ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Sistemas de bases de datos avanzados

2S-2020

Determinar la categoría predominante, respecto a la cantidad de descargas, para las aplicaciones móviles de tipo pago y libre acceso respectivamente

GRUPO 3

Aguirre Larrosa Stefanny Briggitte

Vera García Pedro Gabriel

Profesora: Echeverría Barzola Vanessa

FECHA DE ENTREGA: 07/12/2020

GUAYAQUIL – ECUADOR

Contenido

Descripción del Dataset

El dataset *googleplaystore.csv*, tomado del sitio *Kaggle*, está compuesto por más de 10.000 registros sobre aplicaciones móviles alojadas en la plataforma *Google Playstore*. La información que se puede encontrar sobre una aplicación está conformada por los 13 atributos siguientes:

1. **App. -** Que representa del nombre de la aplicación móvil.
2. **Category. -** Indica a qué categoría pertenece la aplicación móvil.
3. **Rating. -** Muestra la calificación general que el usuario promedio da a la aplicación móvil.
4. **Reviews. -** Este atributo contiene el número total de comentarios que ha recibido la aplicación móvil.
5. **Size. -** Comprende el tamaño, principalmente en MB, de la aplicación móvil.
6. **Installs. -** Indica el número de veces en que ha sido instalada la aplicación móvil, en otras palabras, el total de descargas de esta.
7. **Type. -** Muestra si la aplicación móvil es de libre acceso o de pago.
8. **Price. -** En caso de que la aplicación móvil sea de pago, este atributo muestra el precio de esta.
9. **Content Rating. -** Clasifica a la población de acuerdo con niños, mayores de 21 años y adultos para el acceso y visualización de la aplicación móvil.
10. **Genres. -** Muestra los distintos géneros a la que pertenece la aplicación móvil, además de su categoría principal.
11. **Last Updated. -** Contiene la fecha de la última actualización de la aplicación móvil.
12. **Current Ver. -** De acuerdo con el atributo anterior, este atributo señala en que versión se encuentra la aplicación móvil.
13. **Android Ver. -** Representa la versión de Android con la que es compatible la aplicación móvil.

Este dataset puede resultar bastante útil debido a que ofrece un abanico de posibilidades para analizar el comportamiento de los usuarios con respecto a las aplicaciones de la Play Store.

Código para operaciones CRUD

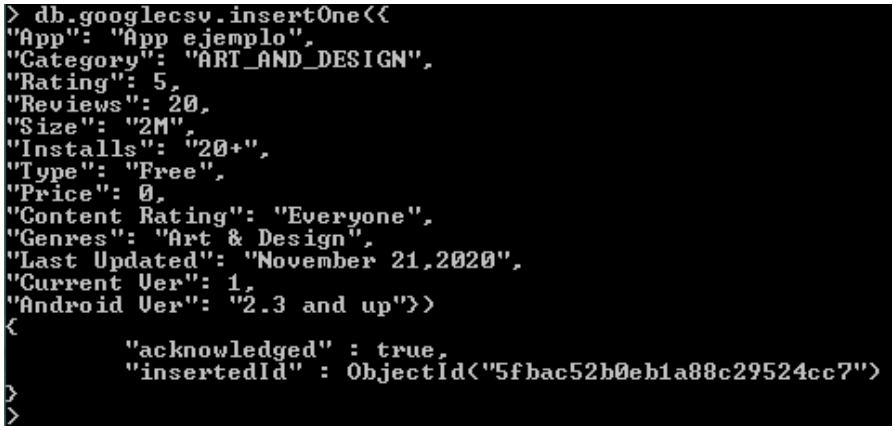
A continuación, se muestran los resultados obtenidos al realizar las operaciones CRUD.

