Couchbase Capella ワークショップ

ラボハンドブック

ラボ **3: SDK**

概要

Python SDK を使用して Couchbase Capella クラスターに接続し、Couchbase Server の主要な機能の実行を確認します。

このラボでは、参加者の環境に Python が既にインストールされていることを前提としています。 Python のバージョンは、3.5 以上を使用することをお勧めします。

準備

ステップ 1: 作業ディレクトリ作成

作業ディレクトリを作成します（ディレクトリ名は、ご自身で決めていただいて構いません）。この作業ディレクトリは、Python コードが置かれるディレクトリになります。

```
$ mkdir ~/cbc_workshop
```

作成した作業ディレクトリに移動します。

```
$ cd ~/cbc_workshop/
```

ステップ 2: Python インストール確認

```
$ python -V

Python 3.8.5
```

注: Python を使用する場合は、仮想環境を使用することを強くお勧めします。 これは、Python プロジェクトを整理しておく最も簡単な方法です(長期的には貴重な時間を節約できます)。

ステップ 3: Couchbase ライブラリーインストール

```
$ pip install couchbase

Collecting couchbase
  Downloading couchbase-3.0.9.tar.gz (1.5 MB)
     |████████████████████████████████| 1.5 MB 4.1 MB/s
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Installing backend dependencies ... done
     Preparing wheel metadata ... done
Requirement already satisfied: mypy-extensions in
~/.virtualenvs/cbc_workshop/lib/python3.8/site-packages (from couchbase) (0.4.3)
```

```
Requirement already satisfied: six in
~/.virtualenvs/cbc_workshop/lib/python3.8/site-packages (from couchbase)
(1.15.0)
Requirement already satisfied: attrs>=19.1.0; python_version > "3.5" in
~/.virtualenvs/cbc_workshop/lib/python3.8/site-packages (from couchbase)
(20.3.0)
Requirement already satisfied: wrapt>=1.11.2 in
~/.virtualenvs/cbc_workshop/lib/python3.8/site-packages (from couchbase)
(1.12.1)
Requirement already satisfied: pyrsistent>=0.15.2 in
~/.virtualenvs/cbc_workshop/lib/python3.8/site-packages (from couchbase)
(0.17.3)
Building wheels for collected packages: couchbase
  Building wheel for couchbase (PEP 517) ... done
  Created wheel for couchbase: filename=couchbase-3.0.9-cp38-cp38-
macosx_10_15_x86_64.whl size=1337870
sha256=27e3f7d492bbe631dbd1ba0f20ae2145f628a99f9577b220780176a0f997b4fb
  Stored in directory:
/private/var/folders/fj/_yt793dj6d9f976tf6mz_nt40000gr/T/pip-ephem-wheel-cache-
iapa7cgg/wheels/84/18/5c/0d8d47d9811eb569ff3ef9af86447e761ca9052eb4aea10058
Successfully built couchbase
Installing collected packages: couchbase
Successfully installed couchbase-3.0.9
```

ステップ 4: Python 実行確認

Python では、モジュールは ".py"という接尾辞を持ちます。新しいファイルを作成して、cbc_workshop.py という名前で保存します。 はじめに、次のコードを使用します。

```
0:  if __name__ == '__main__':
1:      try:
2:          print('Hello Couchbase Cloud Workshop!')
3:      except Exception as ex:
4:          import traceback
5:          traceback.print_exc()
```

cbc_workshop.py 保存した後、モジュールを実行すると、次の出力が出力されます。

```
$ python cbc_workshop.py

Hello Couchbase Cloud Workshop!
```

演習では、上記ファイルを編集しながら機能を確認していきます。
このファイルからコードをコピーした場合、コピー・ペーストに伴う、思わぬ間違いが発生する可能性があるため、それぞれのコード例の後に、実行確認ずみのファイルへのリンクを掲載しています。

