Criterios de evaluación y condiciones de la entrega

Fecha límite de entrega: 26-11-2020.

Hay que entregar la práctica en esta fecha en el buzón Entrega y registro de AC de vuestra aula.

A la hora de presentarla, el aplicativo la renombrará automáticamente a partir de vuestro nombre y apellidos, denominación de la práctica y fecha de entrega. Comprobad que el nombre asignado es correcto.

Es necesario guardar la solución en un **fichero .ZIP** que contenga únicamente los siguientes dos ficheros:

1. **Respuestas a la práctica en formato PDF**. Hay que ser preciso y claro en las respuestas.
2. **Fichero mapReduce.js** completado, en caso de responder el tercer bloque.

Asimismo os recordamos que la práctica es un trabajo **estrictamente individual.**

En el enunciado hay 6 preguntas agrupadas en dos bloques. Primer bloque 1,2,3, y 4, y segundo bloque preguntas 5 y 6. No es necesario responder a todas las preguntas. Se seguirá el siguiente criterio:

Para optar a una A tenemos dos opciones:

- 1 bloque (1,2 o 4), y el bloque 3

Para optar a una B:

- bloque 3
- bloque 1 + bloque 2 + bloque 4

Y para optar a una C+:

- 2 bloques (1,2 o 4)

En la evaluación se dará importancia a que la argumentación (cuando se pida) sea breve y correcta.

Descripción del problema

Detección de tendencias mediante análisis de tweets

Tenemos el encargo de montar un sistema capaz de analizar tendencias, impactos de campañas publicitarias, repercusión de determinados eventos deportivos, etc. a partir de los datos de Twitter. Se trata de capturar durante periodos cortos de tiempo (p.e. una semana) todos los tweets o bien de una zona geográfica o bién con unos temas concretos, para a continuación analizar esta información de forma offline mediante procesos map-reduce.

Volumetría de Twitter

La volumetría de Twitter es muy grande. Se estima que de media se generan unos 6000 tweets por segundo[1] (500 millones de tweets al día / 200.000 millones de tweets al año). El récord de tweets por segundo está actualmente en 143199 tweets por segundo[2].

Twitter ofrece APIs[3] que nos permiten capturar los tweets que hacen referencia a una determinada zona geográfica[4] o lista de palabras clave[5].

Notad que en esta práctica nos centramos sólo en el almacenamiento de los tweets y no vamos a considerar otros aspectos que también serian necesarios como por ejemplo como consumir las APIs de Twitter.

Volumetría de nuestro sistema

Calculamos que en cada estudio sólo capturaremos como máximo un 5% del tráfico total de Twitter y a demás esta captura durará como mucho una semana. Por lo tanto estamos considerando una media de 300 tweets por segundo y un total por evento de 175 millones de tweets a analizar.

Datos almacenados

Para cada tweet almacenaremos el json que se recibe del API de Twiter[6], por ejemplo:


```
{
"_id" : ObjectId("5f5894331499e45a1a74324d"),
"created_at" : "Wed Sep 09 08:37:02 +0000 2020",
"id" : NumberLong("1303613369925148673"),
"id_str" : "1303613369925148673",
"text" : "Són dos quarts i mig d'onze del matí",
"source" : "<a href=\"http://www.olesa.cat/horacatalana\"
rel=\"nofollow\">Hora Catalana</a>",
"truncated" : false,
"in_reply_to_status_id" : null,
"in_reply_to_status_id_str" : null,
"in_reply_to_user_id" : null,
"in_reply_to_user_id_str" : null,
"in_reply_to_screen_name" : null,
"user" : {
"id" : 1384110594,
"id_str" : "1384110594",
```


