SISTEMES DISTRIBUÏTS A GRAN ESCALA	
Segon Semestre	Màster d'Enginyeria Informàtica
Pràctica – 2n lliurament	

Criteris d’avaluació i condicions de lliurament

Data límit de lliurament: 10-5-2018.

Cal lliurar la pràctica en aquesta data i a la bústia *Lliurament i registre d’AC* de la vostra aula.

A l’hora de presentar-la, l’aplicatiu la reanomenarà automàticament a partir del vostre nom i cognoms, denominació de la pràctica i data de lliurament. Comproveu que el nom assignat és correcte.

Cal desar la solució en un **fitxer .ZIP** que contingui únicament els següents fitxers:

1. Respostes a la pràctica en **format PDF**
2. **Fitxer mapReduce.js** completat tal com es demana a la pregunta 1

Cal ser precís i clar en les respostes, respectant les limitacions d’espai indicades.

Us recordem també que la pràctica és un **treball estrictament individual.**

Els criteris d'avaluació seran els següents:

- Primera pregunta opteu a una B
- Primera pregunta + segona pregunta: opteu a una A
- Només segona pregunta: opteu a una C-

En l'avaluació es valorarà que l'argumentació (en els casos que es demana) sigui breu i correcta.

Taula de creuament notes 1r lliurament / 2n lliurament

	2n lliurament				
1r lliurament	**A**	**B**	**C+**	**C-**	**D**
A	A	B	C+	C+	D
B	B	B	C+	C+	D
C+	B	B	C+	C-	D
C-	C+	C+	C+	C-	D
D	C-	C-	D	D	D

Pregunta 1: Map-Reduce - MongoDB

Partint del problema plantejat al primer lliurament i una llista de ciutats europees (amb les seves coordinades) es demana que implementeu el job que mitjançant un map reduce calculi el número d'avions que han sobrevolat cadascuna de les ciutats (considerarem una zona de 50Km de radi a partir de les coordinades donades).

La llista de ciutats per a realitzar el càlcul és la següent:

Ciutat	**Latitud**	**Longitud**
Amsterdam	52.371807	4.896029
Barcelona	41.390205	2.154007
Berlin	52.520008	13.404954
Dublin	53.350140	-6.266155
London	51.509865	-0.118092
Madrid	40.416775	-3.703790
Paris	48.864716	2.349014
Rome	41.906204	12.507516
Zurich	47.451542	8.564572

I l'objectiu és per cadascuna de les ciutats calcular el nombre d'avions diferents que les han sobrevolat amb les dades que tenim. També s'inclouen aquells avions que hagin aterrat a algun dels aeroports d'aquests ciutats, si estan dins del radi de 50Km des de les coordenades de la taula anterior.

Màquina virtual

Per tal de portar a terme la pràctica es proporciona una màquina virtual (Ubuntu Server 16.04 LTS 64 bits) preparada per Virtual Box. Trobareu un document junt amb aquest enunciat on s'explica com descarregar i utilitzar aquesta màquina virtual.

Joc de dades proporcionat

Juntament amb la màquina virtual es proporciona una bdd mongoDB ja carregada que conté aproximadament 2.000.000 de registres ADSB corresponents tots al continent europeu (latituds entre 36 i 61 nord, longituds entre 10 oest i 60 est) recollits mitjançant l'API streaming de ADSBExchange[1] durant unes hores el 17 de febrer de 2018.

La base de dades ocupa aproximdament 1Gb i conté una col·lecció anomenada «adsb» amb les dades ADS-B:

```
student@uoc:~$ mongo uoc
MongoDB shell version: 2.6.10
connecting to: uoc
> show dbs
admin  (empty)
```

1https://www.adsbexchange.com/data/#

```
local  0.078GB
uoc    0.953GB
> show collections
adsb
system.indexes
> db.adsb.stats()
{
       "ns" : "uoc.adsb",
       "count" : 1998323,
       "size" : 306795856,
       "avgObjSize" : 153,
       "storageSize" : 460894208,
       "numExtents" : 15,
       "nindexes" : 1,
       "lastExtentSize" : 124993536,
       "paddingFactor" : 1,
       "systemFlags" : 1,
       "userFlags" : 1,
       "totalIndexSize" : 64852032,
       "indexSizes" : {
              "_id_" : 64852032
       },
       "ok" : 1
}
> db.adsb.findOne()
{
       "_id" : ObjectId("5a8843e7d79a740f272ccc0a"),
       "Vsi" : -832,
       "Call" : "RYR8WE",
       "Spd" : 229,
       "Long" : 6.47644,
       "Trak" : 1.3,
       "Lat" : 51.530952,
       "GAlt" : 4582,
       "Sqk" : "2347",
       "Icao" : "4CA9C4",
       "ts" : ISODate("2018-02-17T16:01:59.173Z")
}
```