演習

ステップ 1: Couchbase Capella クラスター接続

1a: Python プログラム実行環境の IP アドレスを確認

このサイトを表示して確認するか、次の *curl* コマンドを使用します。 この後クラスターに対して、この IP アドレスからの接続を許可する設定を行います。

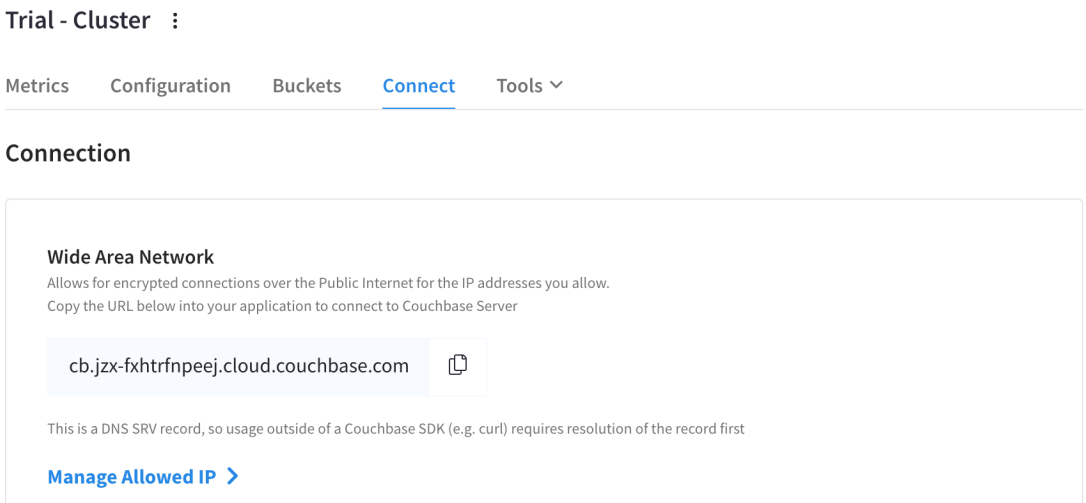
```
$ curl ifconfig.me

***.**.**.**
```

1b: Couchbase Capella クラスターに IPv4 アドレスを追加します。

次の手順で、Couchbase Capella に IPv4 アドレスを追加します。

1. *Couchbase Capella コントロールプレーン*に移動します。
2. 接続するクラスターの画面を表示します。
3. **[Connect]** タブを表示します。
4. **[Connection]**セクションの**[Wide Area Network]**セル中の**[Manage Allowed IP]**リンクをクリックします。



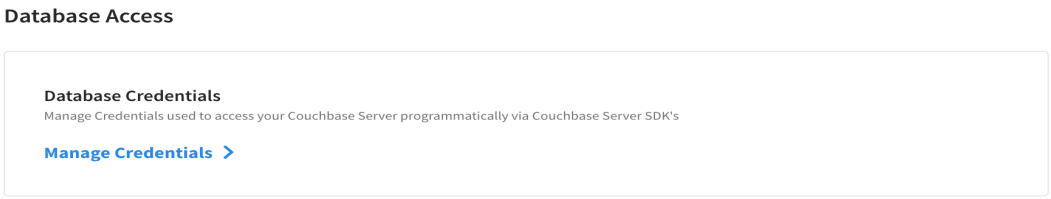
5. 画面右上の**[Add Allowed IP]**リンクをクリックします。
6. **[Add Permanent IP]**セル中の **[IP Address/CIDR Block]** フィールドに先程確認した IP アドレスを貼り付けます。
7. **[Add IP]**ボタンを押下します。**[Allowed IP]**リストに追加した IP が追加されます。

1c: データベース資格情報作成

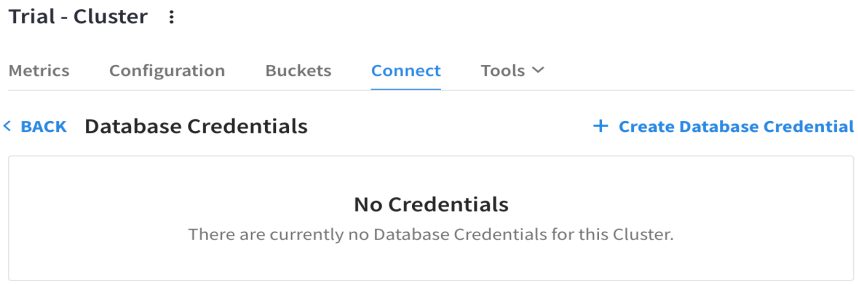
ここでは、travel-sample バケットの読み取り/書き込みアクセス権を持つデータベース資格情報を作成します。