1 http://www.internetlivestats.com/twitter-statistics/
2 https://blog.twitter.com/2013/new-tweets-per-second-record-and-how
3 https://dev.twitter.com/streaming/reference/post/statuses/filter
4 https://dev.twitter.com/streaming/overview/request-parameters#locations
5 https://dev.twitter.com/streaming/overview/request-parameters#track
6 https://dev.twitter.com/overview/api/tweets

```
"name" : "L'hora catalana",
"screen_name" : "HoraCatalana",
"location" : "Olesa de Montserrat",
"url" : "http://ca.wikipedia.org/wiki/Sistema_horari_catal%C3%A0",
"description" : "Ara és l'hora, catalans: aprèn a dir l'hora en català des del
teu TL! · Sóc el community manager de @TorreRellotge · [des del 27/abr/2013
16:05]",
"translator_type" : "none",
"protected" : false,
"verified" : false,
"followers_count" : 6512,
"friends_count" : 34,
"listed_count" : 87,
"favourites_count" : 139,
"statuses_count" : 1030540,
"created_at" : "Sat Apr 27 10:37:30 +0000 2013",
"utc_offset" : null,
"time_zone" : null,
"geo_enabled" : true,
"lang" : null,
"contributors_enabled" : false,
"is_translator" : false,
"profile_background_color" : "131516",
"profile_background_image_url" :
"http://abs.twimg.com/images/themes/theme14/bg.gif",
"profile_background_image_url_https" :
"https://abs.twimg.com/images/themes/theme14/bg.gif",
"profile_background_tile" : true,
"profile_link_color" : "FF6600",
"profile_sidebar_border_color" : "FFFFFF",
"profile_sidebar_fill_color" : "DDEEF6",
"profile_text_color" : "333333",
"profile_use_background_image" : true,
"profile_image_url" :
"http://pbs.twimg.com/profile_images/1064920022143942658/cm5VvuHN_normal.jpg",
"profile_image_url_https" :
"https://pbs.twimg.com/profile_images/1064920022143942658/cm5VvuHN_normal.jpg"
,
"profile_banner_url" :
"https://pbs.twimg.com/profile_banners/1384110594/1398008656",
"default_profile" : false,
"default_profile_image" : false,
"following" : null,
"follow_request_sent" : null,
"notifications" : null
},
"geo" : {
"type" : "Point",
"coordinates" : [41.545639,1.893817]},
"coordinates" : {
"type" : "Point",
"coordinates" : [1.893817,41.545639]
},
"place" : {
"id" : "f91c3faa315d62e6",
"url" : "https://api.twitter.com/1.1/geo/id/f91c3faa315d62e6.json",
"place_type" : "city",
"name" : "Olesa de Montserrat",
"full_name" : "Olesa de Montserrat, Espanya",
"country_code" : "ES",
"country" : "Espanya",
"bounding_box" : {
"type" : "Polygon",
```

```
"coordinates" : [
[[1.872591,41.532654],
[1.872591,41.579219],
[1.933297,41.579219],
[1.933297,41.532654]]]
},
"attributes" : {}
},
"contributors" : null,
"is_quote_status" : false,
"quote_count" : 0,
"reply_count" : 0,
"retweet_count" : 0,
"favorite_count" : 0,
"entities" : {
"hashtags" : [ ],
"urls" : [ ],
"user_mentions" : [ ],
"symbols" : [ ]
},
"favorited" : false,
"retweeted" : false,
"filter_level" : "low",
"lang" : "ca",
"timestamp_ms" : "1599640622012"
}
```


Resumen

El objetivo de la práctica es analizar los requerimientos de nuestro sistema y razonar sobre cuál es la mejor herramienta noSQL que podemos utilizar. Además de ajustarnos a los requerimientos expresados anteriormente, también se nos pide que en un futuro la solución escogida pueda crecer para asumir estudios más amplios (p.e. de más duración) o para poder hacer varios estudios simultáneos.

Bloque 1

En este primer bloque de preguntas se os pide que analicéis el problema presentado desde el punto de vista del almacenamiento de la información utilizando un sistema noSQL.

Después de un exhaustivo estudio se ha reducido la lista de sistemas candidatos a tres:

- Cassandra
- Couchbase
- MongoDB

Responded a las siguientes preguntas (las respuestas deben ser breves y concisas, máximo 10 líneas):

1. Cassandra se considera un sistema noSQL columnar con replicación *materless*, Couchbase es un sistema noSQL documental con replicación *multi-master y* MongoDB es un sistema noSQL documental *master-slave.*

   Des del punto de vista de la alta disponibilidad de escritura (write throughput scalability), ¿cuál es la mejor estrategia (masterless / multi-master / master-slave), ¿por qué?

