Couchbase

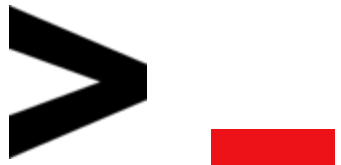
3교시.
SDK 소개 (Python) 및 실습

1 SDK 소개 (Python)

2 SDK 실습 (Python)

A 데이터 저장 방식 및 고가용성

3-1. SDK 소개 (Python)



SDK 개요

10 개 개발 언어에 대한 Couchbase 클러스터 접근을 위한 SDK(Software Development Kit) 제공

Enterprise Support

v4.2

V4.4

v3.6

v2.9

v3.7

v4.3

v3.3

v1.7

v3.5

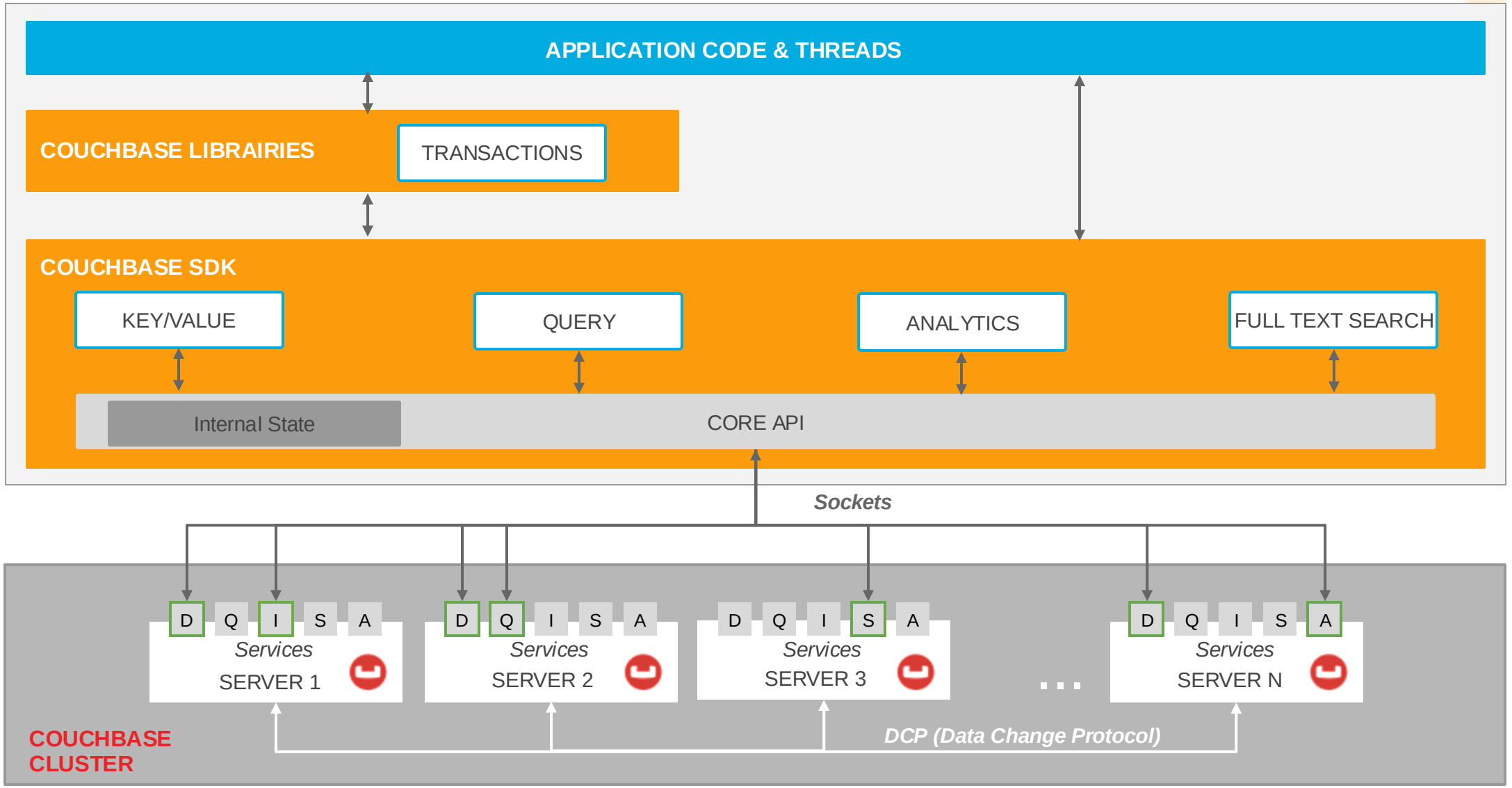
Mobile Support

Java for Android

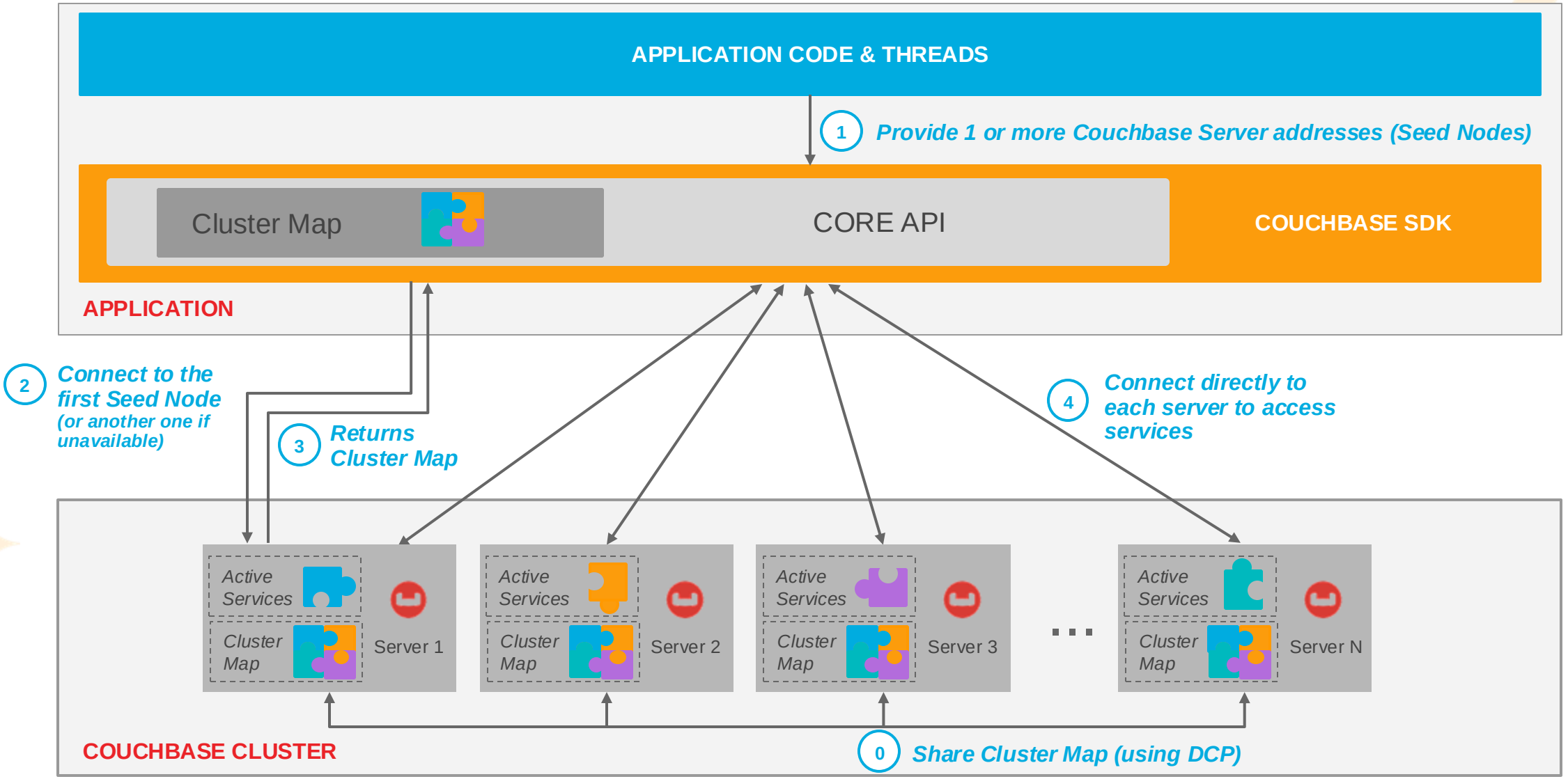
Community Support

v1.0-pre

SDK 아키텍처 : 통합 API 제공



SDK 아키텍처 : Connection 세부 단계(Bootstrapping)



Connection Management | Connect to Cluster

Python provides a well thought out Object Oriented Interface

필요한 Library 등록

접속 정보 정의

인증 정보

카우치베이스 접속 및 대기시간 정의

사용할 scope>collection 지정

```
from datetime import timedelta
# needed for any cluster connection from couchbase.auth
import PasswordAuthenticator from couchbase.cluster
import Cluster
# needed for options -- cluster, timeout, SQL++ (N1QL) query, etc.
from couchbase.options import (ClusterOptions, ClusterTimeoutOptions, QueryOptions)

# Update this to your cluster
endpoint = "couchbase://your-ip"
username = "Administrator"
password = "password"
bucket_name = "travel-sample"

# User Input ends here.
# Connect options - authentication
auth = PasswordAuthenticator( username, password )

# Get a reference to our cluster
# NOTE: For TLS/SSL connection use 'couchbases://<your-ip-address>' instead
cluster = Cluster(endpoint, ClusterOptions(auth))

# Wait until the cluster is ready for use.
cluster.wait_until_ready(timedelta(seconds=5))

# get a reference to our bucket
cb = cluster.bucket(bucket_name)
cb_coll = cb.scope("inventory").collection("airline")
```

