Parcialito 4

Alumno: Alan Ezequiel Valdevenito.
Padrón: 107585.

Consultas de MongoDB

Ejercicio 1

Resolución:

```
collection.find(
  {
    "favorite_count": {$gte: 50},
    "created_at.date": {$regex: "T15:"}
  },
  {
    "_id": 1,
    "entities.hashtags": {$size: "$entities.hashtags"},
    "favorite_count": 1
  }
).sort({"favorite_count": -1})
```

Salida resultante en la consola de MongoDB Compass (solo se muestran los primeros 4 resultados):

```
< {
    _id: '1143898673539162112',
    entities: {
      hashtags: 1
    },
    favorite_count: 1269
  }
  {
    _id: '1143899877459542017',
    entities: {
      hashtags: 1
    },
    favorite_count: 1006
  }
  {
    _id: '1143909533661782017',
    entities: {
      hashtags: 0
    },
    favorite_count: 328
  }
  {
    _id: '1143904304367120384',
    entities: {
      hashtags: 0
    },
    favorite_count: 299
  }
```

Explicación: Utilizamos

```
db.collection.find(<query>, <projection>).sort({fieldName: order})
```

En el campo <query> especificamos el criterio de selección de los documentos que queremos obtener:

- “favorite_count”: {$gte: 50} → Este par clave-valor especifica que buscamos aquellos tweets que tengan 50 o más likes.
- “created_at.date”: {$regex: “T15:”} → Este par clave-valor especifica que buscamos aquellos tweets que hayan sido a las 3 de la tarde.

En el campo <projection> especificamos los campos que deben incluirse en el documento que queremos obtener. En nuestro caso buscamos obtener un documento que incluya los ids, cantidad de hashtags y likes.

Por último, ordenamos la salida de forma descendente por cantidad de likes.

Ejercicio 2

Resolución:

```
collection.aggregate([
  {
    $match: {
      $expr: {$gt: [{$size: "$entities.hashtags"}, 3]}
    }
  },
  {
    $unwind: "$entities.hashtags"
  },
  {
    $group: {
      _id: "$entities.hashtags.text",
      usuarios: {$push: "$user_id"},
      maximo: {$max: "$retweet_count"},
      minimo: {$min: "$retweet_count"},
      promedio: {$avg: "$retweet_count"}
    }
  }
])
```

Salida resultante en la consola de MongoDB Compass (solo se muestran los primeros 3 resultados):

```
< {
    _id: 'BienDeMañana',
    usuarios: [
      '1031885855990996995'
    ],
    maximo: 0,
    minimo: 0,
    promedio: 0
  }
  {
    _id: 'solteros',
    usuarios: [
      '1105871843037204480'
    ],
    maximo: 0,
    minimo: 0,
    promedio: 0
  }
  {
    _id: 'CiudadVerde',
    usuarios: [
      '794573155046162433'
    ],
    maximo: 0,
    minimo: 0,
    promedio: 0
  }
```

Explicación: Se utiliza

```
db.collection.aggregate([
          {$stage1},
          {$stage2},
          ...
          {$stageN},
])
```

Notemos que en nuestra consulta tenemos 3 stages:

1) $match → En este stage filtramos los documentos tales que sean tweets que utilicen más de 3 hashtags. Usamos $expr para realizar una expresión de comparación con $gt. Básicamente comparamos el tamaño del array de hashtags con 3 para seleccionar solo aquellos que tienen más de 3 hashtags.
2) $unwind → En este stage descomponemos los arrays de hashtags en sus elementos individuales.
3) $group → En este stage agrupamos por hashtag. Luego, para cada hashtag obtenemos una lista con los usuarios que lo utilizaron y el máximo, mínimo y promedio de retweets.

Ejercicio 3

Se utiliza

```
db.collection.aggregate([
          {$stage1},
          {$stage2},
          ...
          {$stageN},
])
```

Notemos que en nuestra consulta tenemos 3 stages:

1) $match → En este stage se filtran los documentos tales que sean tweets en idioma español (es) o portugal (pt) y que se hayan hecho desde Brasil.

2) $group → En este stage se agrupa por el campo 'in_reply_to_status_id_str' y en caso de ser nulo entonces agrupa por el campo '_id'. Notemos que el campo 'in_reply_to_status_id_str' nos dice el id del tweet al que se está respondiendo en caso de que sea un tweet respuesta. Luego, para cada agrupamiento obtenemos un array de nombre tweets (contiene el id, el texto, el usuario y la fecha de creación) y el promedio de retweets.