Código para cargar datos

Para cargar los datos provenientes del archivo CSV, se creó una base de datos denominada *googlePlay* y una colección llamada *googlecsv.*

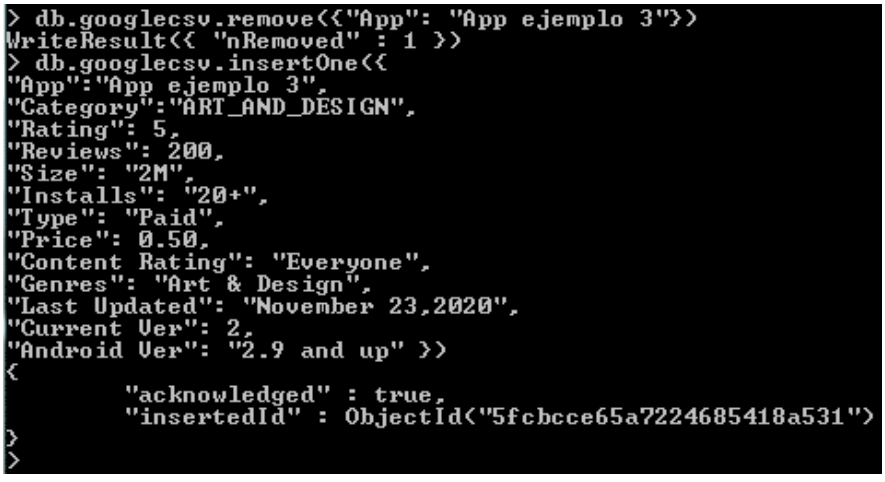
Código para insertar datos

Para agregar un nuevo documento en la colección *googlecsv* se utilizó la instrucción insertOne. A continuación, se muestra el resultado de las nuevas aplicaciones insertadas: “app ejemplo”, “app ejemplo 2” y “app ejemplo 3”.



```
> db.googlecsv.insertOne({
"App": "App ejemplo",
"Category": "ART_AND_DESIGN",
"Rating": 5,
"Reviews": 20,
"Size": "2M",
"Installs": "20+",
"Type": "Free",
"Price": 0,
"Content Rating": "Everyone",
"Genres": "Art & Design",
"Last Updated": "November 21,2020",
"Current Ver": 1,
"Android Ver": "2.3 and up"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5fbac52b0eb1a88c29524cc7")
}
>
```

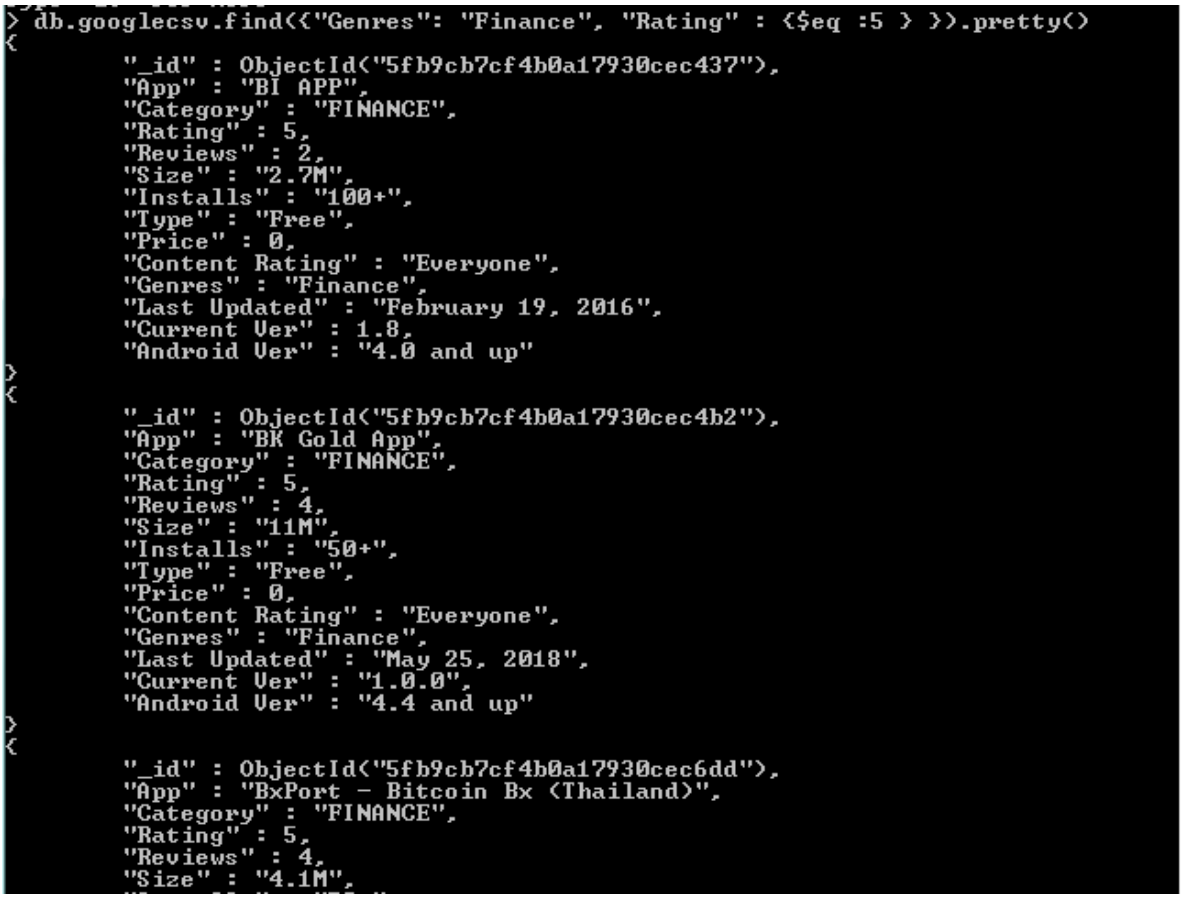
```
> db.googlecsv.insertOne({
"App": "App ejemplo 2",
"Category": "ART_AND_DESIGN",
"Rating": 5,
"Reviews": 20,
"Size": "2M",
"Installs": "20+",
"Type": "Free",
"Price": 0,
"Content Rating": "Everyone",
"Genres": "Art & Design",
"Last Updated": "November 22,2020",
"Current Ver": 1,
"Android Ver": "2.9 and up"})
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5fbac9d40eb1a88c29524cc8")
}
>
```