2. Para poder implementar correctamente nuestro sistema necesitamos distribución geográfica. De los tres sistemas anteriores escoge uno y sólo uno (no hace falta justificar la elección) y razona como lo harías para conseguir la distribución geográfica e indicando de entre las diferentes características que proporciona el sistema cuáles necesitas para poder implementar la distribución geográfica.

3. Define en qué consiste decir que el sistema prima la consistencia de los datos (frente a la disponibilidad o la tolerancia a particiones). ¿Crees que la consistencia de los datos (entendida como consistencia en el teorema CAP) es importante en nuestro caso?

Material de consulta

A continuación podéis encontrar referencias a documentación de soporte para esta parte. De entrada dos referencias muy básicas que nos serviran como introducción:

- http://en.wikipedia.org/wiki/Distributed_data_store
- http://en.wikipedia.org/wiki/CAP_theorem

Una descripción de las herramientas NoSQL disponibles con una clasificación:

- http://en.wikipedia.org/wiki/NoSQL

Un artículo sobre NoSQL de la documentación de Mongo (muy completa) i que explica muy bien las diferencias entre sistemas NoSQL i los sistemas SQL relacionales tradicionales:

- http://www.mongodb.com/nosql-explained

Data management in cloud environments: NoSQL and NewSQL data stores Grolinger et al. Journal of Cloud Computing: Advances, Systems and Applications 2013, 2:22

- http://www.journalofcloudcomputing.com/content/2/1/22
  (podéis bajaros un PDF)

Documentación de Cassandra

- https://cassandra.apache.org/doc/latest/

Documentación de MongoDB

- http://docs.mongodb.org/manual/

Documentación de Couchbase

- https://developer.couchbase.com/documentation/server/current/introduction/intro.html

Bloque 2: Preguntas Cassandra

En este segundo bloc de preguntas supondremos que finalmente hemos escogido Cassandra.

Dado que Cassandra proporciona consistencia ajustable, ahora tenemos que decidir cual es la mejor estrategia de consistencia para cada uno de los diferentes usos:

Responded a las siguientes preguntas (las respuestas tienen que ser breves, máximo 10 líneas):

1. ¿Cuál es el nivel de consistencia que deberíamos utilizar al escribir los jsons de twitter en Cassandra? ¿Por qué?
2. ¿Cuál es el nivel de consistencia más adecuado para realizar las lecturas correspondientes a los procesos de cálculo que se desean realizar? ¿Por qué?
3. Si tenemos un centro de datos con 6 nodos y un replication factor de 3, y un segundo centro de datos con 12 nodos y un replication factor de 5, cuál es el número mínimo de nodos que han de confirmar antes de dar por buena una operación con nivel de consistencia QUORUM? Y con LOCAL_QUORUM? Razona brevemente el porqué.

Material de consulta

Recomendamos las siguientes páginas para resolver este bloque de preguntas:

- https://docs.datastax.com/en/cassandra-oss/3.x/cassandra/dml/dmlAboutDataConsistency.html
- https://docs.datastax.com/en/cassandra-oss/3.x/cassandra/dml/dmlConfigConsistency.html
- http://www.slideshare.net/DataStax/understanding-data-consistency-in-apache-cassandra

Notad que Datastax es la principal empresa que está detrás de Cassandra y dispone de mucha documentación online que también os será útil además de la indicada anteriormente.

Bloque 3: Ejercicio MapReduce - MongoDB

Partiendo del problema planteado anteriormente se os pide que implementéis el job que mediante un map reduce calcule para cada país de origen de tweets los siguientes datos:

- Número de tweets originados en el país
- Las 10 palabras de más de tres letras más utilizadas
- Los 5 usuarios que han enviado más tweets

Máquina virtual

Para llevar a cabo la práctica se proporciona una máquina virtual (Alpine Linux 3.9 64 bits) preparada per Virtual Box. Encontraréis un documento junto a este enunciado dónde se explica cómo descargar y utilizar esta máquina virtual.

