Navaid Queries

These are example N1QL queries that may can performed to retrieve navaid related data.

Navaid By ID

The following query will get a Navaid by its Document ID.

Query

navaid*by*document_id.n1ql

```sql
SELECT navaids.*
FROM `flight-data` AS navaids
USE KEYS 'navaid_89137'
```

Result

```
[
  {
    "_id": "navaid_89137",
    "associated_airport_icao_code": "KICT",
    "dme": {
      "channel": null,
      "elevation": null,
      "frequency_khz": null,
      "latitude": null,
      "longitude": null
    },
    "doc_type": "navaid",
    "elevation": null,
    "frequency_khz": 332,
    "geo": {
      "latitude": 37.57820129,
      "longitude": -97.4559021
    },
    "iso_country": "US",
    "magnetic_variation": 5.011,
    "navaid_id": 89137,
    "navaid_ident": "IC",
    "navaid_name": "Piche",
    "power": "MEDIUM",
    "type": "NDB",
    "usage_type": "TERMINAL"
  }
]
```

The following query will retrieve multiple Navaids by their Document ID.

Query

navaids*by*document_id.n1ql

```
SELECT navaids.*
FROM `flight-data` AS navaids
USE KEYS ['navaid_89137', 'navaid_88592']
```

Result

```
[
  {
    "_id": "navaid_89137",
    "associated_airport_icao_code": "KICT",
    "dme": {
        "channel": null,
        "elevation": null,
        "frequency_khz": null,
        "latitude": null,
        "longitude": null
      },
      "doc_type": "navaid",
      "elevation": null,
      "frequency_khz": 332,
      "geo": {
        "latitude": 37.57820129,
        "longitude": -97.4559021
      },
      "iso_country": "US",
      "magnetic_variation": 5.011,
      "navaid_id": 89137,
      "navaid_ident": "IC",
      "navaid_name": "Piche",
      "power": "MEDIUM",
      "type": "NDB",
      "usage_type": "TERMINAL"
    },
    {
      "_id": "navaid_88592",
      "associated_airport_icao_code": "KGSO",
      "dme": {
        "channel": null,
        "elevation": null,
        "frequency_khz": null,
        "latitude": null,
        "longitude": null
      },
      "doc_type": "navaid",
      "elevation": null,
      "frequency_khz": 254,
      "geo": {
        "latitude": 36.16699982,
        "longitude": -80.03559875
      },
      "iso_country": "US",
      "magnetic_variation": -7.509,
      "navaid_id": 88592,
      "navaid_ident": "GS",
      "navaid_name": "Marky",
      "power": "LOW",
      "type": "NDB",
      "usage_type": "TERMINAL"
    }
  ]
```

Navaids in a Country

The following index and queries allows for finding airports based in a given country by creating an index on the `iso_country` where the `doc_type` is `navaid`

Index

idx*navaids*iso_country.n1ql

```
CREATE INDEX idx_navaids_iso_country ON `flight-data`( iso_country )
WHERE doc_type = 'navaid'
    AND iso_country IS NOT NULL
USING GSI
```

Query

navaid*by*country.n1ql

```
SELECT navaids.*
FROM `flight-data` AS navaids
WHERE navaids.iso_country = 'DE'
    AND navaids.doc_type = 'navaid'
LIMIT 1
```

Result

```json
[
  {
    "_id": "navaid_85221",
    "associated_airport_icao_code": "EDAC",
    "dme": {
      "channel": null,
      "elevation": null,
      "frequency_khz": null,
      "latitude": null,
      "longitude": null
    },
    "doc_type": "navaid",
    "elevation": 581,
    "frequency_khz": 330,
    "geo": {
      "latitude": 50.99380112,
      "longitude": 12.52099991
    },
    "iso_country": "DE",
    "magnetic_variation": 1.561,
    "navaid_id": 85221,
    "navaid_ident": "ABU",
    "navaid_name": "Altenburg",
    "power": "LOW",
    "type": "NDB",
    "usage_type": "TERMINAL"
  }
]
```

Now that we know we can retrieve all navaids in a given country by querying on the `iso_country`.

