**GEOSPATIAL DATA**

The geospatial data, is a specific type of data that is used to query the locations, nearby places etc. right from the mongodb. The geospatial data must be stored in a particular format know as **GeoJson** the syntax is as follows**:**

{

…

<field\_name>: {

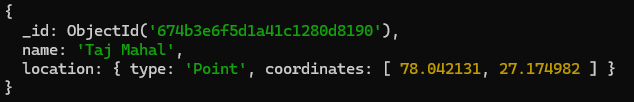
type: ‘geoJson-type’,

coordinates: [<longitude>,<latitude>]

}

…

}



Here the “location” field contains our geoJson object.

Note: The longitude must comes before the latitude in GeoJosn object.

**Official Docs :** [**GeoJSON Objects - MongoDB Manual v8.0**](https://www.mongodb.com/docs/manual/reference/geojson/?msockid=0263853e7e166b3d148e915b7f8d6a40)

**Editable Maps:** [**My Maps – About – Google Maps**](https://www.google.com/maps/about/mymaps/)

**The GeoJson Index (2dsphere):**

To work with the geospatial data that we have stored in our documents as GeoJson embedded/nested object, must be indexed.

* db.<collection>.createIndex({ ‘location’: ‘2dsphere’ })

[Will create a geospatial index on the field location that holds our geoJson object.]

**The Geospatial Queries:**

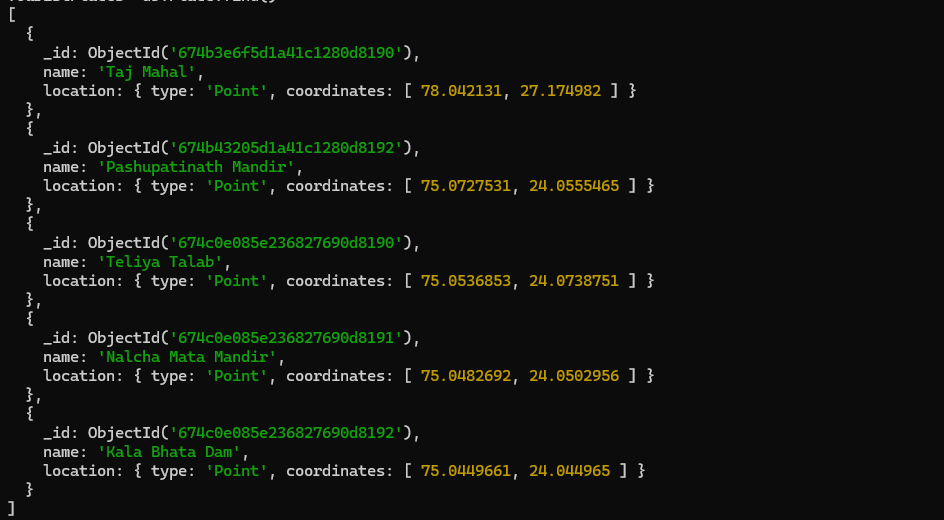
Here we will list down the types of queries we can run on geospatial data stored in our documents.

1. **To Find all the places nearby to a particular location (‘Point to Point’):**

Prerequisites: Source Location Coordinates (maybe a person current location).

* 1. It always return sorted result. (distance: lowest to highest).

Example: We have a collection “Place” which have different places along with their location in geoJson Format (type: ‘Point’) (below image). We have to find all the documents(places) that’s within 1000 meters distance from the source location coordinates.



* + db.<collection>.find({ <location-field-name>: { $near: { $geometry: { type: ‘Point’, coordinates: [source-location-longitude, source-location-latitude] }, $minDistance: 10, $maxDistance: 1000 } } })

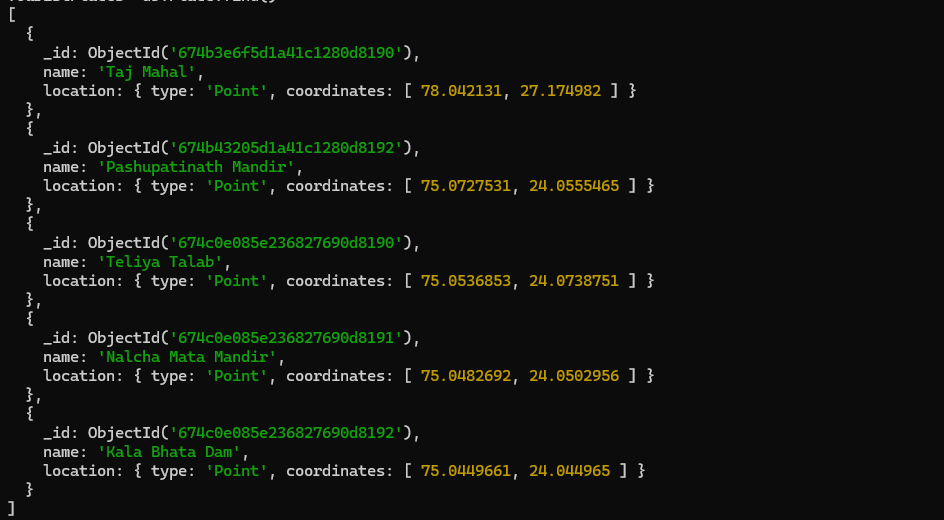
[This query will find all the places that is within 10 to 1000 meters of the source location.]

