37.245

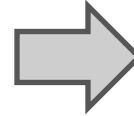
color: indigo

5. #500050

실습 > Vector Search (RGB)

- Hybrid Search, 결과 출력 변경

```
{
  "query": {
    "field": "brightness", "min": 10, "max": 20,
    "inclusive_min": false, "inclusive_max": true },
    "knn": [ {"field": "colorvect_l2", "vector": [0.0, 0.0, 108.0], "k": 5} ],
    "fields": ["color", "brightness"],
    "size": 5
  }
```



{ "query": { "match_none": {}, "knn": [{ "field": "colorvect_l2", "vector": [0.0, 108], "k": 5 }] } }

query syntax help

Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

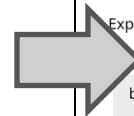
☐ Explain Scoring

5 results (2ms server-side)

1. #000080 [0.0025]
brightness
◦ 14.592
color
◦ navy
2. #00008B [0.001040582726326743]
brightness
◦ 15.846
color
◦ dark blue
3. #191970 [0.0007898894154818325]
brightness
◦ 34.918
color
◦ midnight blue
4. #4B0082 [0.0001636929120969062]
brightness
◦ 37.245
color
◦ indigo
5. #500050 [0.00013919821826280623]
brightness

- Hybrid Search, 결과 출력 변경

```
{
  "query": { "match": "freedom", "field": "description" },
  "fields": ["color", "brightness", "description"],
  "size": 4
}
```



{ "query": { "match": "freedom", "field": "description" } }

query syntax help

Results from: rgb-index
[scoped access - bucket:travel-sample; scope:color]

☐ Explain Scoring

1 results (<1ms server-side)

#CEEB

brightness
◦ 188.077
color
◦ sky blue
description
◦ Sky blue is a calming and serene color that evokes feelings of tranquility. It is a light shade of blue that resembles the color of the sky on a clear day. The color is often associated with peace, relaxation, and a sense of openness. It can also represent a sense of freedom and endless possibilities, as the sky seems to stretch on forever. Sky blue is a refreshing and soothing color that can bring a sense of calmness to any space.

실습 > SQL++ Hybrid Query (RGB)

- Hybrid Search, 결과 출력 변경

```
SELECT color, brightness
FROM `travel-sample`.color.rgb AS t1
WHERE brightness <= 20 AND brightness>=10
AND
SEARCH(t1, {
  "query": { "match_none": {} },
  "knn": [{ "field": "colorvect_l2", "vector": [0.0, 0.0, 108.0], "k": 3 }]
})
```

The screenshot shows the Couchbase Query Editor interface. The left sidebar contains navigation links: Dashboard, Servers, Buckets, Backup, XDCR, Security, Settings, Logs, Documents, Query, Indexes, Search, Analytics, Eventing, and Views. The main area is titled 'Query Editor' and shows a hybrid SQL++ query. The query is highlighted with a red dashed box. Below the query editor, there are buttons for 'Execute', 'Run as TX', 'Index Advisor', and 'Explains'. The status bar indicates 'success just now | 42.8ms | 2 docs | 127 bytes'. The 'Results' section shows the output in JSON format, also highlighted with a red dashed box. The results are two JSON documents.

```
1 SELECT color, brightness
2 FROM `travel-sample`.color.rgb AS t1
3 WHERE brightness <= 20 AND brightness>=10
4 AND
5 SEARCH(t1, {
6   "query": { "match_none": {} },
7   "knn": [{ "field": "colorvect_l2", "vector": [0.0, 0.0, 108.0], "k": 3 }]
8 })
9
```

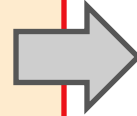
Results (JSON):

```
1 [
2   {
3     "color": "dark blue",
4     "brightness": 15.846
5   },
6   {
7     "color": "navy",
8     "brightness": 14.592
9   }
10 ]
```

실습 > SQL++ Hybrid Query (RGB)