Key Value Operations | Get/Upsert/Insert

Python allows reading JSON docs directly

Key로 문서를 가져오는 Get 함수 정의

Get 함수 호출

Json 문서 정의

Key, Value(Json문서)를 저장하는 Upsert 함수 정의

Upsert 함수 호출

```
# get document function
def get_airline_by_key(key):
print("\nGet Result: ")
try:
   result = cb_coll.get(key)
   print(result.content_as[str])
except Exception as e:
   print(e)

get_airline_by_key("airline_8091")

airline = {
   "type": "airline",
   "id": 8091,
   "callsign": "CBS",
   "iata": None,
   "icao": None,
   "name": "Couchbase Airways"
}

def upsert_document(doc):
   print("\nUpsert CAS: ")
   try:
      # key will equal: "airline_8091"
      key = doc["type"] + "_" + str(doc["id"])
      result = cb_coll.upsert(key, doc)
      print(result.cas)
   except Exception as e:
      print(e)

upsert_document(airline)
```

Type과 id로 Key 생성

Upsert가 아니라 insert로 해도 됨
result = cb_coll.insert(key, doc)

Key Value Operations | SQL++

SQL 호출시, 파라메터 사용

입력 받은 값으로 SQL 실행하여 결과를 출력하는 함수 정의

lookup 함수 호출

```
# query for new document by callsign

def lookup_by_callsign(cs):
    print("\nLookup Result: ")
    try:
        inventory_scope = cb.scope('inventory')
        sql_query = 'SELECT VALUE name FROM airline WHERE callsign = $1'
        row_iter = inventory_scope.query(
            sql_query,
            QueryOptions(positional_parameters=[cs]))
        for row in row_iter:
            print(row)
    except Exception as e:
        print(e)

lookup_by_callsign("CBS")
```

위치 정의 파라메터

Key Value Operations | SQL++ , Metrics 확인

SQL 호출시, 수행한 시간, 결과 건수 등의 메트릭 정보 확인

SQL 실행하여 결과를 출력시, 건수와 실행 시간 확인

```
from couchbase.cluster import Cluster
from couchbase.options import ClusterOptions, QueryOptions
from couchbase.auth import PasswordAuthenticator
from couchbase.exceptions import CouchbaseException

cluster = Cluster.connect(
    "couchbase://your-ip",
    ClusterOptions(PasswordAuthenticator("Administrator", "password")))
bucket = cluster.bucket("travel-sample")
collection = bucket.default_collection()

try:
    result = cluster.query(
        "SELECT * FROM `travel-sample`.inventory.airport LIMIT 10", QueryOptions(metrics=True))

    for row in result.rows():
        print(f"Found row: {row}")

    print(f"Report execution time: {result.metadata().metrics().execution_time()}")

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

다른 예시
QueryOptions(read_only=True)

Key Value Operations | SQL++ , 결과 출력 방법

SQL 호출시, 결과를 원하는 방식으로 출력하는 방법

SQL 실행

결과 출력

```
from couchbase.cluster import Cluster
from couchbase.options import ClusterOptions, QueryOptions
from couchbase.auth import PasswordAuthenticator
from couchbase.exceptions import CouchbaseException

cluster = Cluster.connect(
    "couchbase://your-ip",
    ClusterOptions(PasswordAuthenticator("Administrator", "password")))
bucket = cluster.bucket("travel-sample")
collection = bucket.default_collection()

result = cluster.query(
    "SELECT * FROM `travel-sample`.inventory.airline LIMIT 10")

# iterate over rows
for row in result:
    # each row is an instance of the query call
    try:
        name = row["airline"]["name"]
        callsign = row["airline"]["callsign"]
        print(f"Airline name: {name}, callsign: {callsign}")
    except KeyError:
        print("Row does not contain 'name' key")
```

Key Value Operations | SQL++ , Sub-Document 처리

Sub Documents에 대한 검색, 수정, 추가 방법

key : customer123

https://docs.couchbase.com/python-sdk/current/howtos/subdocument-operations.html

```
{
  "name": "Douglas Reynholm",
  "email": "douglas@reynholmindustries.com",
  "addresses": {
    "billing": {
      "line1": "123 Any Street",
      "line2": "Anytown",
      "country": "United Kingdom"
    },
    "delivery": {
      "line1": "123 Any Street",
      "line2": "Anytown",
      "country": "United Kingdom"
    }
  },
  "purchases": {
    "complete": [
      339, 976, 442, 666
    ],
    "abandoned": [
      157, 42, 999
    ]
  }
}
```

```
result = collection.lookup_in('customer123',
                              [SD.get('addresses.delivery.country')])
country = result.content_as[str](0)  # 'United Kingdom'


result = collection.lookup_in('customer123', [SD.exists('purchases.pending[-1]')])
print(f'Path exists: {result.exists(0)}.')
# Path exists:  False.

result = collection.lookup_in('customer123',[SD.get('addresses.delivery.country'),
                                             SD.exists('purchases.complete[-1]')])

print('{0}'.format(result.content_as[str](0)))
print('Path exists: {}.'.format(result.exists(1)))
# path exists: True.

collection.mutate_in('customer123', [SD.upsert('fax', '311-555-0151')])

collection.mutate_in('customer123', [SD.insert('purchases.pending', [42, True,
'None'])])

try:
    collection.mutate_in('customer123', [SD.insert('purchases.complete',[42, True,
'None'])])
except PathExistsException:
    print('Path exists, cannot use insert.')

collection.mutate_in('customer123',(SD.remove('addresses.billing'),
                                    SD.replace('email','dougr96@hotmail.com')))
```

Key Value Operations | Full Text Search

Full Text Search에도 다양한 검색을 수행할 수 있음.

https://docs.couchbase.com/python-sdk/current/howtos/full-text-searching-with-sdk.html

```
from couchbase.cluster import Cluster
from couchbase.options import ClusterOptions,
SearchOptions
from couchbase.auth import PasswordAuthenticator
from couchbase.exceptions import
CouchbaseException
import couchbase.search as search

auth = PasswordAuthenticator('Administrator',
'password')
cluster = Cluster.connect('couchbase://your-ip',
ClusterOptions(auth))
bucket = cluster.bucket('travel-sample')
scope = bucket.scope('inventory')
collection = scope.collection('hotel')
```

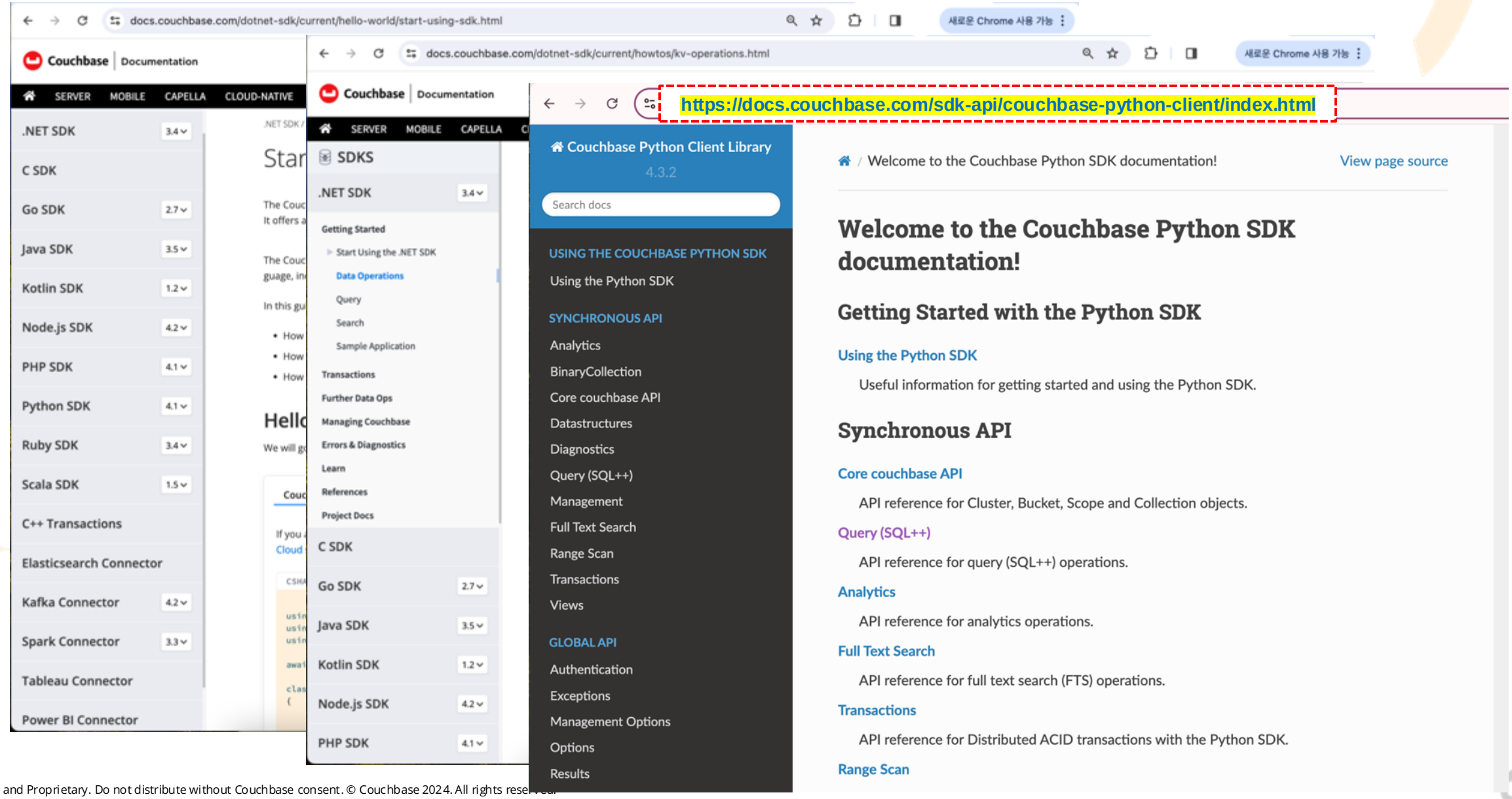
미리 만들어 둔 검색 인덱스 명

