HW 4: Geodatabases and Spatial Joins

Μέρος Α

Για την απάντηση του μέρους Α της εργασίας , φορτώσαμε την βάση δεδομένων στο QGIS και πραγματοποιήσαμε τα ακόλουθα ερωτήματα :

query1 – Νομοί από τους οποίους περνά το δίκτυο του ΟΣΕ

```
SELECT DISTINCT n.*
FROM nomoi n
JOIN sidhrodromiko_diktyo o
  ON st_intersects(n.geometry, o.geometry);
```

query2 – Νομοί με τους οποίους συνορεύει ο νομός Θεσσαλονίκης και έχουν στο έδαφος τους λίμνη

```
SELECT DISTINCT n2.*
FROM nomoi AS n1
JOIN nomoi AS n2
  ON st_touches(n1.geometry, n2.geometry)
JOIN limnes AS l
  ON st_intersects(n2.geometry, l.geometry)
WHERE n1.name_gr = 'Ν. ΘΕΣΣΑΛΟΝΙΚΗΣ';
```

query3 – Νομοί που δεν έχουν αεροδρόμιο

```
SELECT n.*
FROM nomoi AS n
WHERE NOT EXISTS (
  SELECT 1
  FROM aerodromia AS a
  WHERE st_intersects(n.geometry, a.geometry)
);
```

Μέρος Β

Για την απάντηση του μέρους β, κάναμε εξαγωγή των επιπέδων της geodatabase σε αρχεία GeoJson, δημιουργήσαμε μια βάση στην MongoDB και κάναμε εισαγωγή των παραπάνω json αρχείων στην βάση ως συλλογές. Ακολουθούν τα ερωτήματα στην MongoDB:

query0a – Νομοί που έχουν οικισμούς που το όνομά τους ξεκινά από Ω

```
var oikoismoiOmega = db.oikismoi.find({
  "properties.name_oik": { $regex: "^Ω" }
}).toArray();

oikoismoiOmega.forEach(function(oikismos) {
  var point = oikismos.geometry;

  var nomos = db.nomoi.findOne({
    geometry: {
      $geoIntersects: {
        $geometry: point
      }
    }
  });

  if (nomos) {
    print(nomos.properties.name_gr);
  }
});
```

query0b – Πρωτεύουσες που απέχουν ως 10χλμ από τον ποταμό Αλιάκμονα

```
var nearCities = new Set()
db.poleis.find().forEach(c => {
  var coordsC = c.geometry?.coordinates;
  if (!Array.isArray(coordsC)) return;

  var nearPotamoi = db.potamoi.find({
    "properties.name": /ΑΛΙΑΚ/i,
    geometry: {
      $near: {
        $geometry: {
          type: "Point",
          coordinates: coordsC
        },
        $maxDistance: 10000
      }
    }
  }).toArray();

  nearPotamoi.forEach(p => {
    nearCities.add(c.properties.onoma);
  });
});

print([...nearCities]);
```

query1 – Νομοί από τους οποίους περνά το δίκτυο του ΟΣΕ

```
var oseLines = db.sidhrodromiko_diktyo.find().toArray();

db.nomoi.find().forEach(function(nomos) {
  for (var i = 0; i < oseLines.length; i++) {
    var oseLine = oseLines[i];
    var intersects = db.nomoi.findOne({
      _id: nomos._id,
      geometry: {
        $geoIntersects: {
          $geometry: oseLine.geometry
        }
      }
    });

    if (intersects) {
      print(nomos.properties.name_gr);
      break;
    }
  }
});
```

query2 – Νομοί με τους οποίους συνορεύει ο νομός Θεσσαλονίκης και έχουν στο έδαφος τους λίμνη

```
var thess = db.nomoi.findOne({ "properties.name_gr": "Ν. ΘΕΣΣΑΛΟΝΙΚΗΣ" });
var borderPoints = thess.geometry.coordinates.flat(2);
var results = new Map();

borderPoints.forEach(([lon, lat]) => {
  db.nomoi.find({
    _id: { $ne: thess._id },
    geometry: {
      $near: {
        $geometry: {
          type: "Point",
          coordinates: [lon, lat]
        },
        $maxDistance: 1000
      }
    }
  }).forEach(n => results.set(n._id.str, n));
});

const nomoiWithLimnes = [];

[...results.values()].forEach(nomos => {
  const hasLimni = db.limnes.findOne({
    geometry: {
      $geoWithin: {
        $geometry: nomos.geometry
      }
    }
  });

  if (hasLimni) {
    nomoiWithLimnes.push(nomos.properties.name_gr);
  }
});

print(nomoiWithLimnes);
```

query3 – Νομοί που δεν έχουν αεροδρόμιο

```
var aerodromia = db.aerodromia.find().toArray();

db.nomoi.find().forEach(function(nomos) {
  var has_aerodromio = false;

  for (var i = 0; i < aerodromia.length; i++) {
    if (db.aerodromia.findOne({
      geometry: {
        $geoWithin: {
          $geometry: nomos.geometry
        }
      }
    })) {
      has_aerodromio = true;
      break;
    }
  }

  if (!has_aerodromio) {
    print(nomos.properties.name_gr);
  }
});
```