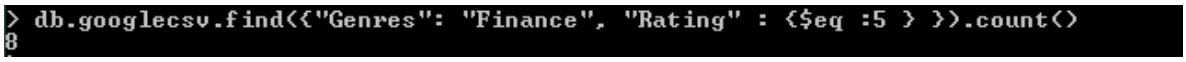
```
> db.googlecsv.remove({"App": "App ejemplo 3"})
WriteResult({ "nRemoved" : 1 })
> db.googlecsv.insertOne({
"App":"App ejemplo 3",
"Category":"ART_AND_DESIGN",
"Rating": 5,
"Reviews": 200,
"Size": "2M",
"Installs": "20+",
"Type": "Paid",
"Price": 0.50,
"Content Rating": "Everyone",
"Genres": "Art & Design",
"Last Updated": "November 23,2020",
"Current Ver": 2,
"Android Ver": "2.9 and up" })
{
        "acknowledged" : true,
        "insertedId" : ObjectId("5fcbcce65a7224685418a531")
}
>
```

Código para consultas

Para realizar las consultas se utilizó la instrucción *find.* En la siguiente imagen se muestra el resultado obtenido para obtener todas las aplicaciones cuyo género sea finanzas y tengan un Rating igual a 5.



Con la instrucción count () se puede conocer cuál es el tamaño del conjunto de documentos retornados.