```
try:
    result = cluster.search_query('travel-sample-index',
                                  search.QueryStringQuery('Paris'))

    for row in result.rows():
        print(f'Found row: {row}')

    print(f'Reported total rows: {result.metadata().metrics().total_rows()}')

except CouchbaseException as ex:
    import traceback
    traceback.print_exc()

result = cluster.search_query('travel-sample-index',
                              search.PrefixQuery('swim'),
                              SearchOptions(fields=['description']))

for row in result.rows():
    print(f'Score: {row.score}')
    print(f'Document Id: {row.id}')

    # print fields included in query:
    print(row.fields)
```

Keyword 검색

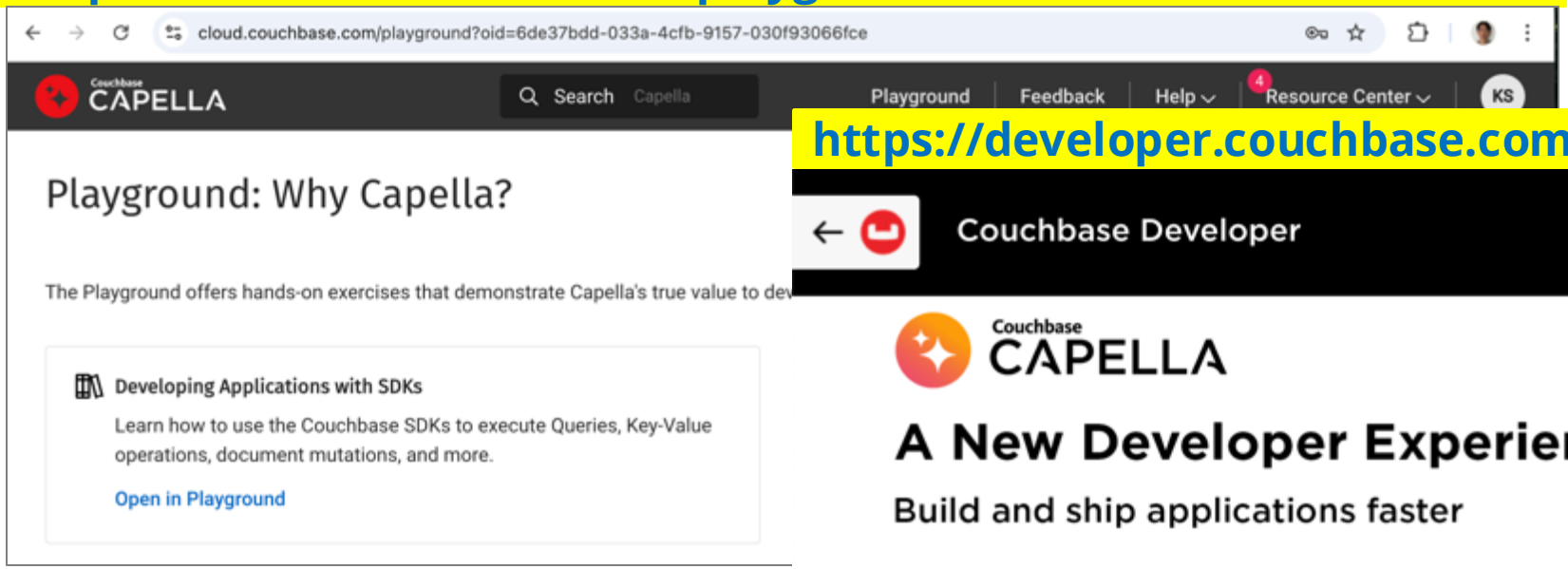
Prefix(접두어) 검색

개발 지원 문서

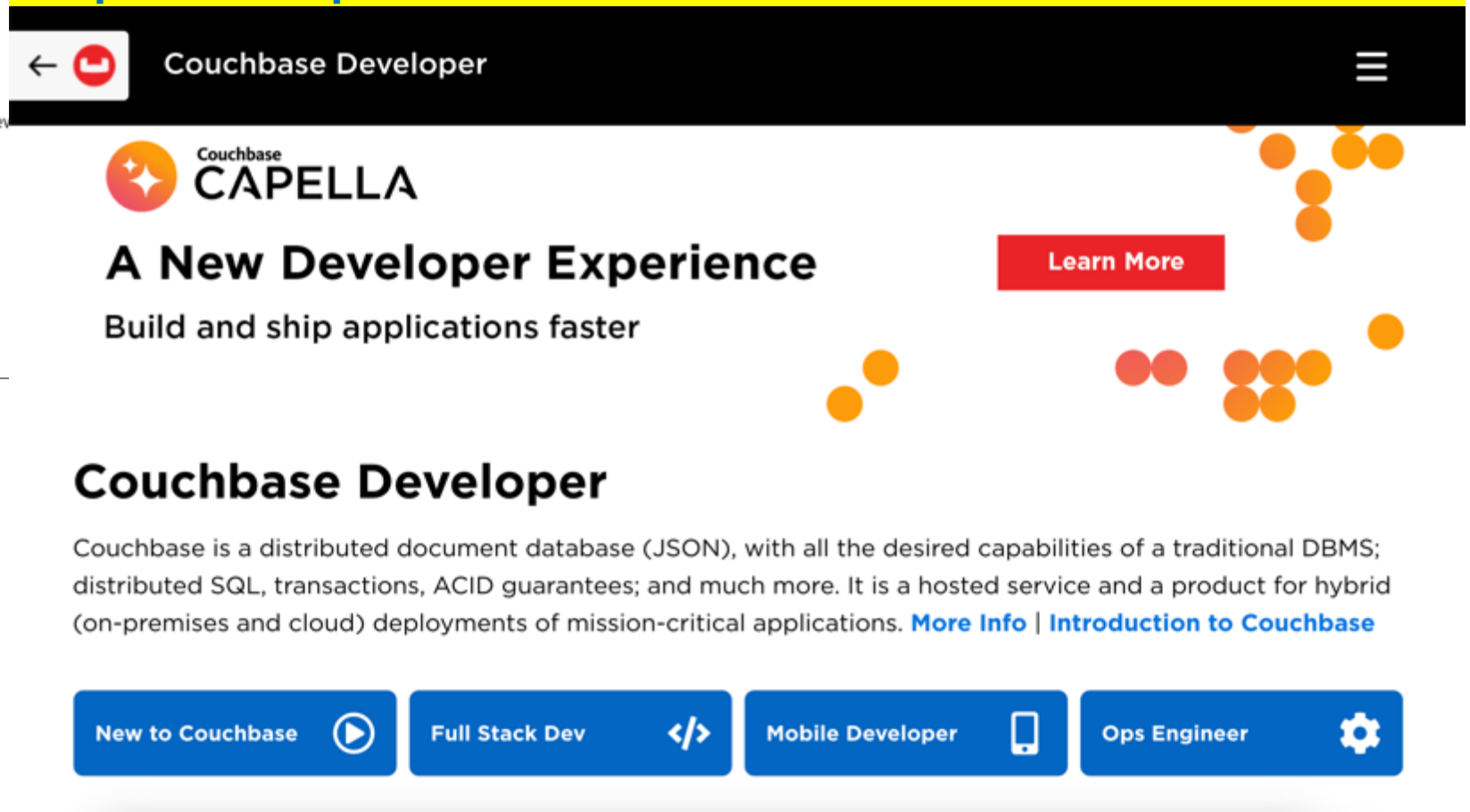


개발자 포털/플레이그라운드

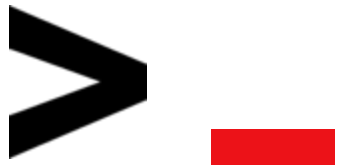
https://cloud.couchbase.com/playground



https://developer.couchbase.com/tutorials



3-2. SDK 실습 (Python, Java)



Windows 환경에서 Python 수행 환경 만들기

1. Python 설치

1. **Python 다운로드**: Python 공식 웹사이트에서 Windows용 설치 파일을 다운로드합니다.
2. **설치 실행**: 다운로드한 파일을 실행합니다.
   - **"Add Python to PATH"** 옵션을 체크한 후 **Install Now**를 클릭합니다. PATH에 추가하면 나중에 명령 프롬프트에서 Python을 쉽게 사용할 수 있습니다.
3. **설치 확인**: 설치가 완료되면 명령 프롬프트를 열고 다음 명령어로 설치가 정상적으로 되었는지 확인합니다.

```
python --version
```

2. 가상 환경 생성

가상 환경을 사용하면 프로젝트마다 Python 및 패키지 종속성을 독립적으로 관리할 수 있습니다.

1. **프로젝트 폴더 생성**: 먼저 작업할 폴더를 만듭니다.

```
mkdir my_project
cd my_project
```

2. **가상 환경 생성**: python -m venv 명령을 사용하여 가상 환경을 생성합니다.

```
python -m venv venv
```

여기서 venv는 가상 환경 폴더 이름으로, my_project 폴더 안에 venv 폴더가 생성됩니다.

3. **가상 환경 활성화**:
   - Windows에서는 다음 명령으로 가상 환경을 활성화합니다.

