Andrea Nicte Vicente Campos

201404104

Requisitos del sistema

> jdk 11.0.7
>
> python 3
>
> linux basado en Debian

Herramientas utilizadas

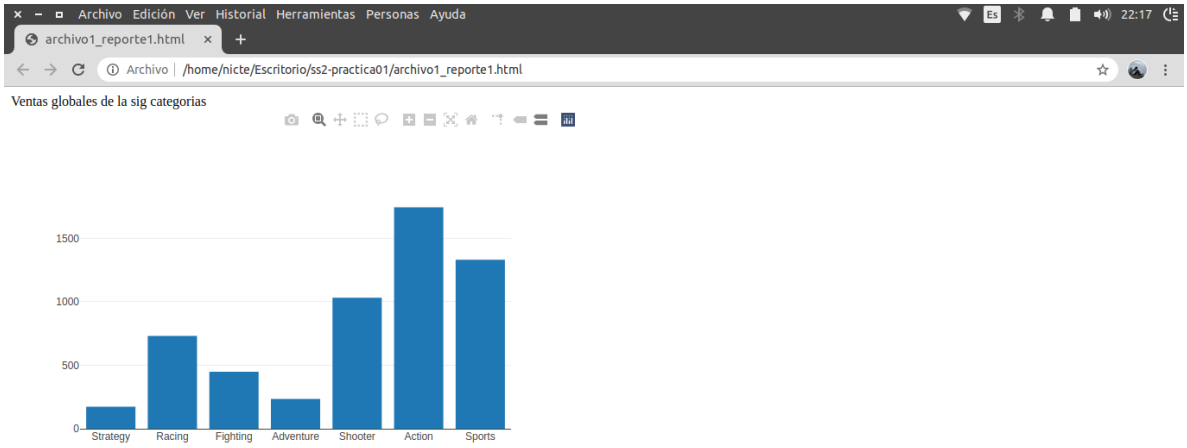
> jdk 11.0.7
>
> IDE PyCharm
>
> python3
>
> Spark 3.0.0
>
> Apache Headoop 2.7
>
> SparkContext

Transformaciones

Ventas glables por categorias de generos

```
 get_rows = self.data.map(lambda row: (row[4], float(row[10])))

        genre_filters= get_rows.filter( lambda row:  (row[0].lower() ==
"action") \
                                        or (row[0].lower() == "sports") \
                                        or (row[0].lower() == "fighting") \
                                        or (row[0].lower() == "shooter") \
                                        or (row[0].lower() == "racing") \
                                        or (row[0].lower() == "adventure") \
                                        or (row[0].lower() == "strategy") )

        total = genre_filters.reduceByKey(lambda x, y: x + y)
```

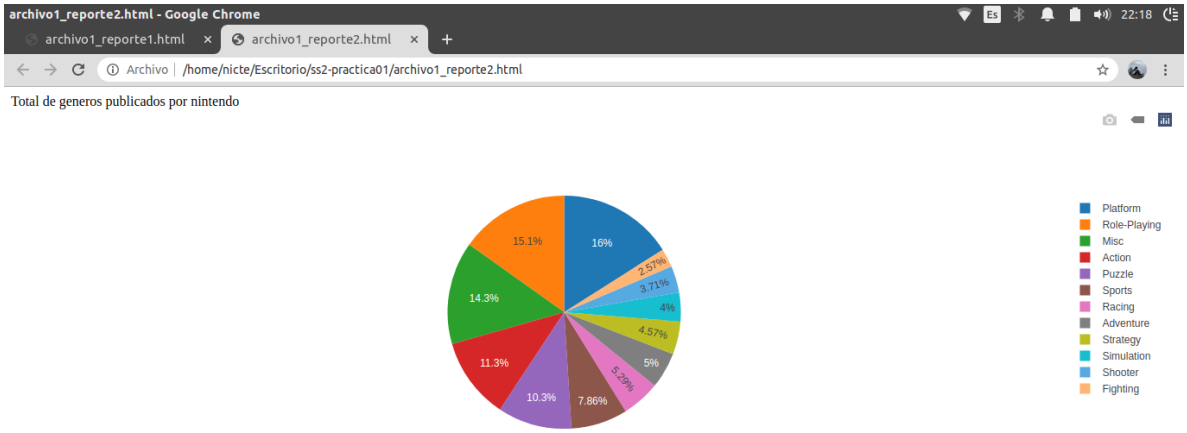
> Se filtra por categorias
>
> Se suman las ventas con el metodo reduceBykey



Total de generos publicados por Nintendo

```
        get_rows = self.data.map(lambda row: (row[5] ,row[4], 1))

        rows_nintendo = get_rows.filter(lambda row: row[0].lower() =="nintendo")

        rows_final = rows_nintendo.map(lambda row: (row[1], row[2]))

        total = rows_final.reduceByKey(lambda x, y: x + y)

        print("segunda consulta archivo 2")
```