La siguiente consulta muestra aquellas aplicaciones que no son de tipo Free y con rating menor o igual a 2.

```
> db.googlecsv.find({"Type" : {$ne : "Free"}, "Rating": {$lte: 2} } ).pretty()
{
        "_id" : ObjectId("5fb9cb7cf4b0a17930cebbd1"),
        "App" : "Speech Therapy: F",
        "Category" : "FAMILY",
        "Rating" : 1,
        "Reviews" : 1,
        "Size" : "16M",
        "Installs" : "10+",
        "Type" : "Paid",
        "Price" : "$2.99",
        "Content Rating" : "Everyone",
        "Genres" : "Education",
        "Last Updated" : "October 7, 2016",
        "Current Ver" : 1,
        "Android Ver" : "2.3.3 and up"
}
{
        "_id" : ObjectId("5fb9cb7cf4b0a17930cebbf3"),
        "App" : "G-Playlists",
        "Category" : "TOOLS",
        "Rating" : 1.8,
        "Reviews" : 53,
        "Size" : "3.4M",
        "Installs" : "1,000+",
        "Type" : "Paid",
        "Price" : "$1.49",
        "Content Rating" : "Everyone",
        "Genres" : "Tools",
        "Last Updated" : "May 19, 2018",
        "Current Ver" : 1.91,
        "Android Ver" : "4.0.3 and up"
}
{
        "_id" : ObjectId("5fb9cb7cf4b0a17930cec30d"),
        "App" : "Truck Driving Test Class 3 BC",
        "Category" : "FAMILY",
        "Rating" : 1,
        "Reviews" : 1,
        "Size" : "2.0M",
        "Installs" : "50+",
        "Type" : "Paid",
        "Price" : "$1.49",
        "Content Rating" : "Everyone",
        "Genres" : "Education",
        "Last Updated" : "April 9, 2012",
        "Current Ver" : 1,
        "Android Ver" : "2.1 and up"
}
{
        "_id" : ObjectId("5fb9cb7cf4b0a17930cec6d8"),
        "App" : "Bitcoin BX Thailand PRO",
        "Category" : "FINANCE",
        "Rating" : 1.7,
        "Reviews" : 21,
```

```
> db.googlecsv.find({"Type" : {$ne : "Free"}, "Rating": {$lte: 2} } ).count()
5
>
```

Código para modificaciones

Para modificar los datos almacenados, se utilizó la instrucción updateOne. En la siguiente imagen se muestra la actualización de los atributos *Type* y *Price* para el documento cuyo nombre es *App ejemplo.*

```
> db.googlecsv.updateOne({"App" : "App ejemplo"}, { $set: { "Type": "Paid", "Pri
ce": 10  } }, { upsert: true } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.googlecsv.find({"App" : "App ejemplo" } ).pretty()
{
        "_id" : ObjectId("5fbac52b0eb1a88c29524cc7"),
        "App" : "App ejemplo",
        "Category" : "ART_AND_DESIGN",
        "Rating" : 5,
        "Reviews" : 20,
        "Size" : "2M",
        "Installs" : "20+",
        "Type" : "Paid",
        "Price" : 10,
        "Content Rating" : "Everyone",
        "Genres" : "Art & Design",
        "Last Updated" : "November 21,2020",
        "Current Ver" : 1,
        "Android Ver" : "2.3 and up"
}
>
```

Para la segunda modificación realizada, se eligió al documento con nombre *App ejemplo 2* y se procedió a cambiar el atributo *Current Ver*.

```
> db.googlecsv.updateOne({"App" : "App ejemplo 2"}, { $set: { "Current Ver": 10
} }, { upsert: true } )
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.googlecsv.find({"App" : "App ejemplo 2" } ).pretty()
{
        "_id" : ObjectId("5fbac9d40eb1a88c29524cc8"),
        "App" : "App ejemplo 2",
        "Category" : "ART_AND_DESIGN",
        "Rating" : 5,
        "Reviews" : 20,
        "Size" : "2M",
        "Installs" : "20+",
        "Type" : "Free",
        "Price" : 0,
        "Content Rating" : "Everyone",
        "Genres" : "Art & Design",
        "Last Updated" : "November 22,2020",
        "Current Ver" : 10,
        "Android Ver" : "2.9 and up"
}
>
```

Para la tercera modificación, se utilizó el documento con nombre *App ejemplo* 3 y se modificó la cantidad de descargas identificado por el atributo *Installs*.

```
> db.googlecsv.updateOne({"App": "App ejemplo 3"}, {$set: {"Installs":"100+" }},
 {upsert:true})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
> db.googlecsv.find({"App": "App ejemplo 3"}).pretty()
{
        "_id" : ObjectId("5fcbd1eb5a7224685418a532"),
        "App" : "App ejemplo 3",
        "Category" : "ART_AND_DESIGN",
        "Rating" : 5,
        "Reviews" : 200,
        "Size" : "2M",
        "Installs" : "100+",
        "Type" : "Paid",
        "Price" : 0.5,
        "Content Rating" : "Everyone",
        "Genres" : "Art & Design",
        "Last Updated" : "November 23,2020",
        "Current Ver" : 2,
        "Android Ver" : "2.9 and up"
}
>
```

