1. Obtener los ids, cantidad de hashtags y likes para tweets que tengan 50 o más likes (‘favorite_count’) y hayan sido a las 3 de la tarde. Ordenar la salida de forma descendente por cantidad de likes.
Utilice una única consulta básica con `find(<query>, <proyeccion>).sort({}).limit({})`

```
# ejercicio 1: Consulta

tweets_1 = list(collection.find({
"favorite_count": {"$gte": 50},
"created_at.date": {"$regex": ".*T15:.*"} # Para tweets a las 3 de la tarde
}, {
"_id": 1,
"favorite_count": 1,
"entities.hashtags": 1
}).sort("favorite_count", -1))

# resultados
print("Ejercicio 1:", tweets_1)
```

2. Para cada hashtag obtener los usuarios que lo utilizaron además del máximo, mínimo y promedio de retweets, sólo teniendo en cuenta aquellos tweets que utilicen más de 3 hashtags (primero se deben filtrar los tweets y luego hallar los valores por cada hashtag).
Se debe utilizar el pipeline de agregación.

```
result_2 = list(collection.aggregate([
{"$match": {
"$expr":{
"$gte": [
{
"$size":"$entities.hashtags"
},
3
]
}
}},
{"$unwind": "$entities.hashtags"},
{"$group": {
"_id": "$entities.hashtags.text",
"users": {"$addToSet": "$user.name"},
"max_retweets": {"$max": "$retweet_count"},
"min_retweets": {"$min": "$retweet_count"},
"avg_retweets": {"$avg": "$retweet_count"}
}}
]))

#resultados
print("Ej 2:", result_2)
```

3. Dada la consulta: Anexo: Consulta ejercicio 3 o disponible en github
Explicar qué sucede en cada paso del pipeline y en forma resumida qué resuelve la query completa.

- $match:

  Filtra los documentos tal que solo se incluyan los tweets en español (es) o portugués (pt) y que además el país sea Brasil (place.country = “Brasil”)

- $group:

  Agrupa los documentos según el campo in_reply_to_status_id_str si es que existe. Si no, usa el campo id.

  Luego, crea dos nuevos campos:
  - ✓ Tweets: array que almacena información de cada tweet (id, texto, usuario, fecha_creacion)
  - ✓ Languages: conjunto ($addToSet) que almacena los conteos de retweets (retweet_count).

- $project:

  tweet: selecciona el tweet original de entre los agrupados.
  - ✓ Usa $filter para buscar en el array tweets el tweet cuyo tweet_id coincide con id.
  - ✓ Usa $arrayElemAt con el índice 0 para seleccionar el primer (y único) elemento del array filtrado.

  replies: Un array de tweets que son respuestas al tweet original, ordenados por la fecha de creación (created_at).

  - ✓ Usa $filter para obtener tweets en tweets que no son el tweet original ($ne).
  - ✓ Usa $sortArray para ordenar estos tweets por created_at.

  avg_retweets: Incluye el campo avg_retweets calculado en el paso de agrupación.

Resumen de la Consulta:

La consulta completa selecciona tweets en español o portugués que provienen de Brasil, luego agrupa los tweets por respuesta o tweet original, y muestra los resultados en una estructura que incluye el tweet original, las respuestas al tweet original ordenadas por fecha de creación y el promedio de retweets del grupo.

Resuelva las siguientes consultas de Neo4j utilizando la base "Crime Investigation" vista en clase (*Taller VIII: Neo4j*).

4. Investigue los crímenes cometidos en 165 Laurel Street, muestre las personas que participaron de algún crimen y si tienen relación entre ellas muéstrela.

```
MATCH (l:Location {address: '165 Laurel
Street'})<-[:OCCURRED_AT]-(c1:Crime)<-[:PARTY_TO]-(p1:Person)
OPTIONAL MATCH (l)<-[:OCCURRED_AT]-(c2:Crime)<-[:PARTY_TO]-(p2:Person)
WHERE p1 <> p2
WITH p1, p2
WHERE EXISTS ((p1)-[:FAMILY_REL]-(p2)) OR EXISTS ((p1)-[:KNOWS]-(p2)) OR
EXISTS
((p1)-[:KNOWS_LW]-(p2)) OR EXISTS ((p1)-[:KNOWS_PHONE]-(p2)) OR EXISTS
((p1)-[:KNOWS_SN]-(p2))
RETURN p1.name ||' '|| p1.surname AS Person1,

p2.name ||' '|| p2.surname || ' - ' ||
CASE
WHEN EXISTS ((p1)-[:FAMILY_REL]-(p2)) THEN 'FAMILY_REL'
WHEN EXISTS ((p1)-[:KNOWS]-(p2)) THEN 'KNOWS'
WHEN EXISTS ((p1)-[:KNOWS_LW]-(p2)) THEN 'KNOWS_LW'
WHEN EXISTS ((p1)-[:KNOWS_PHONE]-(p2)) THEN 'KNOWS_PHONE'
WHEN EXISTS ((p1)-[:KNOWS_SN]-(p2)) THEN 'KNOWS_SN'
END AS Relationship
```

5. Muestre la (o las) persona(s) que ha(n) realizado mas de 7 comunicaciones telefónicas.

```
MATCH
(p:Person)-[:HAS_PHONE]->(phone:Phone)<-[call:CALLED|CALLER]-
(pc:PhoneCall)
WITH p, COUNT(call) AS totalCalls
WHERE totalCalls > 7
RETURN p.name || ' '|| p.surname AS Person, totalCalls AS TotalCalls
```