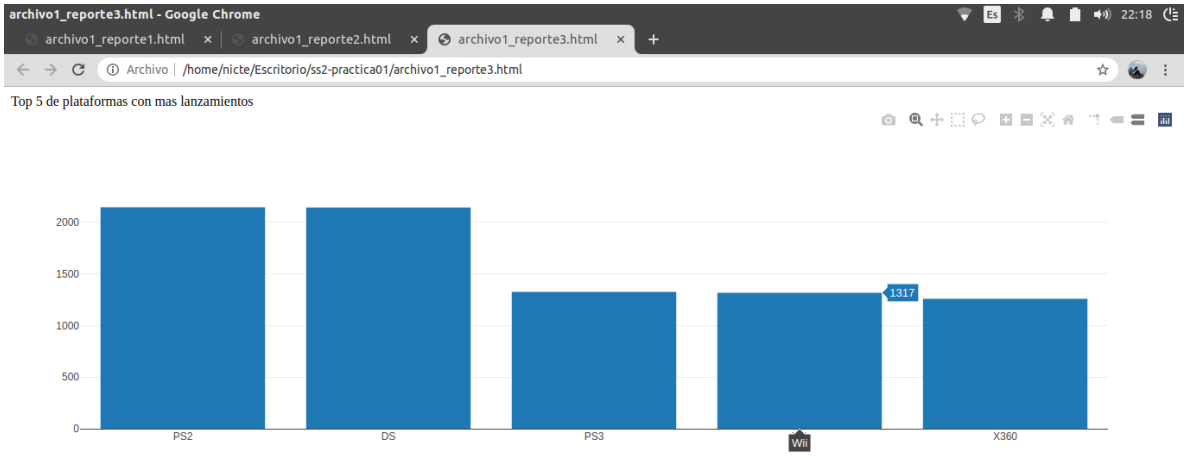
> Se obtienen las columnas que se van a utilizar, y se agrega un uno, el cual va servir para hacer la suma del resultado final
>
> Se filtran solo las que tienen Nintendo en su columna
>
> Se hace un map para quitar la columna de nintendo
>
> Se hace una suma de los generos



Plataformas con mas lanzamientos

```
get_rows = self.data.map(lambda row: (row[2], 1))

      total = get_rows.reduceByKey(lambda x, y: x + y)

      total_ordenado = total.sortBy(lambda row: row[1], ascending=False)

      print("tercera consulta archivo 2")
      #print(get_rows.collect())
      print(total_ordenado.collect()[0:5])

      xxx = get_ejex_ejey(total_ordenado)
```

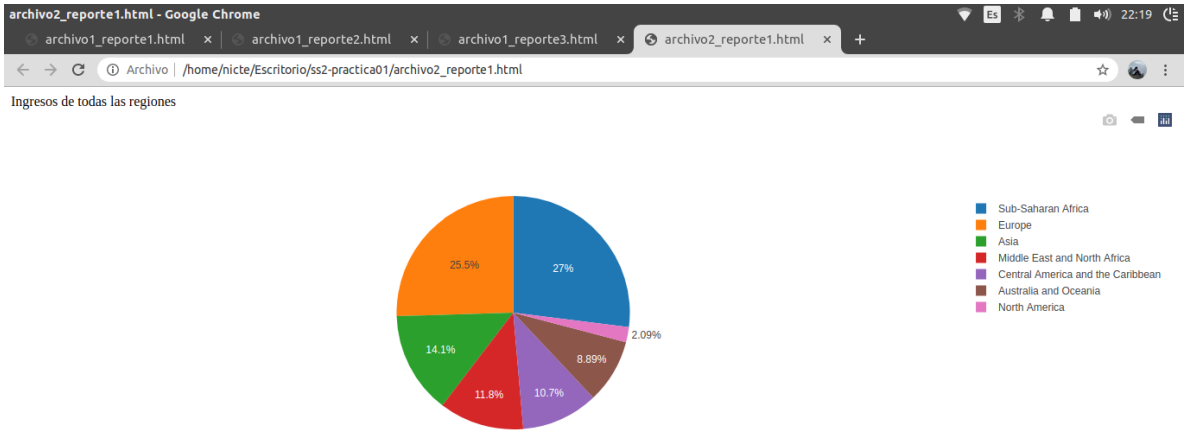
> Se obtiene la columna de prataforma y se le agrega a un uno a nuestro nuevo arreglo de pySpark
>
> Se suman, las plataformas
>
> Se ordean de manera descendente



Ingresos de todas las regiones

```
        get_rows = self.data.map(lambda row: (row[0], float(row[13])))
        total = get_rows.reduceByKey(lambda x, y: x + y)
        print("primera consulta archivo 2")
        #print(get_rows.collect())
        print(total.collect())

        xxx = get_ejex_ejey(total)
```