Juego de datos proporcionado

Para resolver la práctica se os proporciona una máquina virtual preparada para ejecutarse en Virtual Box con una BDD MongoDB cargada con aproximadamente 50.000 tweets.

Estos tweets han sido recogidos aproximadamente durante unas horas en el mes de septiembre mediante la API streaming de twitter[7] y corresponden a tweets originados en la Península Ibérica.

La base de datos ocupa aproximadamente 80Mb:

```
uoc-sdge:~# mongo uoc
MongoDB shell version v4.0.5
connecting to: mongodb://127.0.0.1:27017/uoc?gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("6e43fb9b-6b0f-4b7e-a737-
2bf012ec269a") }
MongoDB server version: 4.0.5
[...]
> show dbs
admin   0.000GB
config  0.000GB
local   0.000GB
uoc     0.083GB
> show collections
tweets
> db.tweets.count()
53146
```

Los tweets recogidos en la base de datos siguen el formato JSON especificado por el API de Twitter[8]. Por ejemplo:

```
> db.tweets.findOne()
{
"_id" : ObjectId("5f5894331499e45a1a74324d"),
"created_at" : "Wed Sep 09 08:37:02 +0000 2020",
"id" : NumberLong("1303613369925148673"),
"id_str" : "1303613369925148673",
"text" : "Són dos quarts i mig d'onze del matí",
"source" : "<a href=\"http://www.olesa.cat/horacatalana\"
rel=\"nofollow\">Hora Catalana</a>",
```

7 https://dev.twitter.com/streaming/reference/post/statuses/filter
8 https://dev.twitter.com/overview/api/tweets

```
"truncated" : false,
"in_reply_to_status_id" : null,
"in_reply_to_status_id_str" : null,
"in_reply_to_user_id" : null,
"in_reply_to_user_id_str" : null,
"in_reply_to_screen_name" : null,
"user" : {
"id" : 1384110594,
"id_str" : "1384110594",
"name" : "L'hora catalana",
"screen_name" : "HoraCatalana",
"location" : "Olesa de Montserrat",
"url" : "http://ca.wikipedia.org/wiki/Sistema_horari_catal%C3%A0",
"description" : "Ara és l'hora, catalans: aprèn a dir l'hora en català des del
teu TL! · Sóc el community manager de @TorreRellotge · [des del 27/abr/2013
16:05]",
"translator_type" : "none",
"protected" : false,
"verified" : false,
"followers_count" : 6512,
"friends_count" : 34,
"listed_count" : 87,
"favourites_count" : 139,
"statuses_count" : 1030540,
"created_at" : "Sat Apr 27 10:37:30 +0000 2013",
"utc_offset" : null,
"time_zone" : null,
"geo_enabled" : true,
"lang" : null,
"contributors_enabled" : false,
"is_translator" : false,
"profile_background_color" : "131516",
"profile_background_image_url" :
"http://abs.twimg.com/images/themes/theme14/bg.gif",
"profile_background_image_url_https" :
"https://abs.twimg.com/images/themes/theme14/bg.gif",
"profile_background_tile" : true,
"profile_link_color" : "FF6600",
"profile_sidebar_border_color" : "FFFFFF",
"profile_sidebar_fill_color" : "DDEEF6",
"profile_text_color" : "333333",
"profile_use_background_image" : true,
"profile_image_url" :
"http://pbs.twimg.com/profile_images/1064920022143942658/cm5VvuHN_normal.jpg",
"profile_image_url_https" :
"https://pbs.twimg.com/profile_images/1064920022143942658/cm5VvuHN_normal.jpg"
,
"profile_banner_url" :
"https://pbs.twimg.com/profile_banners/1384110594/1398008656",
"default_profile" : false,
"default_profile_image" : false,
"following" : null,
"follow_request_sent" : null,
"notifications" : null
},
"geo" : {
"type" : "Point",
"coordinates" : [41.545639,1.893817]},
"coordinates" : {
"type" : "Point",
"coordinates" : [1.893817,41.545639]
},
"place" : {
```

```
"id" : "f91c3faa315d62e6",
"url" : "https://api.twitter.com/1.1/geo/id/f91c3faa315d62e6.json",
"place_type" : "city",
"name" : "Olesa de Montserrat",
"full_name" : "Olesa de Montserrat, Espanya",
"country_code" : "ES",
"country" : "Espanya",
"bounding_box" : {
"type" : "Polygon",
"coordinates" : [
[[1.872591,41.532654],
[1.872591,41.579219],
[1.933297,41.579219],
[1.933297,41.532654]]]
},
"attributes" : {}
},
"contributors" : null,
"is_quote_status" : false,
"quote_count" : 0,
"reply_count" : 0,
"retweet_count" : 0,
"favorite_count" : 0,
"entities" : {
"hashtags" : [ ],
"urls" : [ ],
"user_mentions" : [ ],
"symbols" : [ ]
},
"favorited" : false,
"retweeted" : false,
"filter_level" : "low",
"lang" : "ca",
"timestamp_ms" : "1599640622012"
}
```


