Couchbase

START A REVOLUTION

Developing Applications with N1QL

Nic Raboy | Senior Developer Advocate at Couchbase

AGENDA

01/ Installing and Configuring Couchbase

02/ Getting Started with N1QL and Couchbase

03/ Using N1QL with the Couchbase Developer SDKs

1 Installing and Configuring Couchbase

Obtain Couchbase Server

Package Managers

Use `apt-get`, `yum`, and other package managers to install Couchbase.

Docker

Deploy Couchbase as a container using Docker with official Couchbase images on Docker Hub.

couchbase.com

Download binaries for various operating systems from the Couchbase website.

AWS

Use an available AMI on Amazon Web Services (AWS) to deploy a single Couchbase instance, or a cluster.

Microsoft Azure

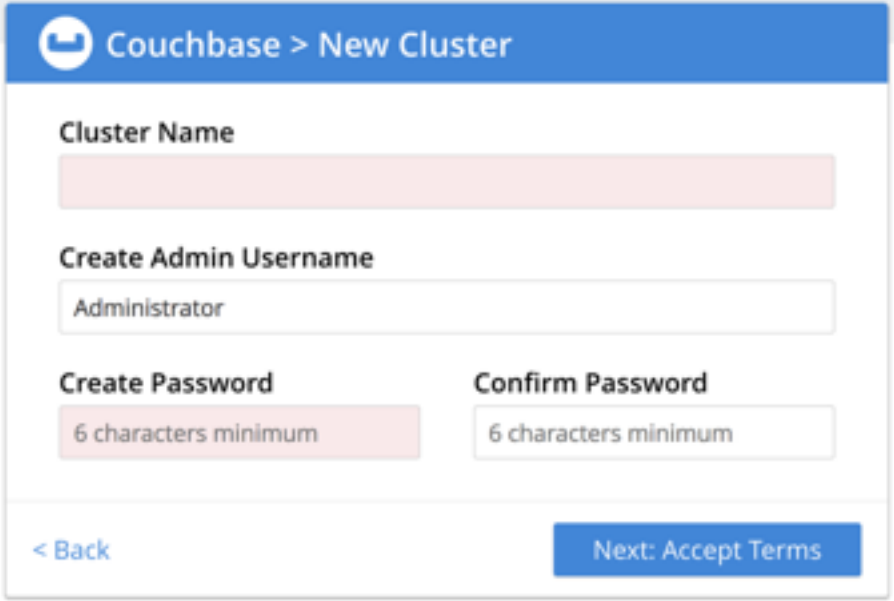
Use available images on Microsoft Azure to deploy Couchbase instances or clusters.

Couchbase with Docker

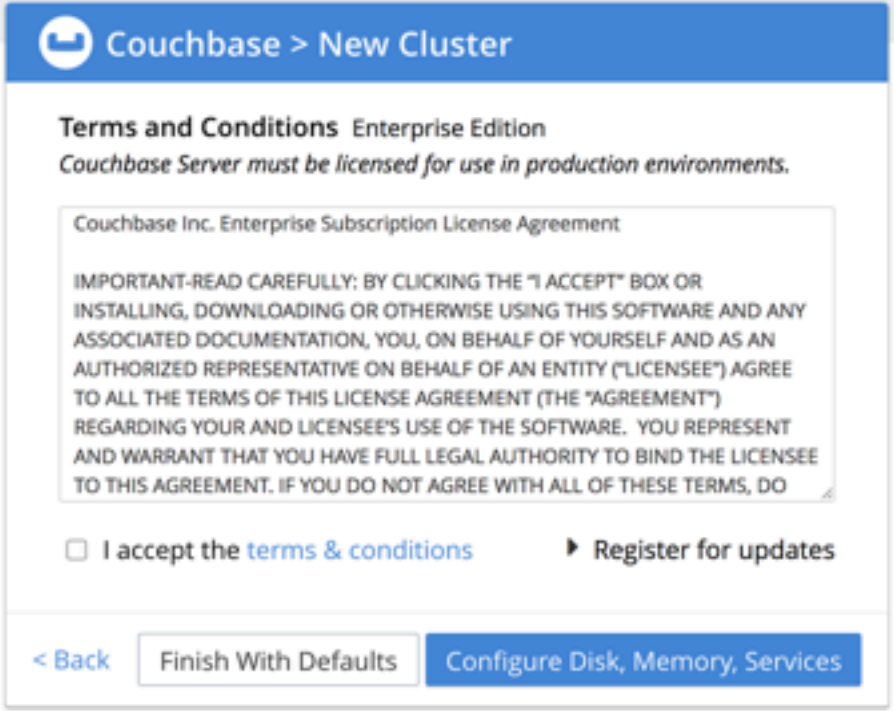
- docker pull couchbase:5.1.0
- docker run -d -p 8091-8095:8091-8095 --name couchbase couchbase:5.1.0
- http://localhost:8091