cloud.couchbase.com からサインインします。

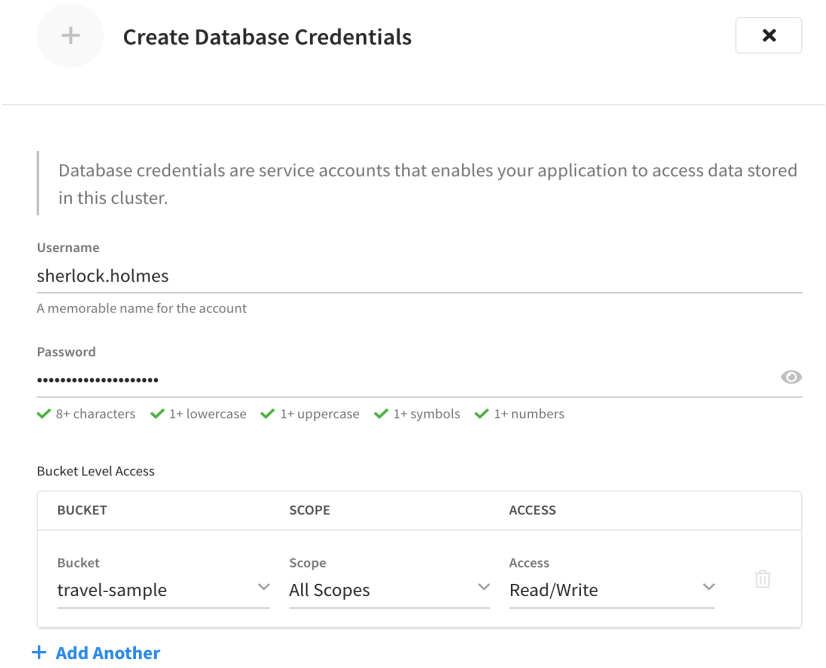
クラスター画面の**[Connect]**タブを開きます。**[Database Access]** セクションの**[Database Credentials]**セル中の**[Manage Credentials]**リンクをクリックします。



画面右上の**[Create Database Credential]**をクリックします。

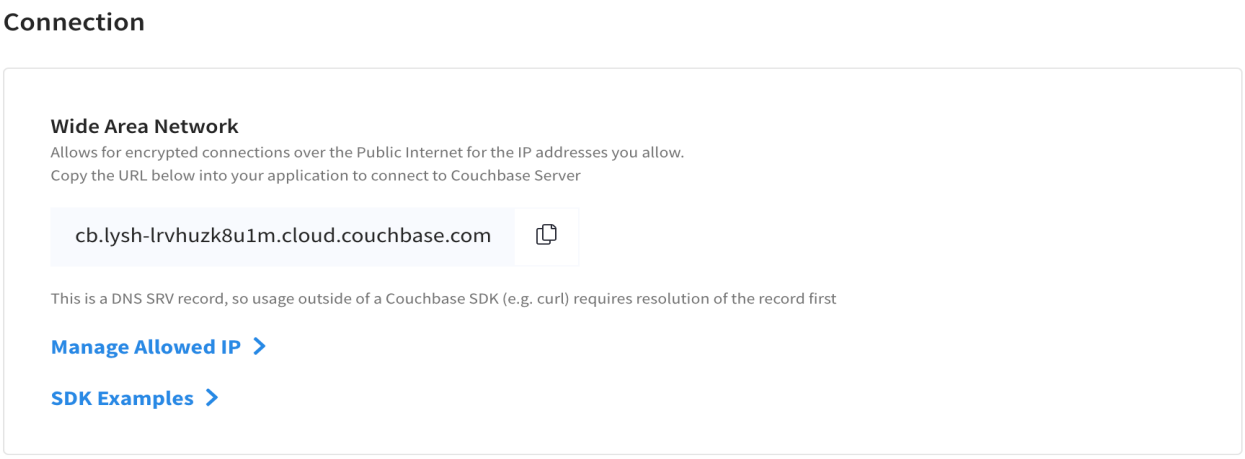


[Username]に"sherlock.homes"、**[Password]**に"P@ssw0rd"を入力します（ここで示した入力値はサンプルプログラムで利用されているものです）。
[Bucket Level Access]で、下記イメージの通り travel-sample バケットの全てのスコープに Read/Write アクセスを設定します。