Query

navaids*by*country.n1ql

```sql
SELECT navaids.navaid_id, navaids.navaid_ident, navaids.navaid_name, navaids.type,
    navaids.frequency_khz, navaids.geo, navaids.elevation, navaids.usage_type
FROM `flight-data` AS navaids
WHERE navaids.doc_type = 'navaid'
    AND navaids.iso_country = 'DE'
ORDER BY navaids.navaid_name ASC
```

Results

```json
[
  {
    "elevation": 1434,
    "frequency_khz": 111200,
    "geo": {
      "latitude": 49.21440125,
      "longitude": 11.22140026
    },
    "navaid_id": 85395,
    "navaid_ident": "ALB",
    "navaid_name": "Allersberg",
    "type": "VOR-DME",
    "usage_type": "BOTH"
  },
  {
    "elevation": 66,
    "frequency_khz": 115800,
    "geo": {
      "latitude": 53.63529968,
      "longitude": 9.994139671
    },
    "navaid_id": 85404,
    "navaid_ident": "ALF",
    "navaid_name": "Alster",
    "type": "DME",
    "usage_type": "TERMINAL"
  },
  ...
]
```

Additionally we can retrieve an aggregate count of the number of navaids in a given country.

Query

total*navaids*by_country.n1ql

```sql
SELECT COUNT(1) AS total_navaids
FROM `flight-data` AS navaids
WHERE navaids.iso_country = 'DE'
    AND navaids.doc_type = 'navaid'
```

Result

```
[
  {
    "total_navaids": 215
  }
]
```

Navaid Codes

The following queries allows for finding navaids by their Ident Code.

Just like Airlines and Airports, our Codes model is keyed by `{{designation}}_code_{{code}}` i.e. `navaid_code_ATL`. Because of how these documents are keyed, we do not even need an index. Using this predictive key pattern we use the code as part of the key name on the codes document.

Query

Query by the Ident code

navaid*by*ident_code.n1ql

```
SELECT navaids.navaid_id, navaids.navaid_ident, navaids.navaid_name, navaids.type,
    navaids.frequency_khz, navaids.geo, navaids.elevation, navaids.usage_type
FROM `flight-data` AS codes
USE KEYS 'navaid_code_ATL'
INNER JOIN `flight-data` AS navaids ON KEYS 'navaid_' || TOSTRING( codes.id )
LIMIT 1
```

Result

```
[
  {
    "elevation": 1000,
    "frequency_khz": 116900,
    "geo": {
      "latitude": 33.6291008,
      "longitude": -84.43509674
    },
    "navaid_id": 85664,
    "navaid_ident": "ATL",
    "navaid_name": "Atlanta",
    "type": "VORTAC",
    "usage_type": "BOTH"
  }
]
```

Navaids Near a Given Airport

For this query we want to find all navaids within a given radius of a given airport code.

Since we are going to be querying on the ISO Country, Latitude and Longitude of a given airport we need to create an index.

Index

idx*navaids*distance.n1ql

```
CREATE INDEX idx_navaids_distance ON `flight-data`( iso_country, geo.latitude, geo.longi
WHERE doc_type = 'navaid'
    AND iso_country IS NOT NULL
    AND geo.latitude IS NOT NULL
    AND geo.longitude IS NOT NULL
USING GSI
```

This query is based on a MySQL example provided by Ollie Jones.

To perform this query we need to provide 5 pieces of information to the query, these are represented in the query below as `{{tokens}}`

Input

- The `iso_country`
- The Source Airports

    - `latitude` i.e. `36.09780121`
    - `longitude` i.e. `-79.93730164`

- A `distance_unit`

    - Kilometers: 111.045
    - Miles: 69

- A `radius` in which to contain results in, i.e. `100`

Radius Query

```
SELECT results.navaid_ident, results.navaid_code, results.type, results.frequency_khz, r
    results.associated_airport_code, ROUND( results.distance, 2 ) AS distance
FROM (
    SELECT navaids.navaid_ident AS navaid_code, navaids.type, navaids.frequency_khz, nav
        navaids.associated_airport_icao_code AS associated_airport_code,
        /* calculate the distance */
        {{distance_unit}} * DEGREES(ACOS(COS(RADIANS( {{source_latitude}} ))
        * COS(RADIANS( navaids.geo.latitude ))
        * COS(RADIANS( {{source_longitude}} ) - RADIANS( navaids.geo.longitude ))
        + SIN(RADIANS( {{source_latitude}} ))
        * SIN(RADIANS( navaids.geo.latitude )))) AS distance
    FROM `flight-data` AS navaids
    WHERE navaids.iso_country = '{{iso_country}}'
        /* limit results to latitudes within {{distance}} north or south of the source l
        AND navaids.geo.latitude BETWEEN
            {{source_latitude}} - ({{radius}} / {{distance_unit}})
            AND
            {{source_latitude}} + ({{radius}} / {{distance_unit}})
        /* limit results to longitudes within {{distance}} east or west of the source lo
        AND navaids.geo.longitude BETWEEN
            {{source_longitude}} - ({{radius}} / ( {{distance_unit}} * COS(RADIANS( {{so
            AND
            {{source_longitude}} + ({{radius}} / ( {{distance_unit}} * COS(RADIANS( {{so
        AND navaids.doc_type = 'navaid'
    ) AS results
WHERE results.distance <= {{radius}} /* remove any of the results that are not within th
ORDER BY results.distance ASC /* sort the results by closest distance */
```

To provide the source airports `iso_country`, `latitude`, and `longitude` we can use the Airport Codes query.