```
venv\Scripts\activate
```

   - 활성화되면 명령 프롬프트에 (venv)라는 표시가 나타납니다.

4. **가상 환경 비활성화**:
   - 작업이 끝나면 deactivate 명령어로 가상 환경을 비활성화할 수 있습니다.

```
deactivate
```

3. 패키지 설치

가상 환경이 활성화된 상태에서 pip를 사용하여 필요한 패키지를 설치할 수 있습니다.

```
pip install package_name
```

예를 들어, requests 라이브러리를 설치하려면 다음과 같이 입력합니다.

```
pip install requests
```

pip install couchbase

4. requirements.txt로 종속성 관리하기

프로젝트의 모든 종속성을 기록하려면 requirements.txt 파일을 생성합니다. 나중에 다른 환경에서 동일한 패키지를 설치할 수 있도록 다음과 같이 만듭니다.

1. **requirements.txt 파일 생성**:

```
pip freeze > requirements.txt
```

2. **requirements.txt로 패키지 설치**:

다른 환경에서 requirements.txt를 사용하여 패키지를 설치할 수 있습니다.

```
pip install -r requirements.txt
```

MacOS 환경에서 Python 수행 환경 만들기

1. Python 설치

MacOS에는 기본적으로 Python 2가 설치되어 있을 수 있지만, 최신 버전인 Python 3를 설치하는 것이 좋습니다.

1. **Python 설치**:

brew update

brew install openssl@1.1 python3

2. **설치 확인**: python3 –version

MacOS에 Homebrew 가 설치되어 있지 않는 경우

Homebrew는 MacOS용 패키지 관리자입니다. 터미널을 열고 다음 명령을 입력하여 Homebrew를 설치합니다.

