```
from pyspark.sql import *
from pyspark.sql.functions import *
from pyspark import SparkContext
import pandas as pd

spark = SparkSession.builder.getOrCreate()
sc = spark.sparkContext


libros = [
 (1, 'Crónicas marcianas', 'ciencia ficción', 'Ray Bradbury'),
 (2, 'La vuelta al mundo en ochenta días', 'aventura', 'Julio Verne'),
 (3, 'El Alquimista', 'romance', 'Paulo Cohelo'),
  (4, 'Libro', 'Drama', 'Alguien'),
  (5, 'Libro2', 'Terror', 'Alguien2'),
  (6, 'Libro3', 'Drama', 'Alguien3'),
 (7, 'Librito', 'Misterio', 'Mateo Rojas')
]
```

```
ventas = [
(100, 1, 10, 'julio', 2020, '13:00:24', 300),
(101, 1, 20, 'julio', 2020, '15:04:00', 300),
(102, 3, 23, 'julio', 2020, '16:01:01', 250),
(101, 5, 20, 'julio', 2020, '15:04:00', 300),
(102, 6, 23, 'julio', 2020, '16:01:01', 250),
(103, 1, 8, 'agosto', 2020, '16:22:23', 300),
(104, 3, 12, 'agosto', 2020, '17:00:00', 300),
(105, 4, 12, 'agosto', 2020, '17:07:07', 300),
(106, 4, 18, 'agosto', 2020, '11:02:00', 250),
(107, 4, 19, 'agosto', 2020, '11:42:00', 250),
(106, 5, 18, 'agosto', 2020, '11:02:00', 250),
(107, 5, 19, 'agosto', 2020, '11:42:00', 250),
(106, 6, 18, 'agosto', 2020, '11:02:00', 250),
(107, 6, 19, 'agosto', 2020, '11:42:00', 250),
(107, 6, 19, 'agosto', 2020, '11:42:00', 250),
(107, 6, 19, 'agosto', 2020, '11:42:00', 250),
(107, 6, 19, 'julio', 2020, '11:42:00', 250),
(107, 6, 19, 'agosto', 2020, '11:42:00', 250),
(108, 2, 22, 'agosto', 2020, '18:33:00', 250)
]
```

```
ventas_rdd = sc.parallelize(ventas)
libros_rdd = sc.parallelize(libros)
```

Se tiene un RDD con la información de libros:

(id_libro, nombre, género, autor)

y otro con las ventas de distintos ejemplares de esos libros.

(id_venta, id_libro, dia_venta, mes_venta, año_venta, hora_venta, precio)

a) Indicar el género con más ventas de agosto de 2020.

```
mas_vendidos = ventas_rdd.filter(lambda x: x[3] == 'agosto' and x[4] == 2020)\
.map(lambda x: (x[1],1)).reduceByKey(lambda x,y: x + y)
mas_vendidos.collect()
```

```
[(4, 3), (6, 5), (2, 1), (1, 1), (3, 1), (5, 2)]
```

Filtro las ventas de agosto de 2020, me quedo con el ID de libro y un uno para contar. Luego sumo esos 1 y me queda el id del libro y la cantidad que se vendieron.

```
generos_ventas = libros_rdd.map(lambda x: (x[0], x[2])).join(mas_vendidos)\
.map(lambda x: (x[1][0], x[1][1])).reduceByKey(lambda x,y : x+ y).cache()
generos_ventas.collect()
```

```
[('Drama', 8),
 ('Terror', 2),
 ('ciencia ficción', 1),
 ('aventura', 1),
 ('romance', 1)]
```

Ahora con el rdd de libros, me quedo con su id y su genero. Lo joineo con los libros mas vendidos. Termino filtrando y me quedo con el genero y la cantidad, luego hago un reduce by key por si hay dos libros de igual genero. Lo caheo porque lo uso mas de una vez luego.

```
generos_ventas.reduce(lambda x,y: x if x[1] > y[1] else y)
```

```
('Drama', 8)
```