Código para eliminaciones

A continuación, se hace uso de la instrucción *remove*, para quitar de la colección a los documentos de nombre App ejemplo, App ejemplo 2 y App ejemplo 3.

```
> db.googlecsv.remove({"App": "App ejemplo"}, true)
WriteResult({ "nRemoved" : 1 })
> db.googlecsv.remove({"App": "App ejemplo 2"}, true)
WriteResult({ "nRemoved" : 1 })
> db.googlecsv.remove({"App": "App ejemplo 3"}, true)
WriteResult({ "nRemoved" : 1 })
>
```

Tarea por realizar con MAP-Reduce

La tarea consiste en determinar la categoría predominante, respecto a la cantidad de descargas, para las aplicaciones de tipo pago y libre acceso respectivamente. Para realizar este cometido se hará uso de los atributos: *Type*, *Installs* y *Category* para obtener el total de descargas que tiene cada categoría y luego proceder a sacar el máximo entre los valores calculados por las funciones Map y reduce.

Implementación

Se utilizó Python como lenguaje de programación y se hizo uso de la librería Pymongo[1] para establecer la conexión con mongoDB. Para instalarla se debe ejecutar el siguiente comando en consola.

```
python -m pip install pymongo
```

En cuanto al algoritmo Map Reduce, este se implementó de acuerdo con los ejemplos realizados en clase, sin embargo, a la función *execute* se le realizaron las siguientes modificaciones:

1. Se tuvo que añadir el argumento *nombre_archivo* para poder generar archivos con distinto nombre.
2. Se hizo uso de la función max de Python para elegir de entre todos los totales el valor más alto.

Estos cambios se hicieron para manejar los dos tipos de aplicaciones **Pago** y **Libre acceso** para así generar los archivos correspondientes que contendrán el resultado de la suma total de sus descargas y la elección del máximo de la lista de categorías. Esta modificación se encuentra en el archivo *MapReduce.py*

[1] https://pymongo.readthedocs.io/en/stable/

```
15      def execute(self, data, mapper, reducer, nombre_archivo):
16          for line in data:
17              mapper(line)
18
19          for key in self.intermediate:
20              reducer(key, self.intermediate[key])
21
22          jenc = json.JSONEncoder()
23          file = open(nombre_archivo, 'w', encoding="utf-8")
24          file.write('La categoría predominante es ')
25          # Obtener la cantidad mayor
26          maxi = max(self.result, key= lambda tupla: tupla[1])
27          # Escribir en el archivo
28          file.write(maxi[0]+' con '+ str(maxi[1])+" descargas")
29          file.write('\n\nResumen del conteo\n')
30          for item in self.result:
31              file.write(jenc.encode(item)+'\n')
32          file.close()
33
```

Los archivos *free.py* y *paid.py* contienen la llamada a las funciones mapper y reducer, y aunque su código es similar, lo que cambiará es la llamada hacia los datos almacenados ya que serán filtrados de acuerdo a su tipo como se lo puede observar a continuación:

```
27  if __name__ == '__main__':
28      # Conexión a MongoDB
29      cliente = pymongo.MongoClient("mongodb://localhost:27017/")
30      base = cliente["googlePlay"]
31      coleccion = base["googlecsv"]
32      proyeccion = {"_id": 0, "Category": 1, "Installs": 1}
33      resultado = coleccion.find({ "Type":"Free"}, proyeccion)
34      t_inicio = time()
35      mr.execute(resultado, mapper, reducer, "appsgratis.txt")
36      t_final = time() - t_inicio
37      print("\nEl tiempo de ejecución del programa es: %.5f segundos\n" %t_final)
```

Para las aplicaciones gratis, el filtro será “Type”: “Free”, mientras que para las de pago será “Type”: “Paid”. En esta sección de código también se notar que se realiza la conexión con mongoDB utilizando las funciones de la librería antes mencionada.