Càlculo del map Reduce

Para realizar el cálculo del map reduce utilizaremos los siguientes campos:

- país: country_code
- usuarios: user.screen_name
- palabras: text

En el caso de las palabras seleccionaremos aquellas palabras con más de 3 caracteres, que empiezan por alguna letra y no empiezan por http. Más adelante se mostrará cómo obtener las palabras en uno de los ejemplos proporcionados.

El resultado esperado para cada uno de los países es el siguiente (en este caso Andorra):

```
  {
    "_id" : "AD",
    "value" : {
      "count" : 102,
      "words" : [
        "bien",
        "andorra",
        "people",
        "when",
        "este",
        "there",
```

```
        "tinc",
        "with",
        "make",
        "c'est"
      ],
      "users" : [
        "InesRG86",
        "Tommy_M_E_J",
        "laurentbourelly",
        "albertllovera",
        "facusantana"
      ]
    }
  }
```

Para la realización de la práctica se proporciona el script mapReduce.js con la estructura del MapReduce ya preparada. El script lo encontraréis en el directorio raíz del usuario student de la máquina virtual proporcionada.

Debéis implementar las tres funciones map, reduce y finalize, indicadas mediante un comentario // TODO.

Ejemplos de mapReduce

Con el enunciado proporcionamos 3 ejemplos de MapReduce. Los tres archivos de ejemplo los encontraréis también en el directorio raíz del usuario student de la máquina virtual proporcionada.

Ejemplo 1:

Suponed que queremos conseguir el número de tweets por cada país. Para ello programamos una función de mapping m, que por cada elemento de la colección genera una pareja clave-valor donde la clave es el código del país y el valor es un 1 (número entero):

```
var m = function() {
  if (this.place != null) {
    emit (this.place.country_code, 1);
  } else {
    emit ("?", 1);
  }
};
```

A continuación debemos programar la función de agregación, que recibirá un array de valores (que inicialmente serán números enteros 1 del emit) y deberá devolver un único valor, también entero, que será la suma de todos ellos. De esta forma podremos contar el número de tweets que se han hecho en cada país (la clave es el código de país):

```
var r = function(key, values) {
  return Array.sum(values);
};
```

Finalmente ejecutamos el mapReduce:

```
db.runCommand( {
                  mapReduce: "tweets",
                  map: m,
                  reduce: r,
                  out: {replace : "result_country"}
```

```
                } );
```

Y el resultado es el siguiente:

```
  > db.result_country.find()
  { "_id" : "", "value" : 2 }
  { "_id" : "?", "value" : 4 }
  { "_id" : "AD", "value" : 102 }
  { "_id" : "DZ", "value" : 239 }
  { "_id" : "ES", "value" : 45538 }
  { "_id" : "FR", "value" : 1567 }
  { "_id" : "GI", "value" : 105 }
  { "_id" : "MA", "value" : 8 }
  { "_id" : "PT", "value" : 5581 }
```

Ejemplo 2:

En el segundo ejemplo contaremos las ocurrencias de cada palabra. Para ello haremos un emit por cada palabra. Para obtener las palabras seguiremos el mismo criterio que se pide en la práctica. Primero dividiremos el texto del tweet en palabras y después filtraremos las que tienen más de 3 letras, empiezan por letra y no empiezan por http:

```
var m = function() {
  this.text.match(/\S+/g).forEach(function(word) {
    if ( word.length > 3
         && ! word.match(/^http/)
         && word.match(/^[a-zA-Z]/) ) {
      emit(word,1)
    }
  })
};
```

El resto del map reduce es similar al ejemplo anterior.