Configuring Couchbase

Configuring Couchbase



Configuring Couchbase



Configuring Couchbase

Couchbase > New Cluster / Configure

Host Name / IP Address — Usually localhost or similar

127.0.0.1

Data Disk Path — Path cannot be changed after setup

/opt/couchbase/var/lib/couchbase/data

Free: 41 GB

Indexes Disk Path — Path cannot be changed after setup

/opt/couchbase/var/lib/couchbase/data

Free: 41 GB

Couchbase Memory Quotas — Per service / per node

Data Service	2368	MB
Index Service	789	MB
Index Service	789	MB
Search Service	256	MB
Query Service	-------	

TOTAL QUOTA 3413MB

RAM Available 3947MB Max Allowed Quota 3157MB

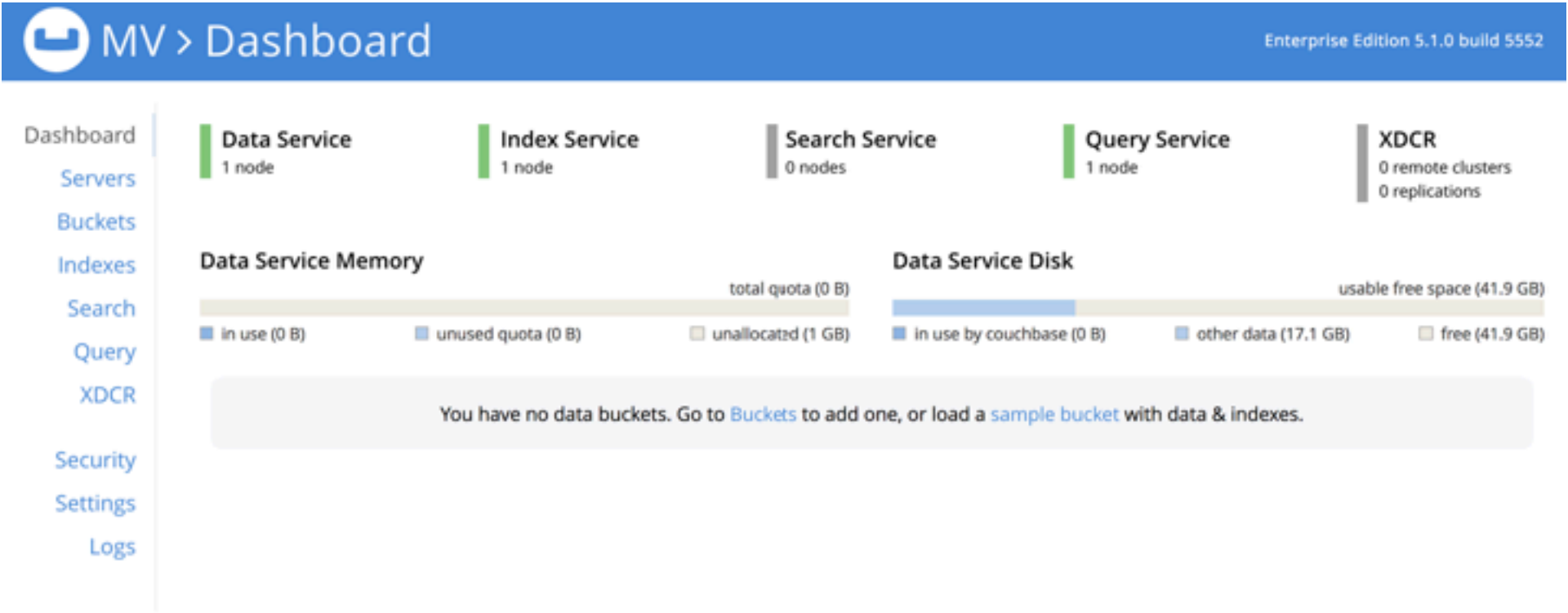
Index Storage Setting

- (•) Standard Global Secondary
- ( ) Memory-Optimized

[x] Enable software update notifications in the web console