En la función *mapper* se realizó la extracción de los principales campos de la base de datos como lo son *Category* e *Installs* lo que ayudará a la obtención de la información para el resultado final.

```
8   def mapper(record):
9       category= record['Category']
10      installs= record['Installs']
11      mr.emit_intermediate(category, installs)
12
```

En la función *reducer* se realizó una limpieza de los datos obtenidos puesto que los valores del atributo *Installs* tenían un formato de tipo string y además se incluían símbolos como comas y signos de suma, por ejemplo: "1,000+". Posterior a este proceso, se realizó la operación para obtener la cantidad total de descargas para cada categoría, esto se pudo realizar con la ayuda de un acumulador declarado fuera del ciclo for. Como resultado se obtuvo un arreglo tipo clave valor: [Categoría, Total descargas].

```
13
14  def reducer(key, list_of_values):
15      total = 0
16      for t in list_of_values:
17          # Proceso necesario para transformar los valores de la clave "Installs"
18          # debido a que están almacenados como una cadena de caracteres
19          # y además utilizan el símbolo "+"
20          noplus = str(t).replace("+",'')
21          nocom = str(noplus).replace(",", '')
22          numero = float(nocom)
23          total += numero
24
25      mr.emit((key, total))
26
```

Por último, para calcular el tiempo de ejecución del programa se usa la función time.time antes de la función de Map-Reduce ***execute*** que es donde se hace el llamado de *mapper* y *reducer* junto con los datos extraídos de la base de datos.

```
34      t_inicio = time()
35      mr.execute(resultado, mapper, reducer, "appsgratis.txt")
36      t_final = time() - t_inicio
37      print("\nEl tiempo de ejecución del programa es: %.5f segundos\n" %t_final)
```

Resultados

Para ejecutar el programa se debe colocar en la línea de comandos: python free.py o paid.py. A continuación, se muestran los resultados para cada archivo.

Free.py

```
python free.py
```

Al ejecutar la línea anterior se genera el archivo appsgratis.txt. En la siguiente imagen se muestra el contenido de este, donde se puede ver que la categoría predominante para las aplicaciones de libre acceso es GAME que se la pudo calcular extrayendo el máximo de la suma total de instalaciones de las aplicaciones móviles del dataset.

MapReduce.py | appsgratis.txt | free.py | paid.py

appsgratis.txt

```
La categoría predominante es GAME con 35064924450.0 descargas

Resumen del conteo
["ART_AND_DESIGN", 124322100.0]
["AUTO_AND_VEHICLES", 53080061.0]
["BEAUTY", 27197050.0]
["BOOKS_AND_REFERENCE", 1921446260.0]
["BUSINESS", 1001502090.0]
["COMICS", 56086150.0]
["COMMUNICATION", 32645916201.0]
["DATING", 264289457.0]
["EDUCATION", 870850000.0]
["ENTERTAINMENT", 2868960000.0]
["EVENTS", 15973160.0]
["FINANCE", 876463132.0]
["FOOD_AND_DRINK", 273838751.0]
["HEALTH_AND_FITNESS", 1582498402.0]
["HOUSE_AND_HOME", 168712461.0]
["LIBRARIES_AND_DEMO", 62995810.0]
["LIFESTYLE", 536464429.0]
```

Paid.py

```
python paid.py
```

Al ejecutar la línea anterior se genera el archivo appspago.txt. En la siguiente captura de pantalla se puede observar que la categoría predominante de las aplicaciones de pago es FAMILY que se la pudo calcular extrayendo el máximo de la suma total de instalaciones de las aplicaciones móviles del dataset.

MapReduce.py | appspago.txt | free.py | paid.py

appspago.txt

```
La categoría predominante es FAMILY con 31271814.0 descargas

Resumen del conteo
["BUSINESS", 412775.0]
["COMMUNICATION", 1360050.0]
["DATING", 21350.0]
["EDUCATION", 602000.0]
["ENTERTAINMENT", 200000.0]
["FOOD_AND_DRINK", 60000.0]
["HEALTH_AND_FITNESS", 574110.0]
["GAME", 21099965.0]
["FAMILY", 31271814.0]
["MEDICAL", 1020033.0]
["PHOTOGRAPHY", 3978740.0]
["SPORTS", 1243815.0]
["PERSONALIZATION", 5258794.0]
["PRODUCTIVITY", 1412055.0]
["WEATHER", 812000.0]
["TOOLS", 1727441.0]
["TRAVEL_AND_LOCAL", 183060.0]
```

Tiempo de Ejecución

Al término de la ejecución del programa, por consola, se mostrará el tiempo total que duró la tarea realizada con Map-Reduce.