El resultado es el siguiente:

```
 > db.result_words.find().sort({value:-1})
  { "_id" : "para", "value" : 2911 }
  { "_id" : "pero", "value" : 1984 }
  { "_id" : "como", "value" : 1628 }
  { "_id" : "este", "value" : 1125 }
  { "_id" : "está", "value" : 909 }
  { "_id" : "esta", "value" : 888 }
  { "_id" : "cuando", "value" : 855 }
  { "_id" : "todo", "value" : 817 }
  { "_id" : "porque", "value" : 729 }
  { "_id" : "tiene", "value" : 684 }
  { "_id" : "gracias", "value" : 671 }
  { "_id" : "pues", "value" : 661 }
  { "_id" : "todos", "value" : 580 }
  { "_id" : "tengo", "value" : 575 }
  { "_id" : "estoy", "value" : 565 }
  { "_id" : "bien", "value" : 550 }
  { "_id" : "esto", "value" : 549 }
  { "_id" : "mejor", "value" : 532 }
  { "_id" : "hace", "value" : 517 }
  { "_id" : "solo", "value" : 511 }
  Type "it" for more
```

Corrección automática

La entrega del bloque 3 se realiza mediante la herramienta de corrección automática que podéis encontrar en la dirección https:/sd.uoc.edu/dslab.

Para llevar a cabo la corrección automática se deben seguir los siguientes pasos:

Preparar fitcheros map.js, reduce.js i finalize.js

Una vez que tengáis el map reduce funcionando en vuestro entorno local utilizando la máquina virtual proporcionada debéis preparar los tres archivos necesarios para la corrección automática. Cada uno de ellos tendrá la definición correspondiente de una de las tres funciones utilizadas en el map reduce:

map.js

```
function() {
  // TODO
};
```

reduce.js

```
function(key, values) {
  // TODO
};
```

finalize.js.

```
function(key, val) {
  // TODO
};
```

Creación de un proyecto

Una vez tengamos estos ficheros podemos acceder al aplicativo de corrección automática (https:/sd.uoc.edu/dslab) utilizando vuestro usuario de la UOC y crear un proyecto nuevo (Project -- New Project) adjuntando los tres ficheros map.js, reduce.js y finalize.js.

Enviar a corregir

A continuación enviamos la práctica a corregir (Submision -- New Submission). La corrección dura aproximadamente 10 minutos y al finalizar os indicará si el resultado es correcto o no.

Material de consulta

La documentación de MongoDB es muy completa y debería ser vuestro referente al realizar la práctica:

Uso de javascript en MongoDB:

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

MongoDB tutorial (Getting Started)

- https://docs.mongodb.com/manual/tutorial/getting-started/

Explicación Map Reduce:

- https://docs.mongodb.org/manual/core/map-reduce/

- https://docs.mongodb.com/manual/reference/command/mapReduce/
  Revisad las secciones: “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/

Ejemplos de Map Reduce

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

Bloque 4: Preguntas Map Reduce

Responded a las siguientes preguntas relacionadas con el algoritmo del map reduce:

1. Razona brevemente porqué hace falta la función finalize, es decir ¿Por qué No podemos hacer el cálculo directamente en la función reduce?
2. Supongamos que queremos reducir el periodo de tiempo sobre el que se ejecuta el map reduce. ¿Cuál es la forma más adecuada y eficiente? ¿Por qué?
   a) Añadiendo un if en el código de la función map
   b) En la función reduce
   c) Con el parámetro query al llamar al map reduce
3. ¿Cuál es la propiedad de la función reduce que permite a un algoritme map reduce ejecutarse en paralelo en muchas máquinas a la vez, es decir de forma distribuida?

Las respuestas deben estar bien razonadas y ser breves y concisas (máximo 15 lineas)