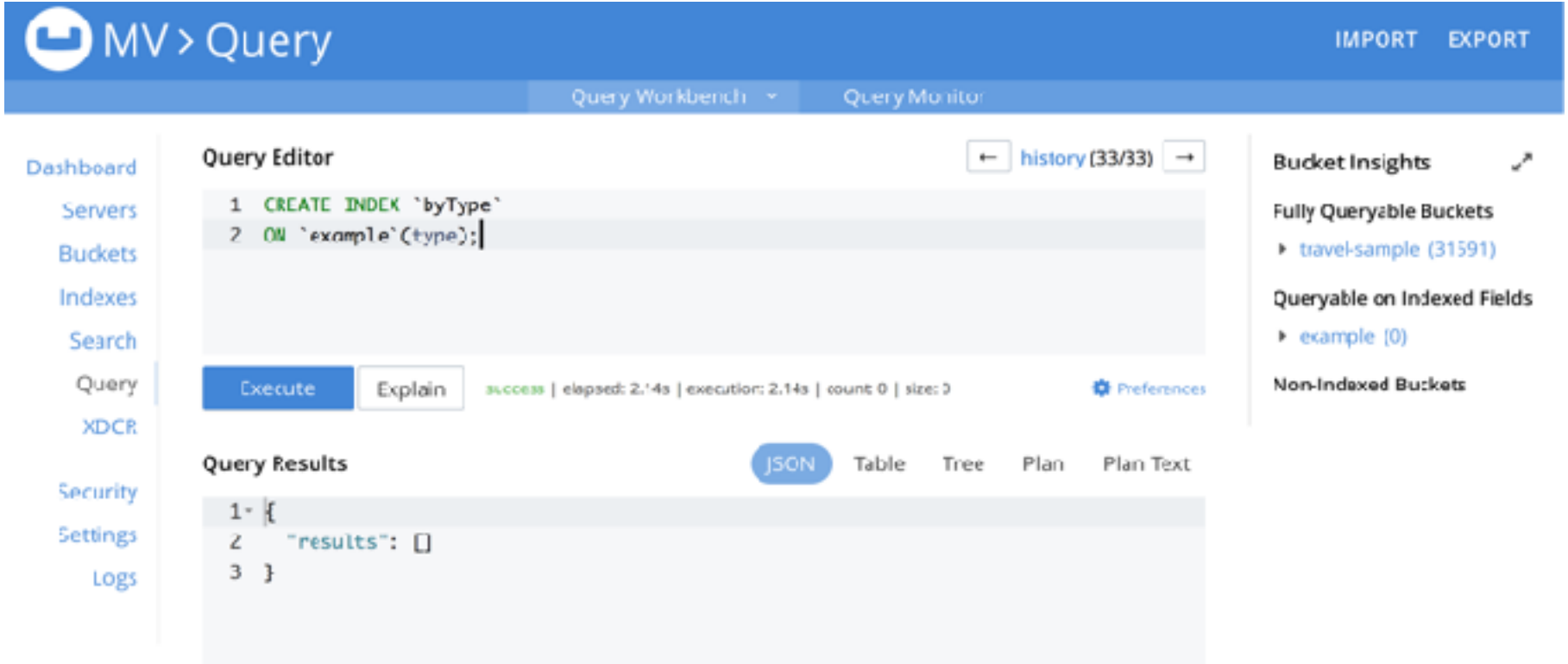
< Back | Save & Finish

Configuring Couchbase



2 | Getting Started with N1QL and Couchbase

Creating Indexes in Couchbase



Querying with Couchbase

MV > Query

IMPORT EXPORT

Query Workbench | Query Monitor

Dashboard
Servers
Buckets
Indexes
Search
Query
XDCR
Security
Settings
Logs

Query Editor

history (32/32)

```
1 SELECT airports.*
2 FROM `travel-sample` AS airports
3 WHERE type = 'airport';
```

Execute | Explain

success | elapsed: 104.02ms | execution: 103.99ms | count: 1968 | size: 805303

Preferences

Query Results

JSON | Table | Tree | Plan | Plan Text

```
1 [
2   {
3     "airportname": "Calais Dunkerque",
4     "city": "Calais",
5     "country": "France",
6     "faa": "CQF",
7     "geo": {
```