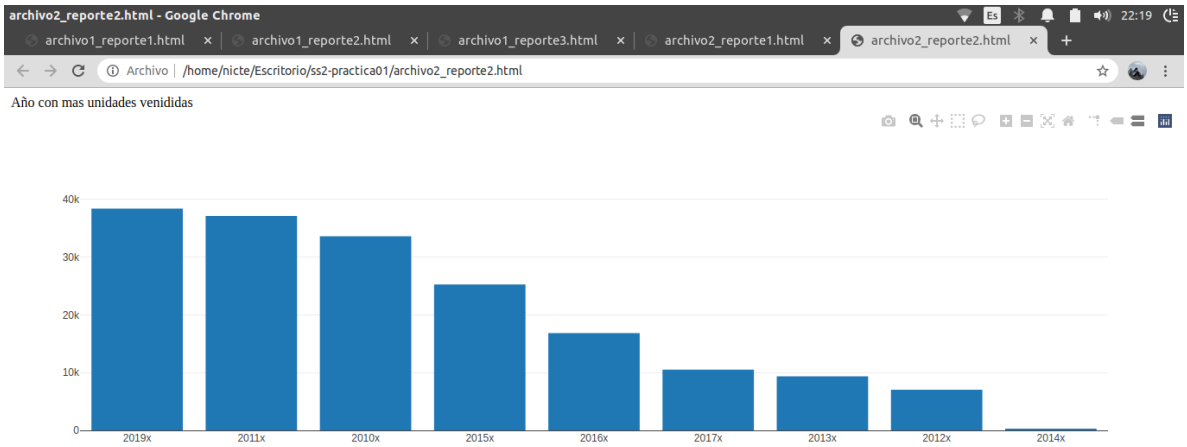
> Se obtiene la columna e regiones y sus ingresos
>
> Se suman, las regiones



Año con mas unidades vendidas en guatemala

```
get_rows = self.data.map(lambda row: (row[1], row[5].split("/")[2],
int(row[8]))) \
    .filter(lambda row: row[0].lower() == 'guatemala')

total = get_rows.map(lambda row: (row[1]+"x", row[2])).reduceByKey(lambda
x, y: x+y)

orden = total.sortBy(lambda row: row[1], ascending = False)
```

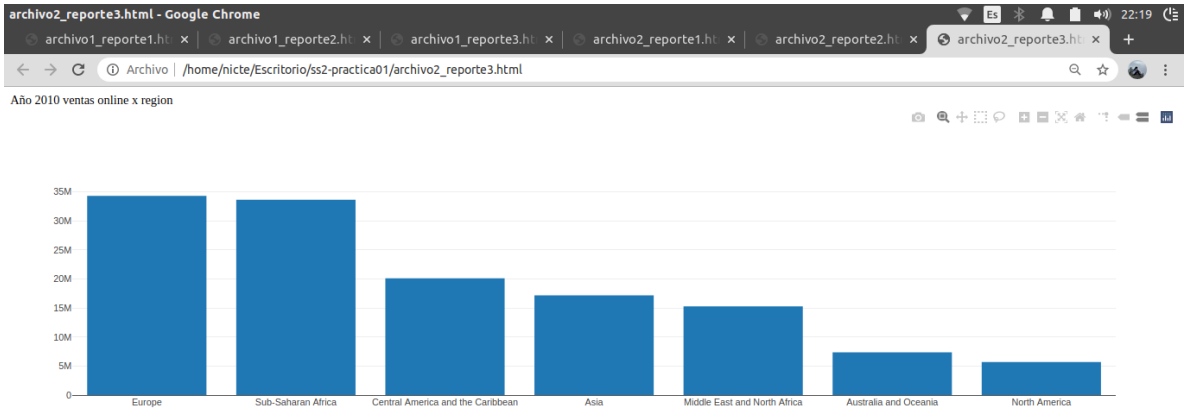
> Se obtienen las columas correspondiente
>
> La fecha se maneja como un string
>
> Se filtran solo los que sean del pais de guatemala
>
> Se hace una suman las unidades
>
> Se ordean



Año 2010 ventas online

```
  get_rows = self.data.map(lambda row: (row[5].split("/")[2], row[0],
float(row[13]), row[3]))
        year2010 = get_rows.filter(lambda row: row[0]=="2010" and row[3].lower()
== "online")

        total = year2010.map(lambda row: (row[1], row[2])).reduceByKey(lambda x,
y: x + y)
        total_ordenado = total.sortBy(lambda row: row[1], ascending=False)
```

> Se obtienen las columnas correspondiente
>
> Se toman los datos den los cuales tienen registro 2010 y online
>
> Se quitan las columnas que no funcionan para el reporte
>
> Se ordenan los datos



Top de muertes por raza

```
get_rows = self.data.map(lambda row: (row[3], 1))

        total_race = get_rows.reduceByKey(lambda x, y: x + y)

        total_sort = total_race.sortBy(lambda row: row[1], ascending=False)
```

> Se obtienen las columnas correspondientes, se agreaga un 1 el cual se va sumar para obtener el total
>
> Se suman por clave
>
> Se ordenan