/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

2. 가상 환경 생성

가상 환경을 사용하면 프로젝트마다 Python 및 패키지 종속성을 독립적으로 관리할 수 있습니다.

1. **프로젝트 폴더 생성**: 먼저 작업할 폴더를 만듭니다.

mkdir my_project

cd my_project

2. **가상 환경 생성**: python -m venv 명령을 사용하여 가상 환경을 생성합니다.

python3 -m venv **venv**

venv라는 이름의 폴더가 프로젝트 폴더 내에 생성되며, 이 폴더가 가상 환경을 담고 있습니다. 이름은 자유롭게 지정할 수 있습니다.

3. **가상 환경 활성화**:

source venv/bin/activate

활성화되면 터미널에 (venv) 표시가 나타나 가상 환경이 활성화된 것을 알 수 있습니다.

4. **가상 환경 비활성화**:

작업이 끝나면 deactivate 명령어로 가상 환경을 비활성화할 수 있습니다.

deactivate

3. 패키지 설치

가상 환경이 활성화된 상태에서 pip를 사용하여 필요한 패키지를 설치할 수 있습니다.

pip install package_name

예를 들어, requests 라이브러리를 설치하려면 다음과 같이 입력합니다.

pip install requests

pip install couchbase

4. requirements.txt로 종속성 관리하기

프로젝트의 모든 종속성을 기록하려면 requirements.txt 파일을 생성합니다. 나중에 다른 환경에서 동일한 패키지를 설치할 수 있도록 다음과 같이 만듭니다.

1. **requirements.txt 파일 생성**:

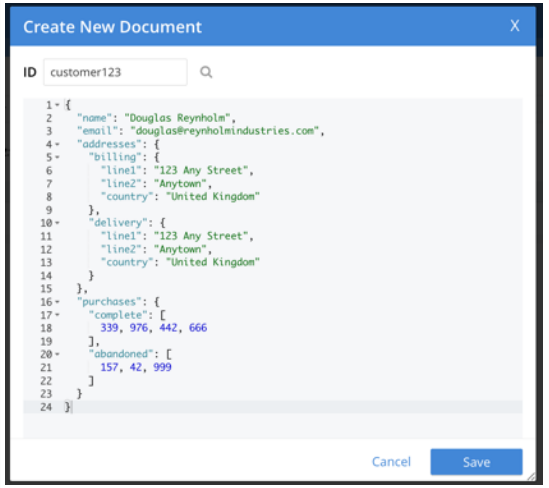
pip freeze > requirements.txt