Bucket Insights

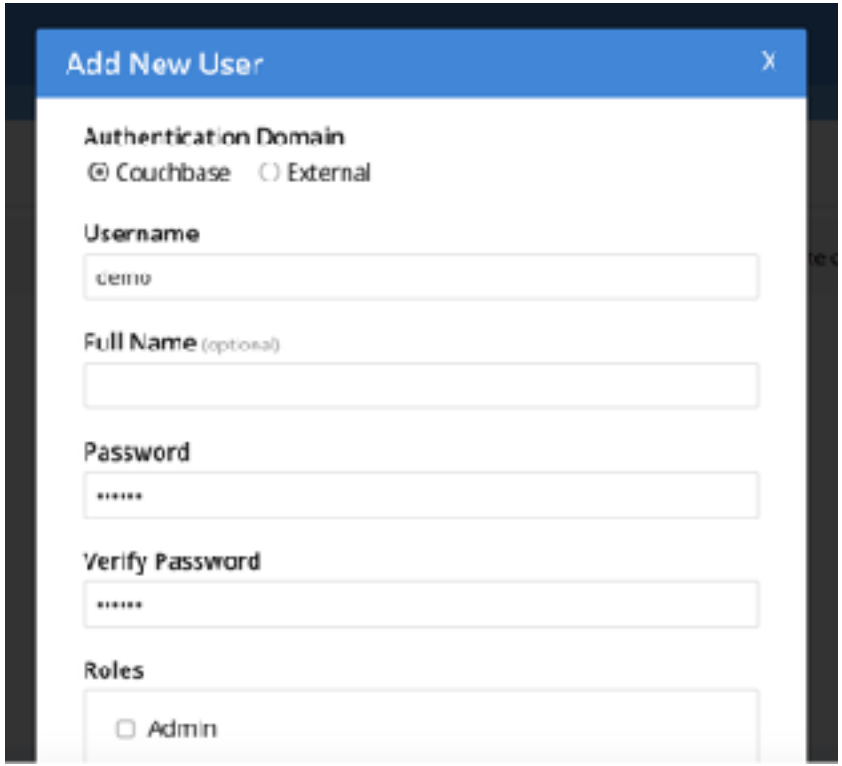
Fully Queryable Buckets

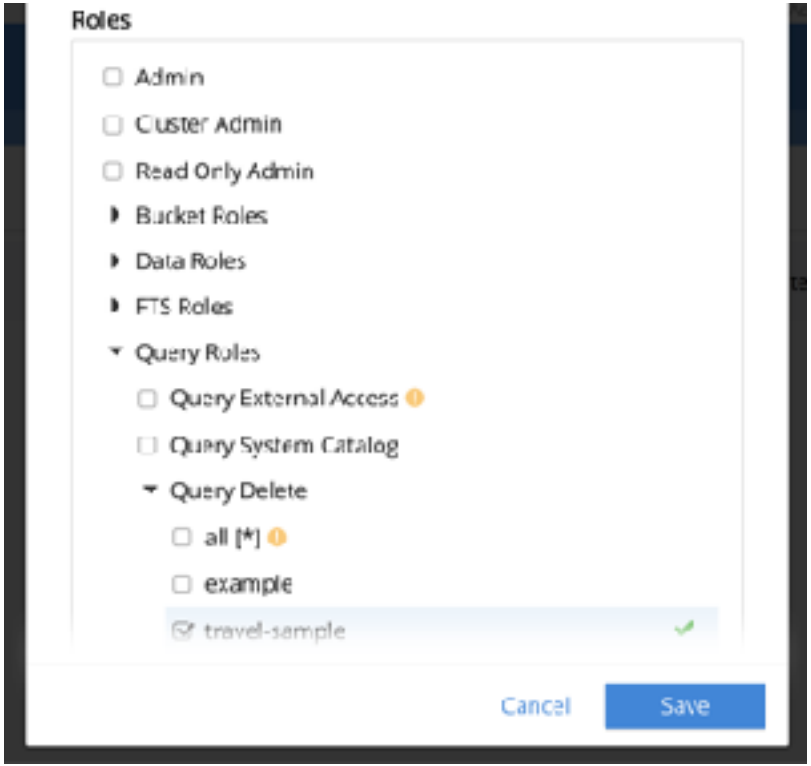
travel-sample (31591)