1d: 接続文字列

[Connect] タブの**[Connection]**セクションの **[Wide Area Network]** セルから、SDK 接続文字列に利用する URL をコピーします。



1e: Python モジュール更新

接続ロジックを追加します。 更新後のプログラムは以下のようになります。 接続管理の詳細については、ドキュメントを参照してください 。
適宜、cloud_endpoint を接続対象のクラスターの URL に置換してください。

```
0:  from couchbase.cluster import Cluster, ClusterOptions
1:  from couchbase.auth import PasswordAuthenticator
2:
3:  if __name__ == '__main__':
4:      try:
5:          # Cluster specific information
6:          cloud_endpoint = \
7:                       'cb.jzx-fxhtrfnpeej.cloud.couchbase.com'
8:          username = 'sherlock.holmes'
9:          password = 'P@ssw0rd'
10:         bucket_name = 'travel-sample'
11:
12:         # connect to cluster
13:         conn_str = 'couchbases://{}?ssl=no_verify'.format(cloud_endpoint)
14:         cluster_opts = ClusterOptions(PasswordAuthenticator(username,
15:                                                             password))
16:         cluster = Cluster(conn_str, cluster_opts)
17:         bucket = cluster.bucket(bucket_name)
18:         collection = bucket.default_collection()
19:
20:     except Exception as ex:
21:         import traceback
22:         traceback.print_exc()
```

ファイル：https://github.com/YoshiyukiKono/cb-dev-days-capella/blob/main/python/cbc_workshop_connect.py

ステップ 2: CRUD (K/V 操作)

*k/V*接続ロジックを追加します。 更新後のプログラムは以下のようになります。
キー値の操作とサブドキュメント操作に関する詳細については、ドキュメントを参照してください。

```
0:  from pprint import pprint
1:
2:  from couchbase.cluster import Cluster, ClusterOptions
3:  from couchbase.auth import PasswordAuthenticator
4:  import couchbase.subdocument as SD
5:
6:
7:  if __name__ == '__main__':
8:      try:
9:          # Cluster specific information
10:         cloud_endpoint = \
11:                     'cb.jzx-fxhtrfnpeej.cloud.couchbase.com'
12:         username = 'sherlock.holmes'
13:         password = 'P@ssw0rd'
14:         bucket_name = 'travel-sample'
15:
16:         # connect to cluster
17:         conn_str = 'couchbases://{}?ssl=no_verify'.format(cloud_endpoint)
18:         cluster_opts = ClusterOptions(PasswordAuthenticator(username,
19:                                       password))
20:         cluster = Cluster(conn_str, cluster_opts)
21:         bucket = cluster.bucket(bucket_name)
22:         collection = bucket.default_collection()
23:
24:         #KV operation data
25:         test_key = 'testDoc::0'
26:         test_doc =  {
27:             'type': 'testDoc',
28:             'info': 'This is a test',
29:             'address': {
30:                 'home': {
31:                     'address1': '8575 Lewis Springs Mountains',
32:                     'city': 'West Mohammed',
33:                     'state': 'WI',
34:                     'zipCode': '40243-4741',
35:                     'country': 'CA'
36:                     }
37:                 }
38:             }
39:         sd_path = 'address.home.address1'
40:         sd_update = '8575 Lewis Springs Mountains Blvd'
41:
42:         #KV - insert
43:         ins_result = collection.insert(test_key, test_doc)
44:         print('\nInserted doc w/ key: {}; CAS: {}\n'.format(test_key,
45:                                                            ins_result.cas))
46:
47:         #KV - get
48:         get_result = collection.get(test_key)
49:         pprint(get_result.content)
50:
51:         #KV - upsert
52:         ins_result = collection.upsert(test_key, test_doc)
53:         print('\nUpserted doc w/ key: {}; CAS: {}\n'.format(test_key,
54:                                                            ins_result.cas))
55:
56:         #KV - subdoc - mutate-in
57:         mti_result = collection.mutate_in(test_key,
58:                                           [SD.upsert(sd_path, sd_update)])
59:         print('Mutated sub-doc w/ key: {} and path: {}; CAS: {}\n'.format(test_key,
```

```
                                                    sd_path,
                                                    mti_result.cas))

        #KV - subdoc - lookup-in
        lki_result = collection.lookup_in(test_key,
                                          [SD.get(sd_path), SD.exists(sd_path)])
        print('Lookup in result, key: {}, path: {}, value: {}\n'.format(test_key,
                                                    sd_path,
                                                    lki_result.content_as[str](0)))

        #KV - remove
        rm_result = collection.remove(test_key)
        print('Removed doc w/ key {}; CAS: {}\n'.format(test_key,
                                                    rm_result.cas))

    except Exception as ex:
        import traceback
        traceback.print_exc()
```