2. **requirements.txt로 패키지 설치**:

다른 환경에서 requirements.txt를 사용하여 패키지를 설치할 수 있습니다.

pip install -r requirements.txt

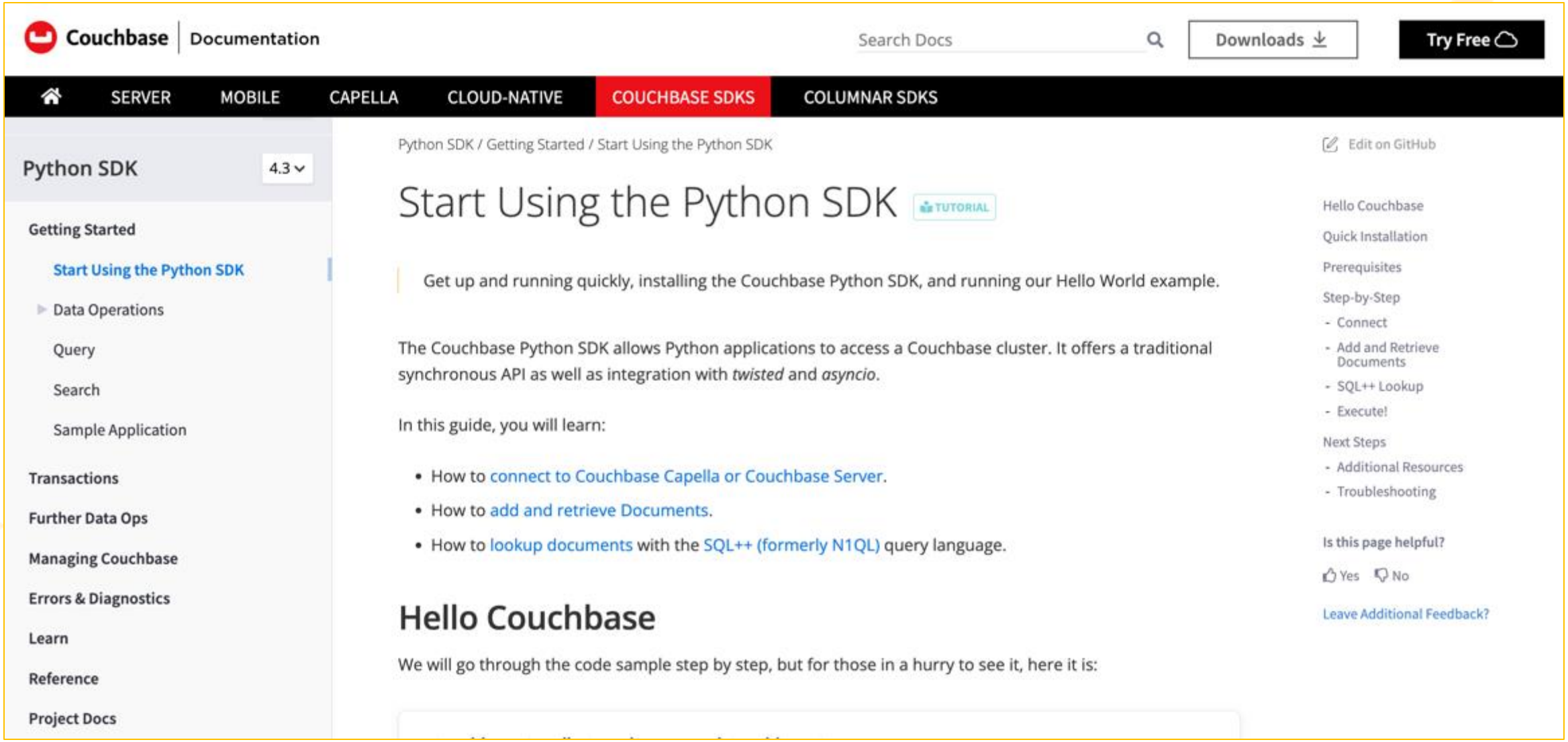
실습

	실습 항목	상세 실습 내용	기타
1	개발 환경 (구성) 확인	• Python 실행 환경 구성	pip install couchbase
2	기본 K/V, SQL++	• 아래에 있는 소스 실행해 보기 • https://docs.couchbase.com/python-sdk/current/hello-world/start-using-sdk.html • https://github.com/unixfree/CouchbaseTraining/blob/main/src/kv_sql_basic.py	
3	Sub-Document 처리	1. travel-sample bucket 의 inventory scope 에 subdoc collection 생성 2. Ducuments UI 에서 오른쪽 상단에 ADD DOCUMENT 클릭 후, ***11 페이지***에 있는 Json 문서 등록 key : `customer123` ***3. 11 페이지***에 있는 소스 완성하여 실행해 보기 https://github.com/unixfree/CouchbaseTraining/blob/main/src/subdoc.py	
4	Range Scan (KV, SQL)	• https://github.com/unixfree/CouchbaseTraining/blob/main/src/rangescan_sql.py • https://github.com/unixfree/CouchbaseTraining/blob/main/src/rangescan_kv.py	