Queryable on Indexed Fields

Non-Indexed Buckets

Role Based Access Control (RBAC)





3

Using N1QL with the Couchbase Developer SDKs

Downloading the Developer SDK

```
// Node.js
npm install couchbase --save

// Go
go get gopkg.in/couchbase/gocb.v1

// .NET (NuGet)
Install-Package CouchbaseNetClient

// Java (Gradle)
compile group: 'com.couchbase.client', name: 'java-client', version:'2.5.1'
```

Development Frameworks

Node.js

- Express.js
- Hapi.js

Java

- Spring
- Ratpack

Python

- Django

Go

- gorilla/mux
- httprouter

SELECT Data with Node.js

```
app.get("/airports", (request, response) => {
    var statement = "SELECT `travel-sample`.* FROM `travel-sample` WHERE type = 'airport'";
    var query = Couchbase.N1qlQuery.fromString(statement);
    bucket.query(query, (error, result) => {
        if(error) {
            return response.status(500).send({ message: error.message });
        }
        response.send(result);
    });
});
```

SELECT Data with Node.js


```
[
  {
    "airportname": "Calais Dunkerque",
    "city": "Calais",
    "country": "France",
    "faa": "CQF",
    "geo": {
      "alt": 12,
      "lat": 50.962097,
      "lon": 1.954764
    },
    "icao": "LFAC",
    "id": 1254,
    "type": "airport",
    "tz": "Europe/Paris"
  },
  {
    "airportname": "Peronne St Quentin",
    "city": "Peronne",
    "country": "France",
    "faa": null,
    "geo": {
      "alt": 295,
      "lat": 49.868547,
      "lon": 3.029578
    },
    "icao": "LFAG",
    "id": 1255,
    "type": "airport",
    "tz": "Europe/Paris"
  },
```

SELECT Data with Go

```go
func GetAirlines(response http.ResponseWriter, request *http.Request) {
    response.Header().Set("Content-Type", "application/json")
    statement := `
        SELECT ` + "`" + `travel-sample` + "`" + `.*
        FROM ` + "`" + `travel-sample` + "`" + `
        WHERE country = 'United States' AND type = 'airline'
        ORDER BY name
        LIMIT 5`
    query := gocb.NewN1qlQuery(statement)
    rows, _ := bucket.ExecuteN1qlQuery(query, nil)
    var result []interface{}
    var row interface{}
    for rows.Next(&row) {
        result = append(result, row)
    }
    json.NewEncoder(response).Encode(result)
}
```

SELECT Data with Go


```
1  [
2    {
3      "callsign": "MILE-AIR",
4      "country": "United States",
5      "iata": "QS",
6      "icao": "MLA",
7      "id": 10,
8      "name": "40-Mile Air",
9      "type": "airline"
10   },
11   {
12     "callsign": "AMTRAN",
13     "country": "United States",
14     "iata": null,
15     "icao": "AMT",
16     "id": 315,
17     "name": "ATA Airlines",
18     "type": "airline"
19   },
20   {
21     "callsign": "CYCLONE",
22     "country": "United States",
23     "iata": "ZA",
24     "icao": "CYD",
25     "id": 792,
26     "name": "Access Air",
27     "type": "airline"
28   },
```

SELECT Data with Go

```go
func GetQualityHotelReviews(response http.ResponseWriter, request *http.Request) {
    response.Header().Set("Content-Type", "application/json")
    routeParams := mux.Vars(request)
    statement := `
        SELECT hotels.name, reviews.*
        FROM ` + "`" + `travel-sample` + "`" + ` AS hotels
        USE KEYS $1
        UNNEST hotels.reviews AS reviews
        WHERE reviews.ratings.Overall = 5
        LIMIT 5`
    query := gocb.NewN1qlQuery(statement)
    var params []interface{}
    params = append(params, routeParams["hotelid"])
    rows, _ := bucket.ExecuteN1qlQuery(query, params)
    var result []interface{}
    var row interface{}
    for rows.Next(&row) {
        result = append(result, row)
    }
    json.NewEncoder(response).Encode(result)
}
```