Source Airport Query

[source*airport*ICT.n1ql][queries/airports/source*airport*ICT.n1ql]

```
SELECT airports.iso_country, airports.geo.latitude AS latitude, airports.geo.longitude A
FROM `flight-data` AS codes
USE KEYS 'airport_code_ICT'
INNER JOIN `flight-data` AS airports
    ON KEYS 'airport_' || TOSTRING( codes.id )
LIMIT 1
```

Result

```
[
  {
    "iso_country": "US",
    "latitude": 37.64989853,
    "longitude": -97.43309784
  }
]
```

Next we replace the tokens from our base radius query with the returned values.

Navaids Near a Given Airport in Miles Query

For our example we want to find any airports within 100 miles of "ICT". Our `{{distance_unit}}` is miles, this value needs to be `69` and our `{{radius}}` is `100`. Replace the `{{source_latitude}}`, `{{source_longitude}}` and `{{iso_country}}` with the values from the previous query.

navaids*near*ICT*by*miles.n1ql

```sql
SELECT results.navaid_ident, results.navaid_code, results.type, results.frequency_khz, r
    results.associated_airport_code, ROUND( results.distance, 2 ) AS distance
FROM (
    SELECT navaids.navaid_ident AS navaid_code, navaids.type, navaids.frequency_khz, nav
        navaids.associated_airport_icao_code AS associated_airport_code,
        69 * DEGREES(ACOS(COS(RADIANS( 37.64989853 ))
        * COS(RADIANS( navaids.geo.latitude ))
        * COS(RADIANS( -97.43309784 ) - RADIANS( navaids.geo.longitude ))
        + SIN(RADIANS( 37.64989853 ))
        * SIN(RADIANS( navaids.geo.latitude )))) AS distance
    FROM `flight-data` AS navaids
    WHERE navaids.iso_country = 'US'
        AND navaids.geo.latitude BETWEEN
            37.64989853 - ( 50 / 69 )
            AND
            37.64989853 + ( 50 / 69 )
        AND navaids.geo.longitude BETWEEN
            -97.43309784 - ( 50 / ( 69 * COS(RADIANS( 37.64989853 ))))
            AND
            -97.43309784 + ( 50 / ( 69 * COS(RADIANS( 37.64989853 ))))
        AND navaids.doc_type = 'navaid'
    ) AS results
WHERE results.distance <= 50
ORDER BY results.distance ASC
```

Navaids Near a Given Airport in Miles Results

```json
[
  {
    "associated_airport_code": "KICT",
    "distance": 5.1,
    "frequency_khz": 332,
    "navaid_code": "IC",
    "type": "NDB",
    "usage_type": "TERMINAL"
  },
  {
    "associated_airport_code": "KIAB",
    "distance": 9.21,
    "frequency_khz": 116500,
    "navaid_code": "IAB",
    "type": "TACAN",
    "usage_type": "BOTH"
  },
  {
    "associated_airport_code": "KICT",
      "distance": 10.54,
      "frequency_khz": 113800,
      "navaid_code": "ICT",
      "type": "VORTAC",
      "usage_type": "BOTH"
    },
    {
      "associated_airport_code": null,
      "distance": 22.63,
      "frequency_khz": 414,
      "navaid_code": "EGT",
      "type": "NDB",
      "usage_type": "TERMINAL"
    },
    {
      "associated_airport_code": null,
      "distance": 29.87,
      "frequency_khz": 281,
      "navaid_code": "EWK",
      "type": "NDB",
      "usage_type": "TERMINAL"
    },
    {
      "associated_airport_code": null,
      "distance": 34.83,
      "frequency_khz": 383,
      "navaid_code": "EQA",
      "type": "NDB",
      "usage_type": "TERMINAL"
    },
    {
      "associated_airport_code": null,
      "distance": 35.21,
      "frequency_khz": 395,
      "navaid_code": "CA",
      "type": "NDB",
      "usage_type": "TERMINAL"
    },
    {
      "associated_airport_code": "KHUT",
      "distance": 36.32,
      "frequency_khz": 116800,
      "navaid_code": "HUT",
      "type": "VOR-DME",
      "usage_type": "LO"
```

```
  },
  {
    "associated_airport_code": "KHUT",
    "distance": 42.33,
    "frequency_khz": 404,
    "navaid_code": "HU",
    "type": "NDB",
    "usage_type": "TERMINAL"
  }
]
```

Navaids Near a Given Airport in Kilometers Query

For our example we want to find any airports within 75 kilometers of "Berlin" (TXL). Our `{{distance_unit}}` is kilometers, this value needs to be `111.045` and our `{{radius}}` is `75`. Replace the `{{source_latitude}}`, `{{source_longitude}}` and `{{iso_country}}` with the values from the previous query.