5	Travel-sample Application	• https://docs.couchbase.com/python-sdk/current/hello-world/sample-application.html • https://github.com/couchbaselabs/try-cb-python • 한글 버전 : https://github.com/unixfree/try-cb-python	Couchbase, Flask, Vue.JS

실습

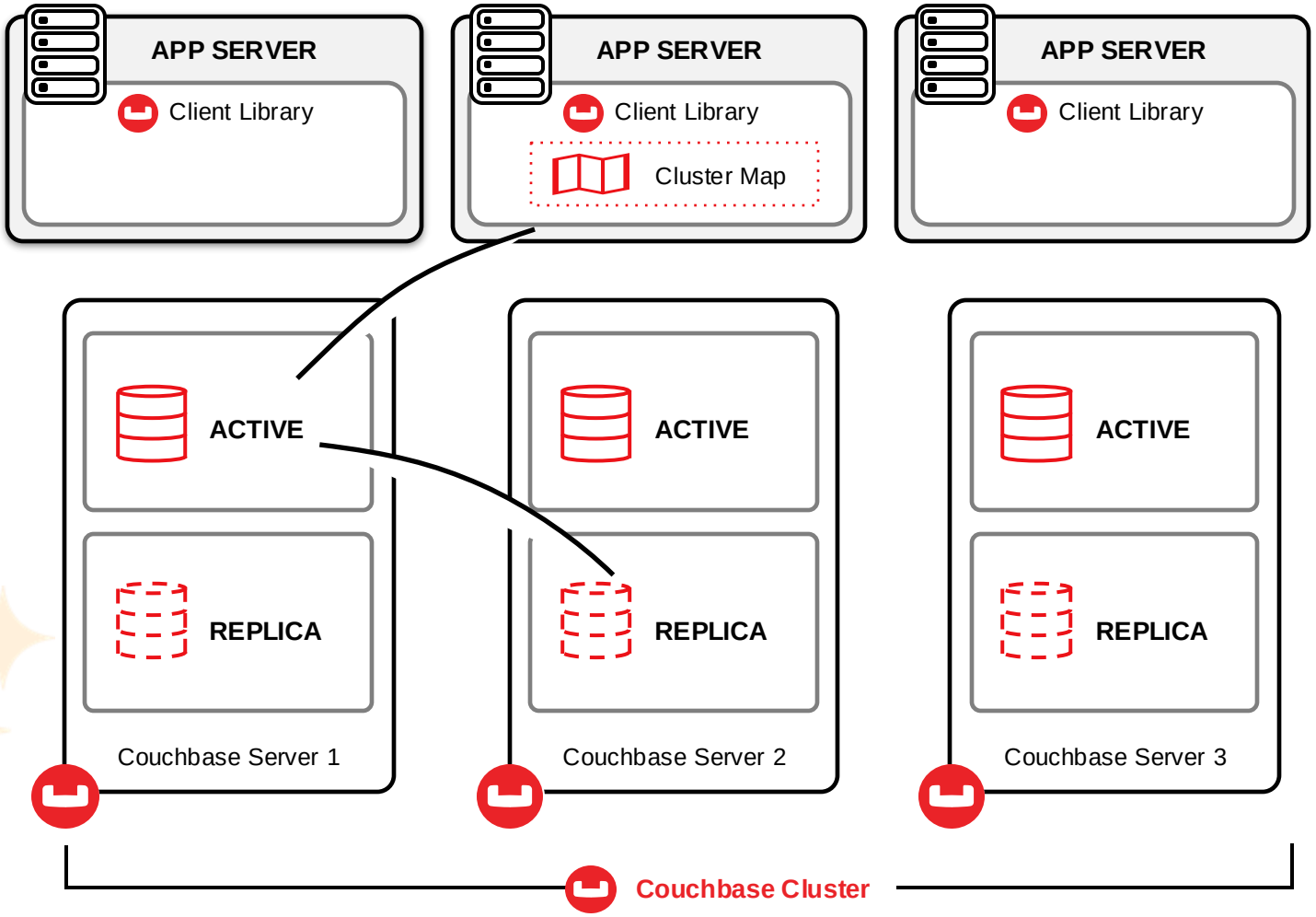
https://docs.couchbase.com/python-sdk/current/hello-world/start-using-sdk.html

KV Rangescan/Prefixscan
동일한 SQL Query



Appendix. 데이터 저장 방식 및 고가용성

A Fully Distributed Database

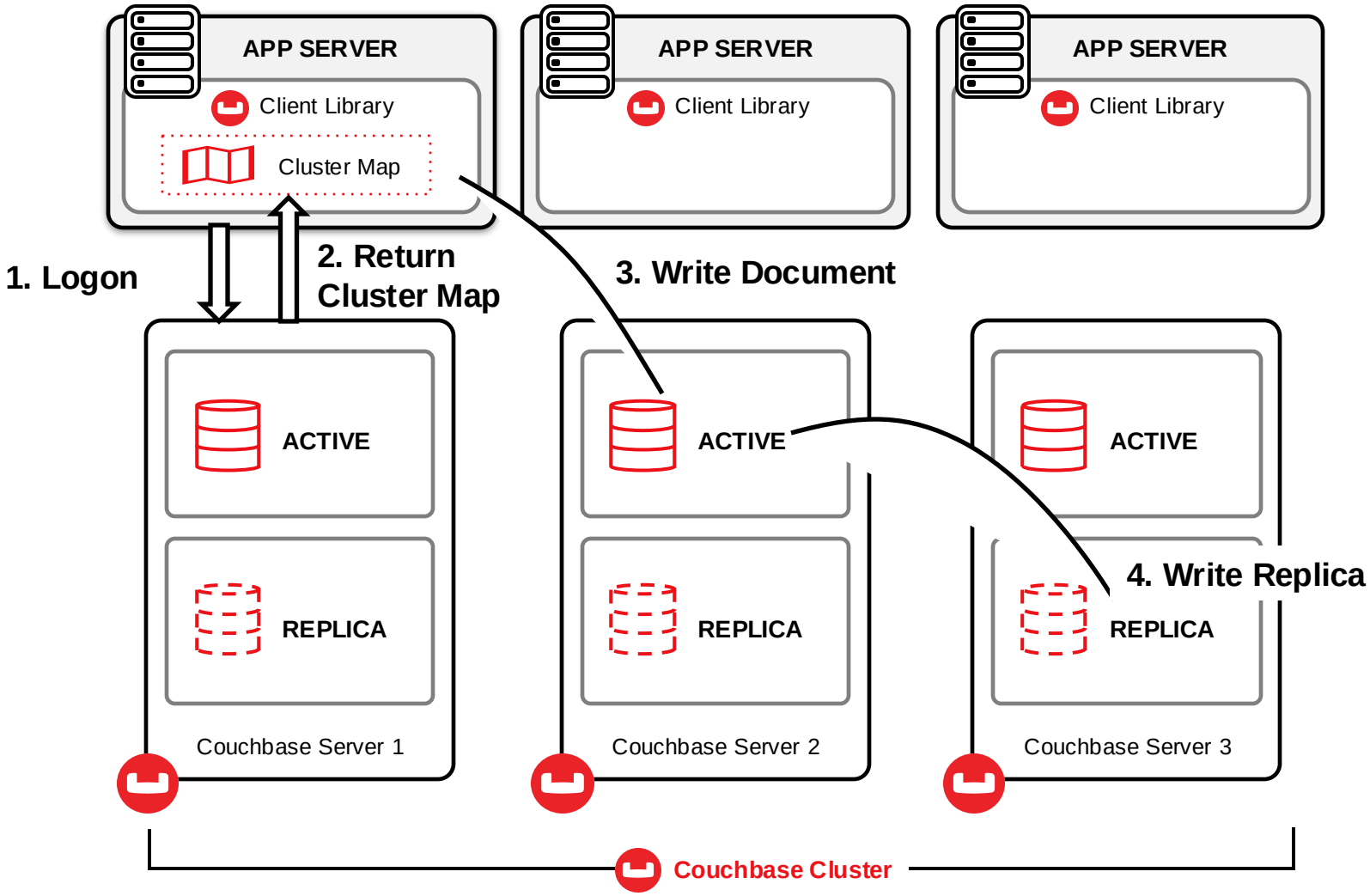


No Master-Slave
Single node type
Direct client communication

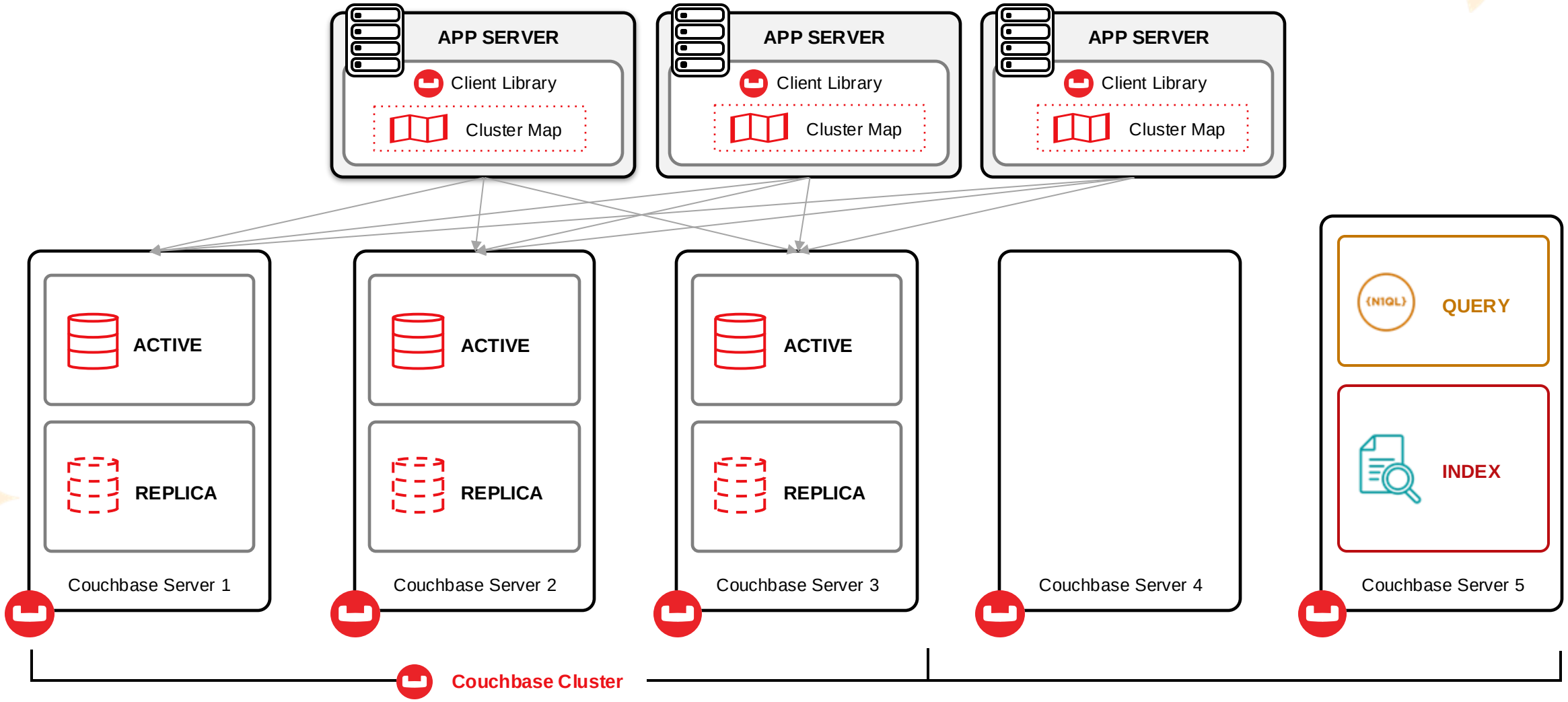
Automatic sharding
Maximum cluster efficiency

Automatic failover
Continuous replication
for no disruption

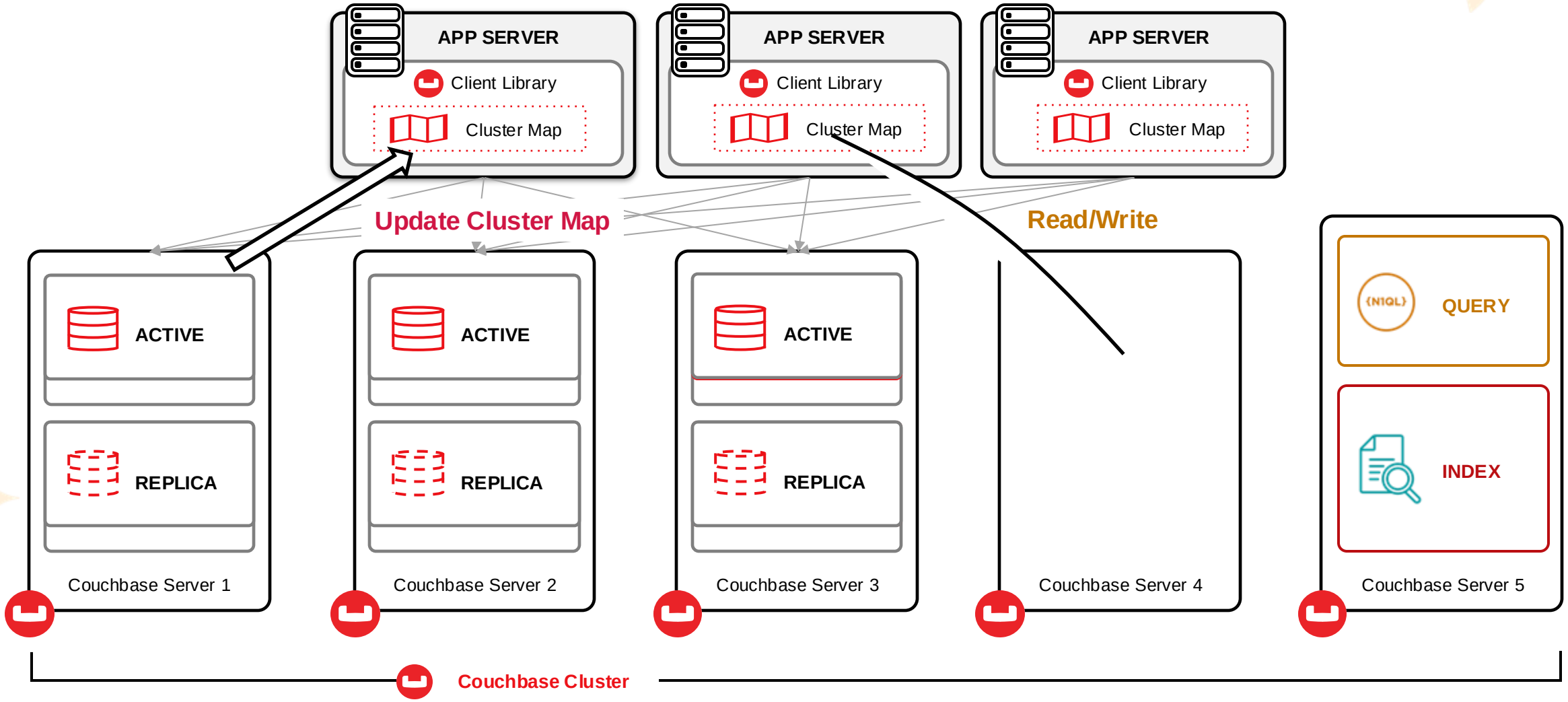
Basics Operations



Scalability – Horizontal & Vertical

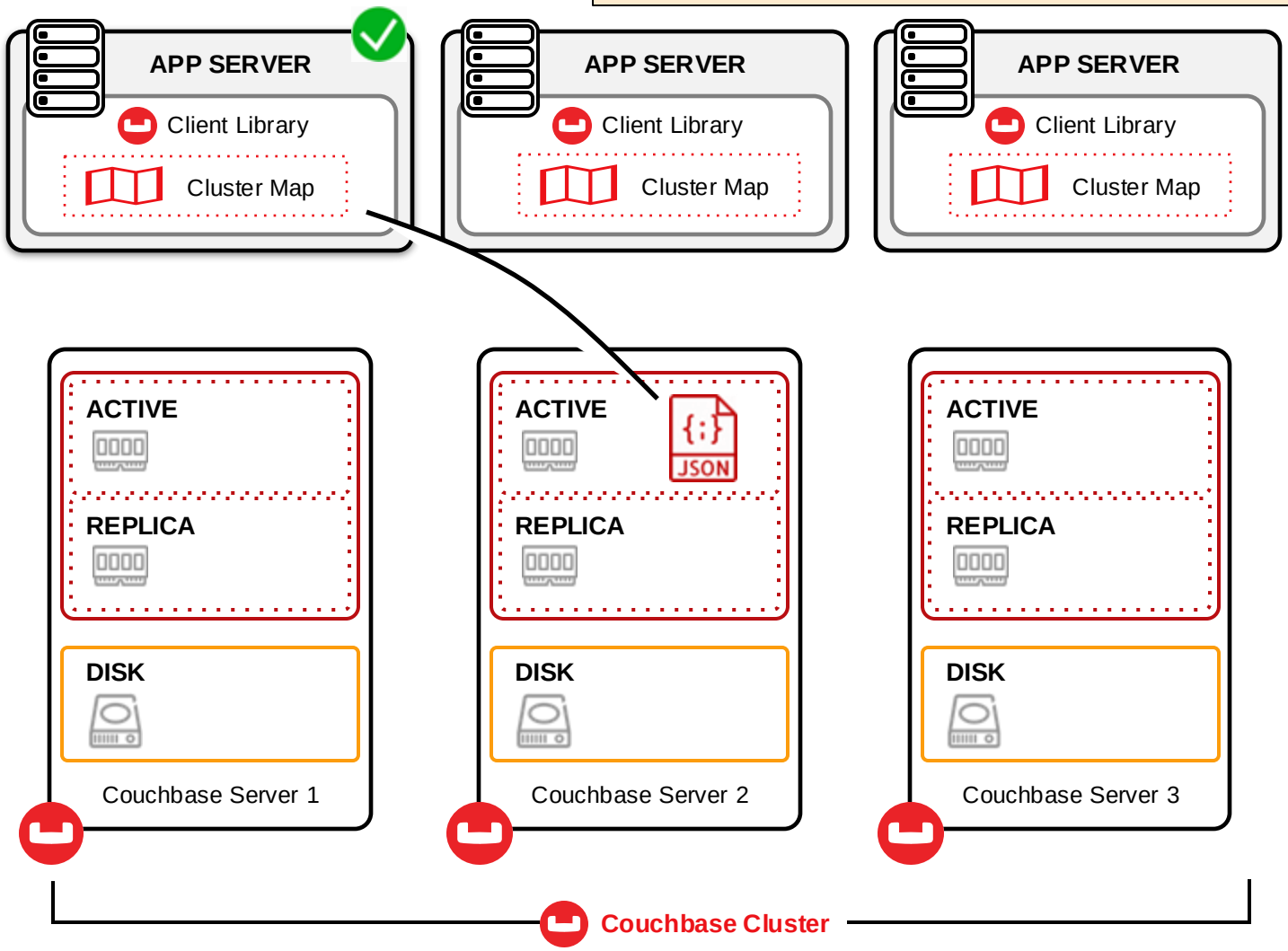


Scalability – Horizontal & Vertical



Durability

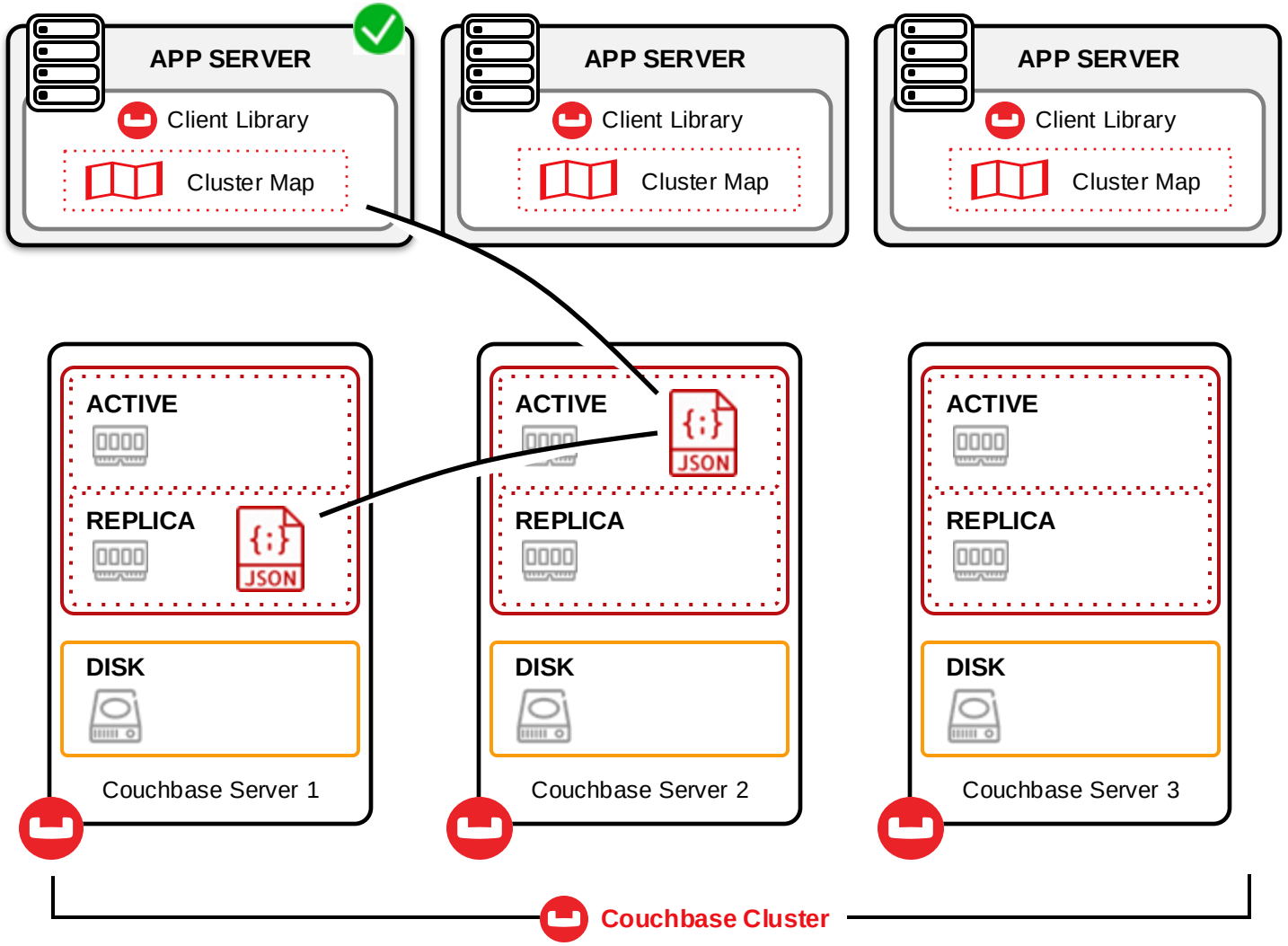
```
# Upsert with Durability (Couchbase Server >= 6.5) level Majority
   document = dict(foo="bar", bar="foo")