ファイル：https://github.com/YoshiyukiKono/cb-dev-days-capella/blob/main/python/cbc_workshop_kv.py

プログラムを実行すると、次の内容が出力されます。

```
$ python cbc_workshop.py

Inserted doc w/ key: testDoc::0; CAS: 1613764919809736704

{'address': {'home': {'address1': '8575 Lewis Springs Mountains',
                      'city': 'West Mohammed',
                      'country': 'CA',
                      'state': 'WI',
                      'zipCode': '40243-4741'}},
 'info': 'This is a test',
 'type': 'testDoc'}

Upserted doc w/ key: testDoc::0; CAS: 1613764919812227072

Mutated sub-doc w/ key: testDoc::0 and path: address.home.address1; CAS:
1613764919813210112

Lookup in result, key: testDoc::0, path: address.home.address1, value: 8575
Lewis Springs Mountains Blvd

Removed doc w/ key testDoc::0; CAS: 1613764919815241728
```

ステップ 3:N1QL 操作

N1QL ロジックを追加します。 更新されたプログラムは以下のようになります。 クエリ操作の詳細については、ドキュメントを参照してください。

```
from pprint import pprint

from couchbase.cluster import Cluster, ClusterOptions, QueryOptions
```

```
3:  from couchbase.auth import PasswordAuthenticator
4:  import couchbase.subdocument as SD
5:
6:
7:  if __name__ == '__main__':
8:     try:
9:         # Cluster specific information
10:        cloud_endpoint = \
11:                    'cb.jzx-fxhtrfnpeej.cloud.couchbase.com'
12:        username = 'sherlock.holmes'
13:        password = 'P@ssw0rd'
14:        bucket_name = 'travel-sample'
15:
16:        # connect to cluster
17:        conn_str = 'couchbases://{}?ssl=no_verify'.format(cloud_endpoint)
18:        cluster_opts = ClusterOptions(PasswordAuthenticator(username,
19:                                            password))
20:        cluster = Cluster(conn_str, cluster_opts)
21:        bucket = cluster.bucket(bucket_name)
22:        collection = bucket.default_collection()
23:
24:        #KV operation data
25:        test_key = 'testDoc::0'
26:        test_doc =  {
27:            'type': 'testDoc',
28:            'info': 'This is a test',
29:            'address': {
30:                'home': {
31:                    'address1': '8575 Lewis Springs Mountains',
32:                    'city': 'West Mohammed',
33:                    'state': 'WI',
34:                    'zipCode': '40243-4741',
35:                    'country': 'CA'
36:                    }
37:                }
38:            }
39:        sd_path = 'address.home.address1'
40:        sd_update = '8575 Lewis Springs Mountains Blvd'
41:
42:        #KV - insert
43:        ins_result = collection.insert(test_key, test_doc)
44:        print('\nInserted doc w/ key: {}; CAS: {}\n'.format(test_key,
45:                                                           ins_result.cas))
46:
47:
48:        #KV - get
49:        get_result = collection.get(test_key)
50:        pprint(get_result.content)
51:
52:        #KV - upsert
53:        ins_result = collection.upsert(test_key, test_doc)
54:        print('\nUpserted doc w/ key: {}; CAS: {}\n'.format(test_key,
55:                                                           ins_result.cas))
56:
57:        #KV - subdoc - mutate-in
58:        mti_result = collection.mutate_in(test_key,
59:                                          [SD.upsert(sd_path, sd_update)])
60:        print('Mutated sub-doc w/ key: {} and path: {}; CAS: {}\n'.format(test_key,
61:                                                                  sd_path,
62:                                                                  mti_result.cas))
63:
64:        #KV - subdoc - lookup-in
        lki_result = collection.lookup_in(test_key,
```

```
                                    [SD.get(sd_path), SD.exists(sd_path)])
        print('Lookup in result, key: {}, path: {}, value: {}\n'.format(test_key,
                                                    sd_path,
                                                    lki_result.content_as[str](0)))

        #KV - remove
        rm_result = collection.remove(test_key)
        print('Removed doc w/ key {}; CAS: {}\n'.format(test_key,
                                                    rm_result.cas))

        #basic query w/ GROUP BY
        query_str = '''
        SELECT
            t.type AS DocType,
            COUNT(1) AS DocCount
        FROM `{}` t
        GROUP BY t.type
        '''.format(bucket_name)
        q_result = cluster.query(query_str)
        for r in q_result.rows():
            pprint(r)
        print()

        #sub-query example
        query_str = '''
        SELECT
            name,
            country,
            (SELECT raw avg(s.ratings.Overall)
             FROM t.reviews  as s)[0] AS overall_avg_rating
        FROM `{}` AS t
        WHERE
            t.type = "hotel"
        ORDER BY
            overall_avg_rating DESC
        LIMIT 10;
        '''.format(bucket_name)
        q_result = cluster.query(query_str)
        for r in q_result.rows():
            pprint(r)
        print()

        #positional query params
        query_str = '''
        SELECT COUNT(1) AS DocCount
        FROM `{}` t
        WHERE t.type=$1
        '''.format(bucket_name)
        query_opts = QueryOptions(positional_parameters=['hotel'])
        q_result = cluster.query(query_str, query_opts)
        for r in q_result.rows():
            pprint(r)
        print()

        #named query params
        query_str = '''
        SELECT COUNT(1) AS DocCount
        FROM `{}` t
        WHERE t.type=$doc_type
        '''.format(bucket_name)
        query_opts = QueryOptions(named_parameters={'doc_type':'airline'})
        q_result = cluster.query(query_str, query_opts)
        for r in q_result.rows():
```

```
127:                 pprint(r)
128:             print()
129:
130:     except Exception as ex:
131:         import traceback
132:         traceback.print_exc()
133:
134:
135:
```