Con un Reduce, muestro el mas vendido

b) Para los libros de los 5 géneros más vendidos en agosto de 2020, indicar el nombre del libro que presenta mayor aumento en el número de ventas con respecto al mes anterior.

```
top_generos = generos_ventas.takeOrdered(5, lambda x: -x[1])
top = sc.parallelize(top_generos)
top.collect()
```

```
[('Drama', 8),
 ('Terror', 2),
 ('ciencia ficción', 1),
 ('aventura', 1),
 ('romance', 1)]
```

Uso el mismo rdd que me dio el punto anterior, tomo los 5 mas vendidos y lo vuelvo a transformar a rdd, ya que me devolvia una lista

```
libros_filtro = libros_rdd.map(lambda x: (x[2], (x[0],x[1])))
libros_filtro.collect()
```

```
[('ciencia ficción', (1, 'Crónicas marcianas')),
 ('aventura', (2, 'La vuelta al mundo en ochenta días')),
 ('romance', (3, 'El Alquimista')),
 ('Drama', (4, 'Libro')),
 ('Terror', (5, 'Libro2')),
 ('Drama', (6, 'Libro3')),
 ('Misterio', (7, 'Librito'))]
```

De los lbros, me quedo con el genero como clave luego como valor el id y el nombre

```
libros_de_generos = top.join(libros_filtro).map(lambda x: (x[1][1][0], x[1][1][1]))
libros_de_generos.collect()
```

```
[(4, 'Libro'),
 (6, 'Libro3'),
 (5, 'Libro2'),
 (1, 'Crónicas marcianas'),
 (2, 'La vuelta al mundo en ochenta días'),
 (3, 'El Alquimista')]
```

Estos son los libros de los 5 géneros más vendidos en agosto de 2020

Lo que hice fue joinear los top generos con los libros y mapear para que quede ID nombre

```
libros_por_mes = ventas_rdd.map(lambda x: (x[1],x[3]))\
.join(libros_de_generos).map(lambda x: ((x[1][1],x[1][0]),1))
libros_por_mes.collect()
```

```
[(('Libro3', 'julio'), 1),
 (('Libro3', 'agosto'), 1),
 (('Libro3', 'agosto'), 1),
 (('Libro3', 'agosto'), 1),
 (('Libro3', 'agosto'), 1),
 (('Libro3', 'julio'), 1),
 (('Libro3', 'agosto'), 1),
 (('Crónicas marcianas', 'julio'), 1),
 (('Crónicas marcianas', 'julio'), 1),
 (('Crónicas marcianas', 'agosto'), 1),
 (('La vuelta al mundo en ochenta días', 'agosto'), 1),
 (('El Alquimista', 'julio'), 1),
 (('El Alquimista', 'agosto'), 1),
 (('Libro', 'agosto'), 1),
 (('Libro', 'agosto'), 1),
 (('Libro', 'agosto'), 1),
 (('Libro2', 'julio'), 1),
```

```
 (('Libro2', 'agosto'), 1),
 (('Libro2', 'agosto'), 1)]
```

Ahora mapeo las ventas para que quede Ide libro y mes, lo joineo con los libros anteriormente obtenidos y mapeo para que el titulo y el mes sea la clave y el valor un uno para contar

```
libros_por_mes.reduceByKey(lambda x,y: x+y).map(lambda x: (x[0][0], x[1]))\
.reduceByKey(lambda x,y: x-y).reduce(lambda x,y: x if x[1] > y[1] else y)
```

```
('Libro3', 3)
```

Sumo por clave y obtengo las ventas por mes de los libros. Me quedo con el libro y sus ventas por mes, las resto y obtengo la diferencia. Por ultimo muestro el que mayo diferencia tuvo.

En este punto b no use cache ya que no use nada mas de una vez.