El significat dels camps[2] és el següent:

Camp	**Descripció**
`_id`	Identificador registre
`Vsi`	Velocitat vertical en peus per minut
`Call`	El «callsign» de l'avió.
`Spd`	Velocitat respecte terra en nusos
`Long`	Longitud
`Trak`	Direcció de l'avió (0 graus equival a nord)
`Lat`	Latitud
`GAlt`	Altitud en peus
`Sqk`	Codi «squack»
`Icao`	Codi ICAO (International Civil Aviation Organization) de l'avió
`ts`	Timestamp

2https://www.adsbexchange.com/datafields/

Càlcul del map Reduce

Per a fer el map reduce haureu de seguir les següents passes:

1. En el map s'ha de fer un emit per cada registre ADS-B que estigui dins del radi d'una ciutat. S'haurà d'indicar el codi ICAO de l'avió que l'ha generat per després poder comptar els avions. Aquest codi ICAO és únic per aeronau.
2. En el reduce s'han d'agrupar els avions que han sobrevolat cada ciutat evitant així els duplicats (un avió haurà emès molts de registres ADS-B mentre sobrevolava una ciutat).
3. En el reduce comptem els avions diferents que han sobrevolat cada ciutat i donem el resultat final.

Per a la realització del map reduce es proporciona el script mapReduce.js amb l'estructura del mapreduce ja preparada. A més ja s'ha inclòs un array amb la llista de ciutats i les seves coordinades geogràfiques i la funció que permet calcular la distància entre dos jocs de coordinades. El Script és el següent:

```
// Map function
var m = function() {
  // Distance between two coordinates in Km
  // http://www.movable-type.co.uk/scripts/latlong.html
  var d = function(lat1, lon1, lat2, lon2) {
    var R = 6371; // Radius of the earth in km
    var dLat = (lat2-lat1) * (Math.PI/180);
    var dLon = (lon2-lon1) * (Math.PI/180);
    var a = Math.sin(dLat/2) * Math.sin(dLat/2) +
            Math.cos(lat1 * (Math.PI/180)) * Math.cos(lat2 * (Math.PI/180)) *
            Math.sin(dLon/2) * Math.sin(dLon/2);
    var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
    return R * c; // Distance in km
  }
  // List of cities and coordinates
  var cities = [ [52.371807, 4.896029, "Amsterdam"],
                 [41.390205, 2.154007, "Barcelona"],
                 [52.520008, 13.404954, "Berlin"],
                 [53.350140, -6.266155, "Dublin"],
                 [51.509865, -0.118092, "London"],
                 [40.416775, -3.703790, "Madrid"],
                 [48.864716, 2.349014, "Paris"],
                 [41.906204, 12.507516, "Rome"],
                 [47.451542, 8.564572, "Zurich"] ];
  // TODO
};

// Reduce function
var r = function(key, values) {
  // TODO
};

// Finalize function
var f = function(key,value) {
  // TODO
}

db.runCommand( {
                 mapReduce: "adsb",
                 map: m,
                 reduce: r,
                 finalize: f,
                 out: {replace : "result"}
```

```
} );
```

Aquest script el podeu trobar al directori arrel de l'usuari student de la màquina virtual proporcionada. Heu d'implementar les tres funcions map, reduce i finalize, completant allà on s'indica un //TODO.

Exemple de mapReduce

Amb l'enunciat s'inclou un exemple de map reduce que calcula el número de missatges ADS-B segons el «flight level», és a dir, les franges d'alçada de 1000 peus en les que es divideix l'espai aeri. L'exemple el podreu trobar a la màquina virtual i adjunt a aquest enunciat.