ファイル：https://github.com/YoshiyukiKono/cb-dev-days-capella/blob/main/python/cbc_workshop_n1ql.py

プログラムを実行すると、次の内容が出力されます。

```
$ python cbc_workshop.py

Inserted doc w/ key: testDoc::0; CAS: 1613777143116070912

{'address': {'home': {'address1': '8575 Lewis Springs Mountains',
                      'city': 'West Mohammed',
                      'country': 'CA',
                      'state': 'WI',
                      'zipCode': '40243-4741'}},
 'info': 'This is a test',
 'type': 'testDoc'}

Upserted doc w/ key: testDoc::0; CAS: 1613777143118495744

Mutated sub-doc w/ key: testDoc::0 and path: address.home.address1; CAS:
1613777143119544320

Lookup in result, key: testDoc::0, path: address.home.address1, value: 8575
Lewis Springs Mountains Blvd

Removed doc w/ key testDoc::0; CAS: 1613777143121772544

{'DocCount': 24024, 'DocType': 'route'}
{'DocCount': 187, 'DocType': 'airline'}
{'DocCount': 1968, 'DocType': 'airport'}
{'DocCount': 917, 'DocType': 'hotel'}
{'DocCount': 4495, 'DocType': 'landmark'}

{'country': 'United Kingdom',
 'name': 'Park Plaza County Hall',
 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': 'La Reserve Hotel Chelsea',
 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': 'Marriott London County Hall',
 'overall_avg_rating': 5}
{'country': 'United Kingdom', 'name': 'Abbey Hotel', 'overall_avg_rating': 5}
{'country': 'France', 'name': 'Avignon Hotel Monclar', 'overall_avg_rating': 5}
{'country': 'France', 'name': 'La Pradella', 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': 'Culloden House Hotel',
 'overall_avg_rating': 5}
{'country': 'France',
```

```
 'name': 'Auberge-Camping Bagatelle',
 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': '8 Clarendon Crescent',
 'overall_avg_rating': 5}
{'country': 'United Kingdom', 'name': 'The Bulls Head', 'overall_avg_rating': 5}

{'DocCount': 917}

{'DocCount': 187}
```