• Hybrid Search, 결과 출력 변경

```
SELECT color, brightness, description
FROM `travel-sample`.color.rgb AS t1
WHERE
SEARCH(t1, {
  "query": {"match": "freedom", "field": "description"},
  "fields": ["color", "description"], "size": 4}
)
```



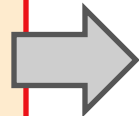
Execute Run as TX Index Advisor Explain success just now | 26.5ms | 1 docs | 528 bytes

Results Table JSON Chart Plan Plan Text Advice

```
1- [
2- {
3-   "color": "sky blue",
4-   "brightness": 188.077,
5-   "description": "Sky blue is a calming and serene color that evokes feelings of tranquility. It is a light shade of blue
that resembles the color of the sky on a clear day. The color is often associated with peace, relaxation, and a sense of
openness. It can also represent a sense of freedom and endless possibilities, as the sky seems to stretch on forever. Sky
blue is a refreshing and soothing color that can bring a sense of calmness to any space."
6- }
7-
```

• Hybrid Search, 결과 출력 변경

```
SELECT color, brightness, description
FROM `travel-sample`.color.rgb AS t1
WHERE
SEARCH(t1, {
  "query": {"match": "freedom", "field": "description"},
  "knn": [ { "field": "colorvect_l2", "vector": [135,204,232], "k": 4} ],
  "fields": ["color", "description"], "size": 4}
)
```



Execute Run as TX Index Advisor Explain success just now | 14.4ms | 4 docs | 2403 bytes

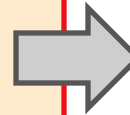
Results Table JSON Chart Plan Plan Text Advice

```
1- [
2- {
3-   "color": "sky blue",
4-   "brightness": 188.077,
5-   "description": "Sky blue is a calming and serene color that evokes feelings of tranquility. It is a light shade of
blue that resembles the color of the sky on a clear day. The color is often associated with peace, relaxation, and a sense
of openness. It can also represent a sense of freedom and endless possibilities, as the sky seems to stretch on forever.
Sky blue is a refreshing and soothing color that can bring a sense of calmness to any space."
6- },
7- {
8-   "color": "light blue",
9-   "brightness": 204.739,
10-  "description": "Light blue is a soft and soothing color that evokes feelings of calmness and tranquility. It is often
associated with clear skies and calm waters, giving it a refreshing and airy quality. This color is perfect for creating a
peaceful and serene atmosphere, making it a popular choice for bedrooms and spa-like spaces. Its gentle hue can also bring
a sense of clarity and clarity to the mind, making it a great color for promoting relaxation and mental clarity. Light
blue is a versatile color that can be used in a variety of settings, from home decor to fashion, and is loved for its
ability to create a sense of balance and harmony."
11- },
12- {
13-   "color": "light steel blue",
14-   "brightness": 192.984,
15-   "description": "Light steel blue is a soft and delicate color that exudes a sense of calmness and tranquility. It is a
```

실습 > SQL++ Hybrid Query (RGB)

- Hybrid Search, 결과 출력 변경

```
SELECT color, brightness, description
FROM `travel-sample`.color.rgb AS t1
WHERE brightness >= 180 AND brightness <= 190
AND
SEARCH(t1, {
  "query": {"match": "freedom", "field": "description"},
  "knn": [ { "field": "colorvect_l2", "vector": [135,204,232], "k": 4} ],
  "fields": ["color","description"],"size": 4}
)
```



Execute Run as TX Index Advisor Explain ✓ success just now | 13.1ms | 2 docs | 1111 bytes format

Results Table JSON Chart Plan Plan Text Advice

```
1 [
2   {
3     "color": "sky blue",
4     "brightness": 188.077,
5     "description": "Sky blue is a calming and serene color that evokes feelings of tranquility. It is a light shade of
6   },
7   {
8     "color": "light sky blue",
9     "brightness": 189.787,
10    "description": "Light sky blue is a soft and delicate color that evokes a sense of tranquility and peace. It is a
11  }
12 ]
```



수고하셨습니다.



paul.son@couchbase.com

www.couchbase.com

cloud.couchbase.com



Couchbase