A la funció map es fa un emit per cada flight level:

```
var m = function() {
  if (this.GAlt > 10000 && this.GAlt < 45000) {
    var fl = Math.round(this.GAlt / 1000) * 1000;
    emit (fl, 1);
  }
};
```

I a la funció reduce es compten els registres ADS-B per a cada flight level:

```
var r = function(key, values) {
  // print("reduce key=" + key + " values=" + values);
  return Array.sum(values);
};
```

Executem el map reduce:

```
student@uoc:~$ mongo uoc mapreduce-example.js
MongoDB shell version: 2.6.10
connecting to: uoc
```

I finalment podem veure el resultat:

```
student@uoc:~$ mongo uoc
MongoDB shell version: 2.6.10
connecting to: uoc
> db.result_example.find().sort({value:-1})
{ "_id" : 36000, "value" : 88474 }
{ "_id" : 35000, "value" : 76898 }
{ "_id" : 37000, "value" : 70560 }
{ "_id" : 34000, "value" : 67528 }
{ "_id" : 33000, "value" : 58301 }
{ "_id" : 11000, "value" : 55246 }
{ "_id" : 31000, "value" : 53113 }
{ "_id" : 38000, "value" : 50382 }
{ "_id" : 32000, "value" : 50211 }
{ "_id" : 30000, "value" : 47850 }
{ "_id" : 12000, "value" : 47794 }
{ "_id" : 29000, "value" : 47742 }
{ "_id" : 28000, "value" : 46906 }
{ "_id" : 27000, "value" : 45655 }
{ "_id" : 13000, "value" : 45237 }
{ "_id" : 24000, "value" : 45100 }
{ "_id" : 26000, "value" : 44739 }
{ "_id" : 25000, "value" : 44610 }
{ "_id" : 21000, "value" : 44088 }
{ "_id" : 15000, "value" : 43913 }
```

```
Type "it" for more
```

Material de consulta

La documentació de MongoDB és molt complerta i ha de ser el vostre referent alhora de realitzar la pràctica:

Ús de javascript a MongoDB:

- https://docs.mongodb.org/manual/core/server-side-javascript/

MongoDB tutorial (Getting Started)

- https://docs.mongodb.org/getting-started/shell/

Explicació Map Reduce:

- https://docs.mongodb.org/manual/core/map-reduce/
- https://docs.mongodb.com/manual/reference/command/mapReduce/
  Reviseu les seccions: “Requirements for the map function”, “Requirements for the reduce function” i “Requirements for the finalize function”
- https://docs.mongodb.org/manual/tutorial/troubleshoot-map-function/
- https://docs.mongodb.org/manual/tutorial/troubleshoot-reduce-function/

Exemples de Map Reduce

- https://docs.mongodb.org/manual/tutorial/map-reduce-examples/

Pregunta 2: Cassandra

Suposant que al final la eina escollida fos Cassandra, argumenta quin hauria de ser el nivell de consistència (Consistency level) en els següents casos:

- La escriptura de dades ADS-B recollides per la xarxa d'estacions.
- La lectura de dades al realitzar processos de map-reduce com el plantejat a la pregunta 1

Infrastructura disponible

Suposem que hem utilitzat Cassandra per a resoldre el problema plantejat al primer lliurament amb una configuració de 9 màquines distribuïdes en tres centres de càlcul utilitzant una estratègia NetworkTopologyStrategy:

- 3 màquines al centre de càlcul 1 – factor de replicació 3
- 3 màquines al centre de càlcul 2 – factor de replicació 3
- 3 màquines al centre de càlcul 3 – factor de replicació 3

A cada centre de càlcul els tres servidors es consideraran a racks diferents.

La consistència "ajustable" de Cassandra

El factor de replicació (replication factor) és el numero de vegades que cada dada es replica en un centre de càlcul (datacenter). El nivell de consistència sempre es defineix en base a aquest número, que ha de ser menor o igual que el número de màquines disponibles.

Cassandra disposa de consistència "ajustable". És a dir, permet establir nivells diferents de consistència a cadascuna de les operacions de forma diferent. Això equival a que l'usuari pot escollir quantes màquines del cluster contestaran a un determinat SELECT o qualsevol altra instrucció enviada a la base de dades.

Als links del proper apartat podreu trobar els diferents nivells de consistència possibles explicats.

Material de consulta

Datastax és la principal empresa que hi ha darrera Cassandra i disposa de molta documentació online. Recomanem les següents planes per a resoldre aquesta pregunta:

- https://docs.datastax.com/en/cassandra/3.0/cassandra/dml/dmlAboutDataConsistency.html
- https://docs.datastax.com/en/cassandra/3.0/cassandra/dml/dmlConfigConsistency.html
- http://www.slideshare.net/DataStax/understanding-data-consistency-in-apache-cassandra