SELECT Data with Go


```
1  [
2    {
3      "author": "Alvina Abbott MD",
4      "content": "Everyone I met at the hotel made me feel very welcome and comfortable from the moment I arri
5      "date": "2013-07-02 14:32:55 +0300",
6      "name": "The George Hotel",
7      "ratings": {
8        "Cleanliness": 5,
9        "Location": 5,
10       "Overall": 5,
11       "Rooms": 3,
12       "Service": 5,
13       "Sleep Quality": 5,
14       "Value": 4
15     }
16   },
17   {
18     "author": "Carmella O'Keefe",
19     "content": "This hotel was cozy, conveniently located and allowed you to feel as a resident for a stay, 
20     "date": "2012-06-09 20:29:38 +0300",
21     "name": "The George Hotel",
22     "ratings": {
23       "Cleanliness": 5,
24       "Location": 5,
25       "Overall": 5,
26       "Rooms": 5,
27       "Service": 4,
28       "Value": 5
29     }
30   },
```

SELECT Data with Node.js

```
app.get("/hotel/reviewed-by/:name", (request, response) => {
    var statement = `
        SELECT META(hotels).id, hotels.name
        FROM \`travel-sample\` AS hotels
        WHERE ANY review IN hotels.reviews SATISFIES review.author = $1 END;`;
    var query = Couchbase.N1qlQuery.fromString(statement);
    bucket.query(query, [request.params.name], (error, result) => {
        if(error) {
            return response.status(500).send({ message: error.message });
        }
        response.send(result);
    });
});
```

SELECT Data with Node.js

```
1 [
2   {
3     "id": "hotel_10158",
4     "name": "The George Hotel"
5   }
6 ]
```

SELECT Data with Node.js

```
app.get("/flight-paths/:source/:destination", (request, response) => {
    var statement = `
        SELECT airline.name, schedule.flight, route.sourceairport, route.destinationairport
        FROM \`travel-sample\` AS route
        UNNEST route.schedule AS schedule
        JOIN \`travel-sample\` AS airline ON KEYS route.airlineid
        WHERE route.sourceairport = $1
        AND route.destinationairport = $2
        ORDER BY airline.name ASC`;
    var query = Couchbase.N1qlQuery.fromString(statement);
    bucket.query(query, [request.params.source, request.params.destination], (error, result) => {
        if(error) {
            return response.status(500).send({ message: error.message });
        }
        response.send(result);
    });
});
```

SELECT Data with Node.js


```
1  [
2    {
3      "destinationairport": "SFO",
4      "flight": "AA983",
5      "name": "American Airlines",
6      "sourceairport": "LAX"
7    },
8    {
9      "destinationairport": "SFO",
10     "flight": "AA031",
11     "name": "American Airlines",
12     "sourceairport": "LAX"
13   },
14   {
15     "destinationairport": "SFO",
16     "flight": "AA691",
17     "name": "American Airlines",
18     "sourceairport": "LAX"
19   },
20   {
21     "destinationairport": "SFO",
22     "flight": "AA516",
23     "name": "American Airlines",
24     "sourceairport": "LAX"
25   },
```

INSERT Data with Go

```go
func CreateCustomer(response http.ResponseWriter, request *http.Request) {
    response.Header().Set("Content-Type", "application/json")
    var customer Customer
    json.NewDecoder(request.Body).Decode(&customer)
    customer.Type = "customer"
    var params []interface{}
    params = append(params, uuid.NewV4().String())
    params = append(params, customer)
    statement := "INSERT INTO example (KEY, VALUE) VALUES ($1, $2) RETURNING *"
    query := gocb.NewN1qlQuery(statement)
    bucket.ExecuteN1qlQuery(query, params)
    json.NewEncoder(response).Encode(customer)
}
```

UPSERT Data with Node.js

```
app.post("/customer", (request, response) => {
    var statement = "UPSERT INTO example (KEY, VALUE) VALUES ($1, $2) RETURNING *";
    var query = Couchbase.N1qlQuery.fromString(statement);
    var data = request.body;
    data.type = "customer";
    bucket.query(query, [UUID.v4(), data], (error, result) => {
        if(error) {
            return response.status(500).send({ message: error.message });
        }
        response.send(result);
    });
});
```

THANK YOU

Expanding Your Couchbase Knowledge

- Couchbase Developer Portal – https://developer.couchbase.com
- Couchbase Blog – https://blog.couchbase.com
- Couchbase Forums – https://forums.couchbase.com
- Twitter – @couchbase, @couchbasedev, and @nraboy

APPENDIX