ステップ 4: FTS 操作

フルテキスト検索 (FTS) ロジックを追加します。更新後のプログラムは以下のようになります。
全文検索操作の詳細については、SDK のドキュメントを参照してください。

```
from pprint import pprint

from couchbase.cluster import Cluster, ClusterOptions, QueryOptions
from couchbase.auth import PasswordAuthenticator
import couchbase.subdocument as SD
from couchbase.search import TermQuery, SearchOptions

if __name__ == '__main__':
    try:
        # Cluster specific information
        cloud_endpoint = \
                    'cb.jzx-fxhtrfnpeej.cloud.couchbase.com'
        username = 'sherlock.holmes'
        password = 'P@ssw0rd'
        bucket_name = 'travel-sample'

        # connect to cluster
        conn_str = 'couchbases://{}?ssl=no_verify'.format(cloud_endpoint)
        cluster_opts = ClusterOptions(PasswordAuthenticator(username,
                                      password))
        cluster = Cluster(conn_str, cluster_opts)
        bucket = cluster.bucket(bucket_name)
        collection = bucket.default_collection()

        #KV operation data
        test_key = 'testDoc::0'
        test_doc =  {
            'type': 'testDoc',
            'info': 'This is a test',
            'address': {
                'home': {
                    'address1': '8575 Lewis Springs Mountains',
                    'city': 'West Mohammed',
                    'state': 'WI',
                    'zipCode': '40243-4741',
                    'country': 'CA'
                    }
                }
            }
        sd_path = 'address.home.address1'
        sd_update = '8575 Lewis Springs Mountains Blvd'
```

```
#KV - insert
ins_result = collection.insert(test_key, test_doc)
print('\nInserted doc w/ key: {}; CAS: {}\n'.format(test_key,
                                                   ins_result.cas))

#KV - get
get_result = collection.get(test_key)
pprint(get_result.content)

#KV - upsert
ins_result = collection.upsert(test_key, test_doc)
print('\nUpserted doc w/ key: {}; CAS: {}\n'.format(test_key,
                                                   ins_result.cas))

#KV - subdoc - mutate-in
mti_result = collection.mutate_in(test_key,
                                  [SD.upsert(sd_path, sd_update)])
print('Mutated sub-doc w/ key: {} and path: {}; CAS: {}\n'.format(test_key,
                                                  sd_path,
                                                  mti_result.cas))

#KV - subdoc - lookup-in
lki_result = collection.lookup_in(test_key,
                                  [SD.get(sd_path), SD.exists(sd_path)])
print('Lookup in result, key: {}, path: {}, value: {}\n'.format(test_key,
                                                  sd_path,
                                                  lki_result.content_as[str](0)))

#KV - remove
rm_result = collection.remove(test_key)
print('Removed doc w/ key {}; CAS: {}\n'.format(test_key,
                                                  rm_result.cas))

#basic query w/ GROUP BY
query_str = '''
SELECT
    t.type AS DocType,
    COUNT(1) AS DocCount
FROM `{}` t
GROUP BY t.type
'''.format(bucket_name)
q_result = cluster.query(query_str)
for r in q_result.rows():
    pprint(r)
print()

#sub-query example
query_str = '''
SELECT
    name,
    country,
    (SELECT raw avg(s.ratings.Overall)
     FROM t.reviews  as s)[0] AS overall_avg_rating
FROM `{}` AS t
WHERE
    t.type = "hotel"
ORDER BY
    overall_avg_rating DESC
LIMIT 10;
'''.format(bucket_name)
q_result = cluster.query(query_str)
for r in q_result.rows():
    pprint(r)
```

```
        print()

        #positional query params
        query_str = '''
        SELECT COUNT(1) AS DocCount
        FROM `{}` t
        WHERE t.type=$1
        '''.format(bucket_name)
        query_opts = QueryOptions(positional_parameters=['hotel'])
        q_result = cluster.query(query_str, query_opts)
        for r in q_result.rows():
            pprint(r)
        print()

        #named query params
        query_str = '''
        SELECT COUNT(1) AS DocCount
        FROM `{}` t
        WHERE t.type=$doc_type
        '''.format(bucket_name)
        query_opts = QueryOptions(named_parameters={'doc_type':'airline'})
        q_result = cluster.query(query_str, query_opts)
        for r in q_result.rows():
            pprint(r)
        print()

        #FTS TermQuery data
        search_term = 'london'
        fuzzy = 1

        #FTS operations
        result = cluster.search_query('default',
                    TermQuery(term=search_term, fuzziness=fuzzy),
                    SearchOptions(limit=100))
        doc_ids = list(map(lambda r: r.id, result))
        docs_found = result.metadata().metrics.total_rows
        print(
            '\nFound {} docs that matched search for {} with fuzziness of {}\n'
            .format(docs_found, search_term, fuzzy))
        if len(doc_ids) > 0:
            sample_doc = collection.get(doc_ids[0])
            pprint(sample_doc.content)

    except Exception as ex:
        import traceback
        traceback.print_exc()
```