1. **To find a certain area contains which places. (‘Polygon to Point’):**

Prerequisites: The location coordinates of all the four corners of the area,

c1 = [long1,lat1], c2 = [long2, lat2], c3 = [long3, lat3], c4 = [long4, lat4].

Example: We have a collection ‘Place’ which have different places along with their location in geoJson Format (type: ‘Point’) (below image). We have to find the Places that’s falls within a certain area.



* + db.<collection>.find({ <location-field-name>: { $geoWithin: { $geometry: { type : ‘Polygon’, coordinates: [ [ c1, c2, c3, c4, c1 ] ] } } })

[This query will return as all the places that falls inside the area.

Here c1, c2, c3 and c4 belongs to the coordinates of all the four corners of this area.

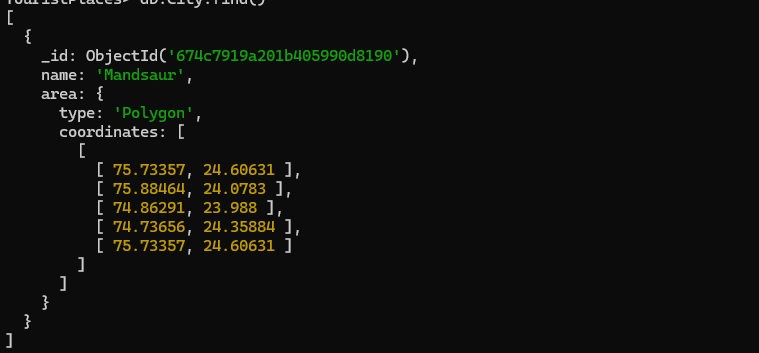


(A Area/Polygon)

1. **To Find a certain location falls in which area. (geoJson Type : ‘Point To Polygon’):**

Prerequisites:  The source location coordinates.

Example: We have a collection ‘City’ which have different cities along with their areas in geoJson Form(type: ‘Polygon’) (below image). We have a the area a certain source location falls in.



* + db.<collection>.find({ <area-field-name>: { $geoIntersects: { $geometry: { type: ‘Point’, coordinates: [long1, lat1] } } } })

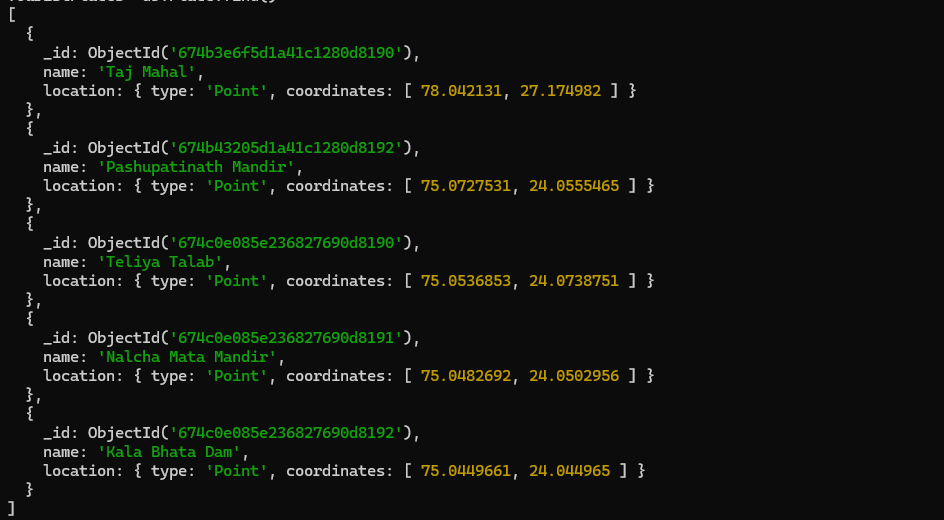
[This query return all the documents whose area contain the source location]

1. **To find all the places in a certain radius from a location (similar to $near):**

Prerequisites:  The source location coordinates and the distance to calculate the radius.

* 1. This Query is similar to $near query the only difference between them is that the $near query returns result in sorted order and this query returns unsorted result.
  2. This query requires the radius to be calculated manually, see docs for more.

Example: We have a collection “Place” which have different places along with their location in geoJson Format (type: ‘Point’) (below image). We have to find all the documents(places) that’s within 1 mile distance from the source location coordinates.



* + db.<collection>.find({ <location-field-name>: { $geoWithin: { $centerSphere: [ [ source-long, source-lat ], 1/ 3963.2] } } })

[returns all the places that is within 1 mile radius of the source location]

**Geospatial Operators :**

* **$near:**  This operator is used to find the nearby places.
* **$geometry:**  This operator is used to take the source location geoJson object for matching.
* **$minDistance:**  This operator takes the minimum distance in meters.
* **$maxDistance:**  This operator takes the maximum distance in meters.
* **$geoWithin:**  This operator is used to find places within a certain area aka polygon.
* **$geoIntersects:**  This operator is used to find areas that contains a location.
* **$centerSphere:** This operator is used to find places in a certain radius.