   opts = UpsertOptions(durability=ServerDurability(Durability.MAJORITY))
   result = collection.upsert("document-key", document, opts)
```



replica = 1

Level of Durability
None
Majority
MajorityAndPersistToActive
PersistToMajority

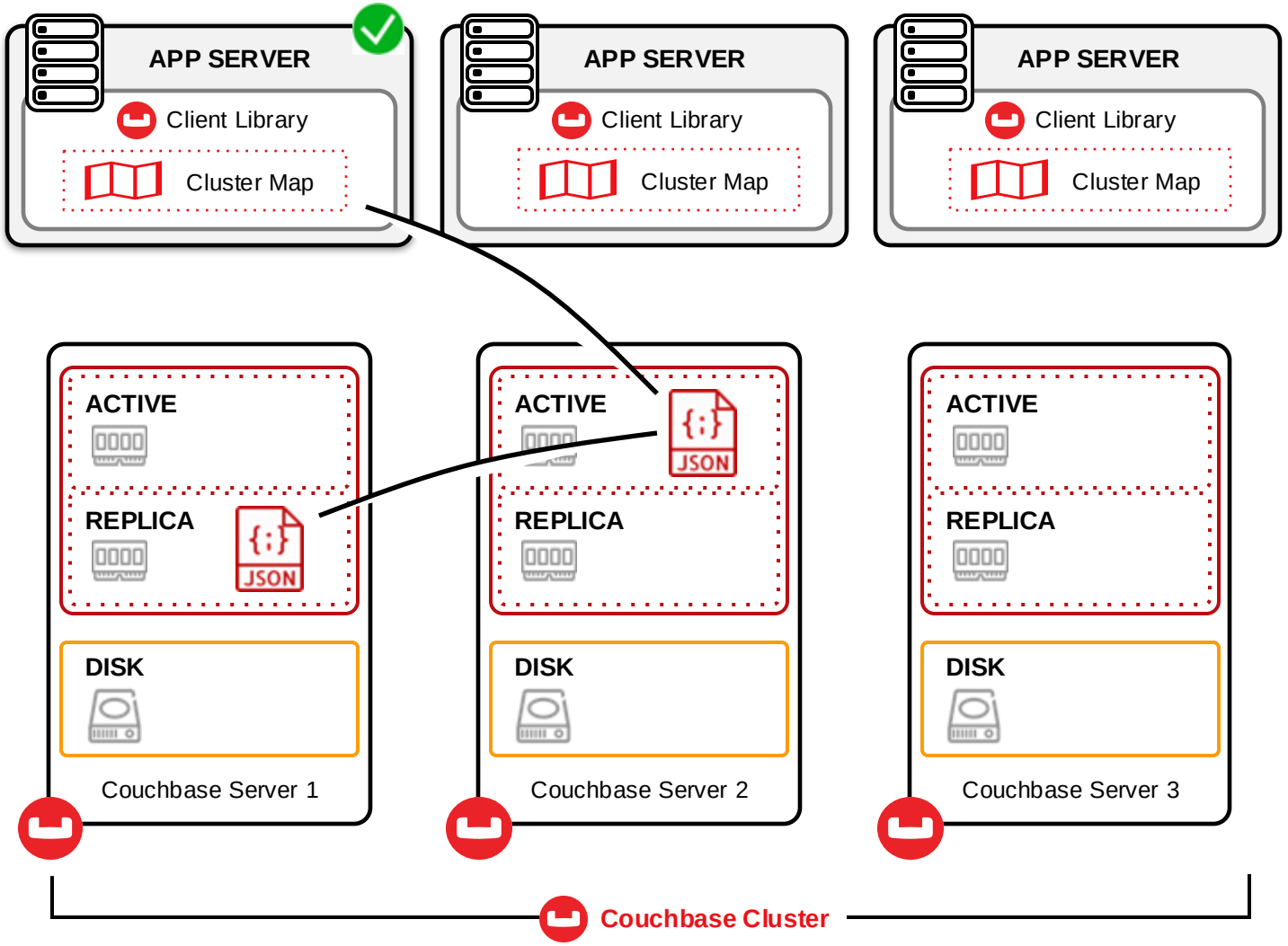
Durability



replica = 1

Level of Durability
None
Majority
MajorityAndPersistToActive
PersistToMajority

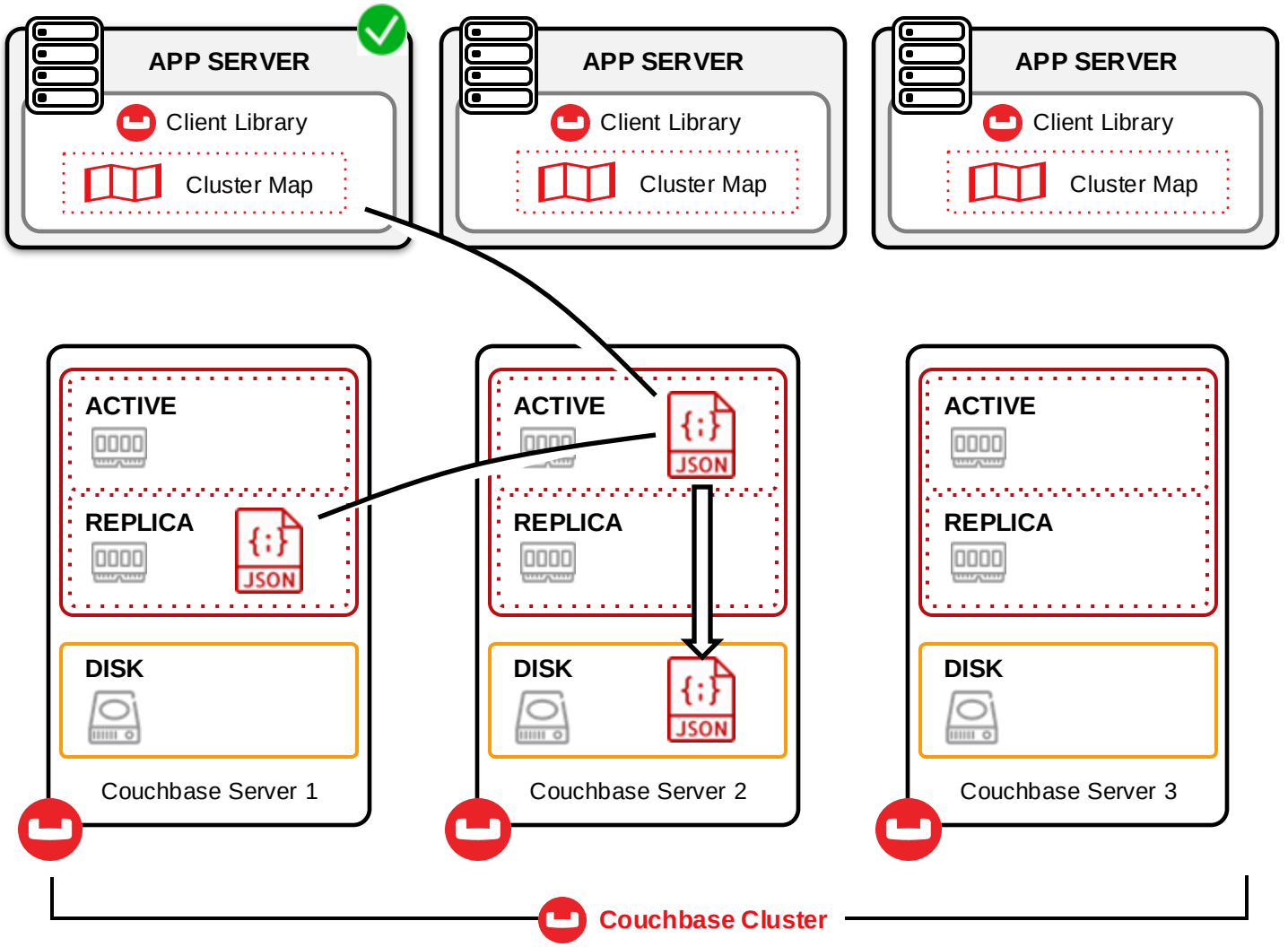
Durability



replica = 1

Level of Durability
None
Majority
MajorityAndPersistToActive
PersistToMajority

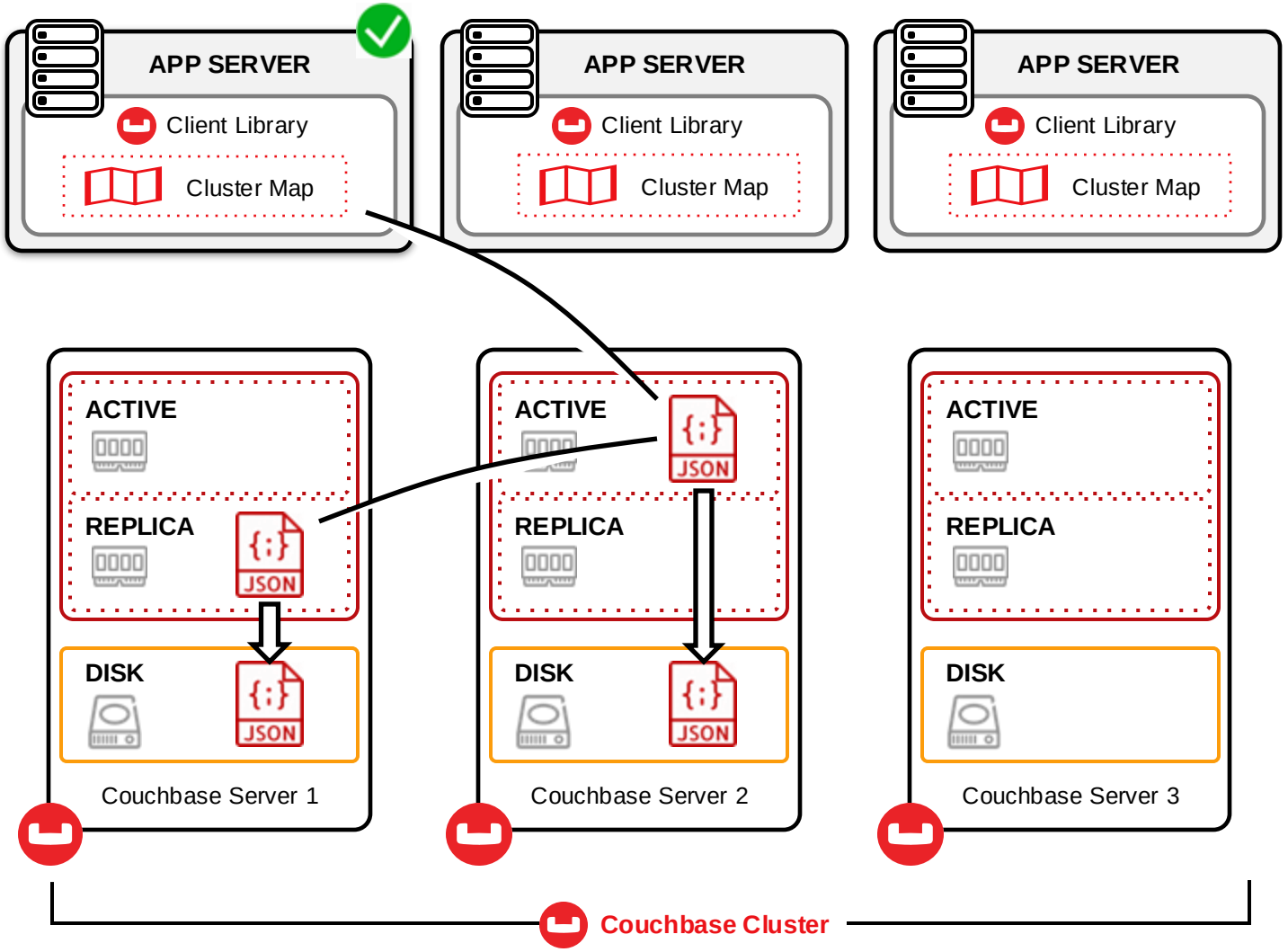
Durability



replica = 1

Level of Durability
None
Majority
MajorityAndPersistToActive
PersistToMajority

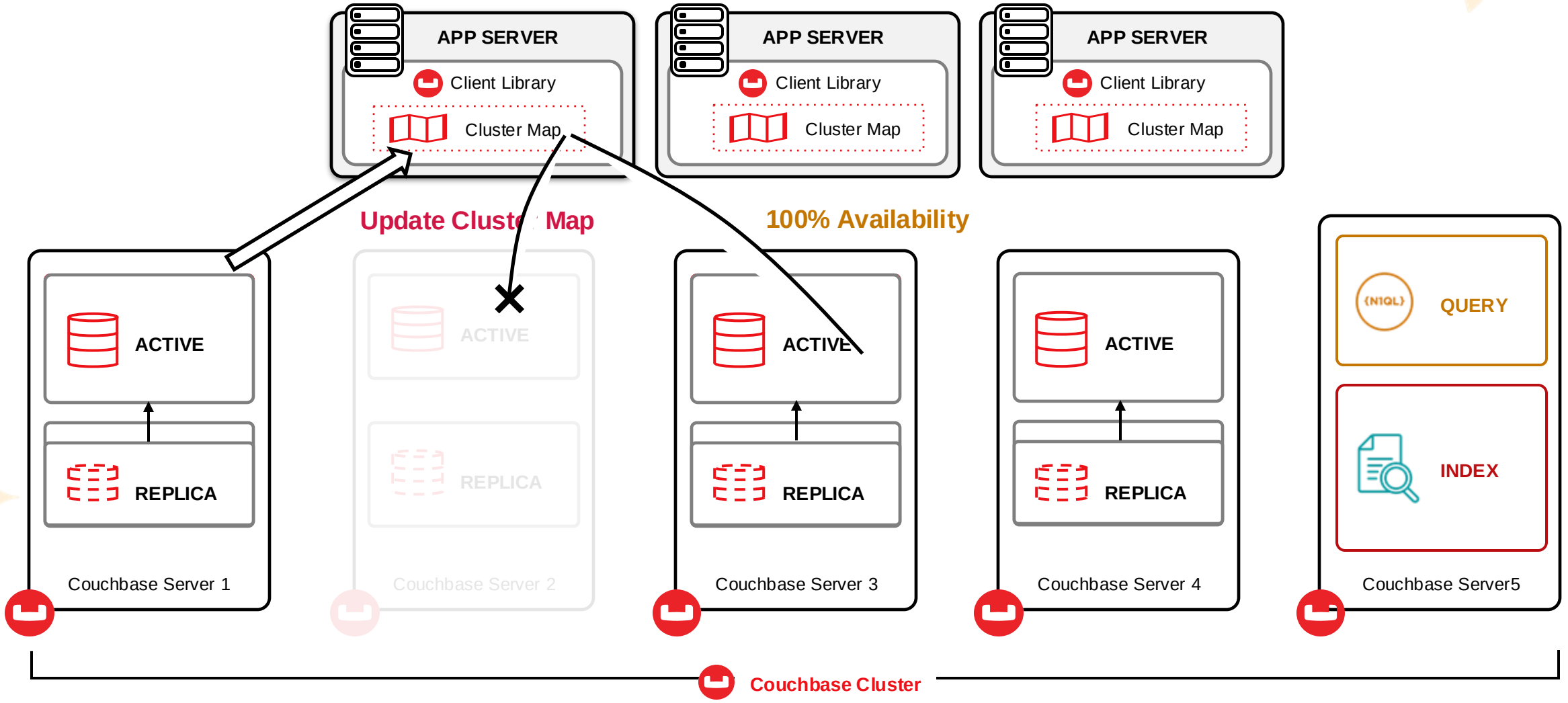
Durability



replica = 1

Level of Durability
None
Majority
MajorityAndPersistToActive
PersistToMajority

Node Failover



수고하셨습니다.

paul.son@couchbase.com

www.couchbase.com

cloud.couchbase.com

Couchbase