ファイル：https://github.com/YoshiyukiKono/cb-dev-days-capella/blob/main/python/cbc_workshop_fts.py

プログラムを実行すると、次の内容が出力されます。

```
$ python cbc_workshop.py

Inserted doc w/ key: testDoc::0; CAS: 1613777143116070912

{'address': {'home': {'address1': '8575 Lewis Springs Mountains',
                      'city': 'West Mohammed',
                      'country': 'CA',
                      'state': 'WI',
                      'zipCode': '40243-4741'}},
 'info': 'This is a test',
 'type': 'testDoc'}
```

```
Upserted doc w/ key: testDoc::0; CAS: 1613777143118495744

Mutated sub-doc w/ key: testDoc::0 and path: address.home.address1; CAS:
1613777143119544320

Lookup in result, key: testDoc::0, path: address.home.address1, value: 8575
Lewis Springs Mountains Blvd

Removed doc w/ key testDoc::0; CAS: 1613777143121772544

{'DocCount': 24024, 'DocType': 'route'}
{'DocCount': 187, 'DocType': 'airline'}
{'DocCount': 1968, 'DocType': 'airport'}
{'DocCount': 917, 'DocType': 'hotel'}
{'DocCount': 4495, 'DocType': 'landmark'}

{'country': 'United Kingdom',
 'name': 'Park Plaza County Hall',
 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': 'La Reserve Hotel Chelsea',
 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': 'Marriott London County Hall',
 'overall_avg_rating': 5}
{'country': 'United Kingdom', 'name': 'Abbey Hotel', 'overall_avg_rating': 5}
{'country': 'France', 'name': 'Avignon Hotel Monclar', 'overall_avg_rating': 5}
{'country': 'France', 'name': 'La Pradella', 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': 'Culloden House Hotel',
 'overall_avg_rating': 5}
{'country': 'France',
 'name': 'Auberge-Camping Bagatelle',
 'overall_avg_rating': 5}
{'country': 'United Kingdom',
 'name': '8 Clarendon Crescent',
 'overall_avg_rating': 5}
{'country': 'United Kingdom', 'name': 'The Bulls Head', 'overall_avg_rating': 5}

{'DocCount': 917}

{'DocCount': 187}

Found 809 docs that matched search for london with fuzziness of 1

{'activity': 'see',
 'address': None,
 'alt': None,
 'city': 'London',
 'content': 'Remains of the wall that surrounded the City of London for almost '
            'two thousand years. The parts around the Barbican are mostly '
            'Tudor due to maintenance (Roman remains can be seen in and around '
            'the Tower of London). Other local remains are the',
 'country': 'United Kingdom',
 'directions': 'Near the street called &amp;quot;London Wall&amp;quot;',
 'email': None,
```

```
'geo': {'accuracy': 'RANGE_INTERPOLATED', 'lat': 51.5177, 'lon': -0.0952},
'hours': None,
'id': 37138,
'image': 'https://en.wikivoyage.org/wiki/File:London Wall bastion, Barber '
          "Surgeons' Hall Gardens, London EC2.jpg",
'image_direct_url': '',
'name': 'London Wall',
'phone': None,
'price': None,
'state': None,
'title': 'Wikimania 2014 London Guidebook',
'tollfree': None,
'type': 'landmark',
'url': None}
```