Source Airport Query

source*airport*TXL.n1ql

```
SELECT airports.iso_country, airports.geo.latitude AS latitude, airports.geo.longitude A
FROM `flight-data` AS codes
USE KEYS 'airport_code_TXL'
INNER JOIN `flight-data` AS airports
    ON KEYS 'airport_' || TOSTRING( codes.id )
LIMIT 1
```

Result

```
[
  {
    "iso_country": "DE",
    "latitude": 52.55970001,
    "longitude": 13.2876997
  }
]
```

Navaids Near a Given Airport in Kilometers Query

navaids*near*TXL*by*kilometers.n1ql

```sql
SELECT results.navaid_ident, results.navaid_code, results.type, results.frequency_khz, r
    results.associated_airport_code, ROUND( results.distance, 2 ) AS distance
FROM (
    SELECT navaids.navaid_ident AS navaid_code, navaids.type, navaids.frequency_khz, nav
        navaids.associated_airport_icao_code AS associated_airport_code,
        111.045 * DEGREES(ACOS(COS(RADIANS( 52.55970001 ))
        * COS(RADIANS( navaids.geo.latitude ))
        * COS(RADIANS( 13.2876997 ) - RADIANS( navaids.geo.longitude ))
        + SIN(RADIANS( 52.55970001 ))
        * SIN(RADIANS( navaids.geo.latitude )))) AS distance
    FROM `flight-data` AS navaids
    WHERE navaids.iso_country = 'DE'
        AND navaids.geo.latitude BETWEEN
            52.55970001 - ( 75 / 111.045 )
            AND
            52.55970001 + ( 75 / 111.045 )
        AND navaids.geo.longitude BETWEEN
            13.2876997 - ( 75 / ( 111.045 * COS(RADIANS( 52.55970001 ))))
            AND
            13.2876997 + ( 75 / ( 111.045 * COS(RADIANS( 52.55970001 ))))
        AND navaids.doc_type = 'navaid'
    ) AS results
WHERE results.distance <= 75
ORDER BY results.distance ASC
```

Airport Navaids Radius in Kilometers Results

```json
[
  {
    "associated_airport_code": "EDDT",
    "distance": 0.2,
    "frequency_khz": 112300,
    "navaid_code": "TGL",
    "type": "VOR-DME",
    "usage_type": "BOTH"
  },
  {
    "associated_airport_code": null,
    "distance": 7.9,
    "frequency_khz": 414,
    "navaid_code": "DLS",
    "type": "NDB",
    "usage_type": "LO"
  },
  {
    "associated_airport_code": "EDDT",
      "associated_airport_code": "EDDT",
      "distance": 9.37,
      "frequency_khz": 392,
      "navaid_code": "RW",
      "type": "NDB",
      "usage_type": "TERMINAL"
    },
    {
      "associated_airport_code": "EDDT",
      "distance": 9.45,
      "frequency_khz": 321,
      "navaid_code": "GL",
      "type": "NDB",
      "usage_type": "TERMINAL"
    },
    {
      "associated_airport_code": "EDDB",
      "distance": 25.23,
      "frequency_khz": 114400,
      "navaid_code": "SDD",
      "type": "DME",
      "usage_type": "TERMINAL"
    },
    {
      "associated_airport_code": "EDDB",
      "distance": 26.62,
      "frequency_khz": 362,
      "navaid_code": "SLN",
      "type": "NDB",
      "usage_type": "LO"
    },
    {
      "associated_airport_code": null,
      "distance": 40.27,
      "frequency_khz": 114550,
      "navaid_code": "LWB",
      "type": "VOR-DME",
      "usage_type": "BOTH"
    },
    {
      "associated_airport_code": null,
      "distance": 59.33,
      "frequency_khz": 113300,
      "navaid_code": "FWE",
      "type": "VOR-DME",
      "usage_type": "BOTH"
```

```json
  },
  {
    "associated_airport_code": "EDAZ",
    "distance": 62.85,
    "frequency_khz": 115150,
    "navaid_code": "KLF",
    "type": "VOR-DME",
    "usage_type": "BOTH"
  }
]
```