3) $project → En este stage se especifican los campos que deben incluirse en el documento que se quiere obtener. Notemos que se puede incluir, excluir y modificar campos existentes, así como crear nuevos:

- tweet → Este campo se proyecta usando una combinación de $arrayElemAt y $filter. Básicamente se filtran los elementos del array tweets que cumplen con la condición de tener el mismo valor para id y tweet_id. Es decir, filtra y se queda con el tweet principal.
- replies → Este campo se proyecta usando una combinación de $sortArray y $filter. Básicamente se filtran los elementos del array tweets que cumplen con la condición de tener un distinto valor para id y tweet_id. Es decir, filtra y se queda con las respuestas. Luego, se ordena el array resultante de forma ascendente por fecha de creación.
- avg_retweets → Este campo se proyecta tal y como esta. Es decir, se mantiene igual al obtenido en el stage anterior.

Esta query filtra los tweets en español y portugués originados en Brasil, agrupa los tweets por conversación (identificando el tweet principal y sus respuestas), y luego proyecta un documento que contiene:

- tweet: El tweet principal de la conversación.
- replies: Las respuestas al tweet principal, ordenadas cronológicamente.
- avg_retweets: El promedio de retweets para los tweets en la conversación.

Esta query es útil para obtener un documento cuya estructura tenga al tweet principal y sus respuestas de forma organizada.

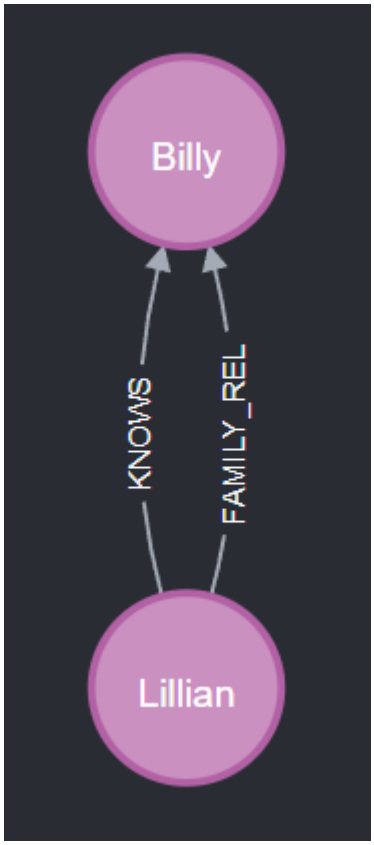
Consultas de Neo4j

Ejercicio 4

Resolución:

```
1 MATCH (l:Location)-[:OCCURRED_AT]-(c:Crime)-[:PARTY_TO]-(p:Person)
2 WHERE l.address = '165 Laurel Street'
3 RETURN p
```

Resultado:



Ejercicio 5

Resolución:

```
1 MATCH (p:Person)-[:HAS_PHONE]→(ph:Phone)-[]-(pc:PhoneCall)
2 WITH p, COUNT(pc) AS cantidad_llamadas
3 WHERE cantidad_llamadas > 7
4 RETURN p.name AS nombre, p.surname AS apellido, cantidad_llamadas
```

Resultado:

p.name	p.surname	cantidad_llamadas
"Benjamin"	"Hamilton"	8
"Todd"	"Garcia"	8
"Louis"	"Parker"	8
"Anne"	"Nguyen"	8
"Bobby"	"Thompson"	8
"Paul"	"Arnold"	8
"Janice"	"Coleman"	8
"Ann"	"Fox"	8
"Wanda"	"Weaver"	8
"Bruce"	"Baker"	8
"Angela"	"Mccoy"	8
"Jacqueline"	"Holmes"	8
"Judith"	"Fernandez"	8
"Harold"	"Robertson"	8
"Frances"	"Sullivan"	8
"Dorothy"	"Hall"	10
"Kimberly"	"Wood"	10
"Jerry"	"Johnston"	8
"Brandon"	"Martin"	8
"Kathy"	"Wheeler"	8
"Phillip"	"Perry"	8

"Jack"	"Reyes"	8
"Andrew"	"Foster"	8
"Craig"	"Marshall"	8
"Joan"	"Shaw"	8
"Janet"	"Cunningham"	8
"Henry"	"Jacobs"	8
"Patricia"	"Wheeler"	8
"Kenneth"	"Fernandez"	8
"Jessica"	"Kelly"	8