```python
from pyspark.sql import SparkSession
from pyspark.sql.functions import explode, size

spark = SparkSession.builder.appName("ComplexDataExample").getOrCreate()

# Crear un DataFrame con datos complejos
data = [(1, ["apple", "banana"], {"a": 1, "b": 2}),
        (2, ["orange", "grape"], {"x": 3, "y": 4})]
df = spark.createDataFrame(data, ["id", "fruits", "scores"])

# Operaciones con arrays
with_array_length = df.withColumn("fruit_count", size(df.fruits))

# Operaciones con mapas
with_map_value = df.withColumn("score_a", df.scores.getItem("a"))

# Explotar (explode) un array
exploded_fruits = df.select(df.id, explode(df.fruits).alias("fruit"))

with_array_length.show()
with_map_value.show()
exploded_fruits.show()

spark.stop()
```

Este código demuestra cómo trabajar con **datos complejos** (arrays y mapas) dentro de DataFrames de PySpark. Spark SQL proporciona funciones integradas para manipular estos tipos de datos.

Desglose del Código:

1. **`from pyspark.sql import SparkSession`**: Importa la clase `SparkSession`.
2. **`from pyspark.sql.functions import explode, size`**: Importa funciones específicas para trabajar con datos complejos:
    - `size()`: Devuelve el tamaño (número de elementos) de un array o un mapa.
    - `explode()`: Desestructura un array o mapa en múltiples filas, con cada elemento del array o cada par clave-valor del mapa en una fila separada.
3. **`spark = SparkSession.builder.appName("ComplexDataExample").getOrCreate()`**: Crea o obtiene una sesión de Spark con el nombre de aplicación "ComplexDataExample".
4. **Creación del DataFrame con Datos Complejos:**
    - Se crea una lista de tuplas `data`. Cada tupla contiene:
        - Un `id` (entero).
        - Una lista de strings (`fruits`).

        - Un diccionario (map) de strings a enteros (`scores`).
    - Se crea un DataFrame `df` a partir de esta lista, con las columnas "id", "fruits" y "scores". Spark automáticamente infiere los tipos de datos complejos (array y map).
5. **Operaciones con Arrays:**
    - **`with_array_length = df.withColumn("fruit_count", size(df.fruits))`**: Se agrega una nueva columna llamada "fruit_count" al DataFrame. La función `size(df.fruits)` se aplica a la columna "fruits" (que es un array) para obtener el número de elementos en cada array.
6. **Operaciones con Mapas:**
    - **`with_map_value = df.withColumn("score_a", df.scores.getItem("a"))`**: Se agrega una nueva columna llamada "score_a". Se utiliza la función `getItem("a")` para acceder al valor asociado con la clave "a" dentro de la columna "scores" (que es un mapa). Si la clave "a" no existe en un mapa, el valor resultante será `null`.
7. **Explotar (Explode) un Array:**
    - **`exploded_fruits = df.select(df.id, explode(df.fruits).alias("fruit"))`**: Se crea un nuevo DataFrame llamado `exploded_fruits`.
        - `df.select(df.id, ...)`: Se seleccionan la columna "id" del DataFrame original.
        - `explode(df.fruits).alias("fruit")`: La función `explode(df.fruits)` toma la columna "fruits" (el array) y crea una nueva fila para cada elemento del array. El valor de cada elemento del array se coloca en una nueva columna, que se le asigna el alias "fruit". Las otras columnas del DataFrame original se duplican para cada elemento explotado.
8. **Mostrar Resultados:**
    - `with_array_length.show()`: Muestra el DataFrame con la columna "fruit_count".
    - `with_map_value.show()`: Muestra el DataFrame con la columna "score_a".
    - `exploded_fruits.show()`: Muestra el DataFrame con las filas "explotadas" para cada fruta.
9. **`spark.stop()`**: Detiene la sesión de Spark.

Salida Esperada del Código:

```
+---+--------------+--------------+-----------+
| id|        fruits|        scores|fruit_count|
+---+--------------+--------------+-----------+
|  1| [apple, banana]|{a -> 1, b -> 2}|          2|
|  2|[orange, grape]|{x -> 3, y -> 4}|          2|
+---+--------------+--------------+-----------+

+---+--------------+--------------+-------+
| id|        fruits|        scores|score_a|
+---+--------------+--------------+-------+
|  1| [apple, banana]|{a -> 1, b -> 2}|      1|
|  2|[orange, grape]|{x -> 3, y -> 4}|   null|
+---+--------------+--------------+-------+
```

```
+---+------+
| id| fruit|
+---+------+
|  1| apple|
|  1|banana|
|  2|orange|
|  2| grape|
+---+------+
```