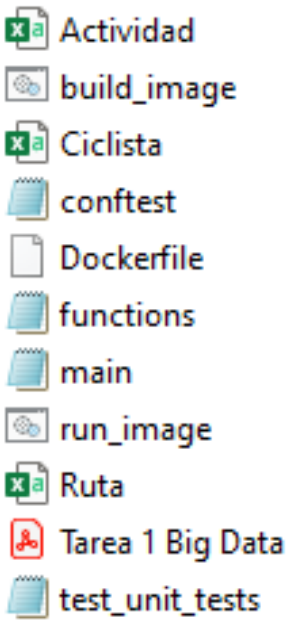
Algunas pistas para la Tarea 1

Contenido

A. Qué archivos podría tener en la carpeta:



B. Qué podría contener el archivo conftest:

conftest.py

C: > LuisAlex > CURSO_BigData > Lec2 > Tarea 1 > solucion_tarea1 > conftest.py > ...

```python
import pytest

from pyspark.sql import SparkSession


@pytest.fixture(scope="module")
def spark_session():
    """A fixture to create a Spark Context to reuse across tests."""
    s = SparkSession.builder.appName('pytest-local-spark').master('local') \
        .getOrCreate()

    yield s

    s.stop()

```

C. Cómo se podría organizar el archivo main:

- Imports
- SparkSession.builder
- armar los conjuntos de datos
- función para hacer el join
- función para calcular
- función para top

D. Cómo podría inicar el archivo main:

main: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
# Importar bibliotecas necesarias

from pyspark.sql import SparkSession
from pyspark.sql.types import (IntegerType, StringType, StructField, StructType, FloatType, DateType)
import sys
from functions import join_dfs, aggregate, top_n

# Inputs para crear los dataframes

input_1= sys.argv[1]
input_2= sys.argv[2]
input_3= sys.argv[3]

# Creación de la sesión de Spark
```

E. Cómo podría iniciar el archivo functions:

functions: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
from pyspark.sql.functions import col, dense_rank, sum, mean, col
from pyspark.sql.window import Window


def join_dfs(df_1, df_2, df_3, on_column1, on_column2, type1 = 'inner', type2 = 'inner'):
    """
    Une tres dataframe según las columnas y el tipo de join indicados por el usuario.
    Retorna un dataframe con los datos unidos de los tres dataframes.
```

F. Cómo podría iniciar el archivo test_unit_tests:

```
from functions import join_dfs, aggregate, top_n
# Unit tests para el join
# Test 1
def test_join_pass(spark_session):
    print("Unit Test: join pass")
    ciclistas = [(336996814, 'Javier Rodriguez', 'Cartago'),
                 (618541295, 'Olga Martinez', 'Puntarenas'),
                 (449300413, 'Jose Fernandez', 'Heredia'),
                 (442900777, 'Francisco Vargas', 'Heredia')]
    rutas = [(602, 'Puerto Viejo', 6.63),
             (901, 'Puntarenas - Esparza', 6.09),
             (401,'Heredia - San Joaquín', 4.45),
             (402,'Alajuela - San Rafael', 9.43)]
    actividad = [(901, 336996814, '2021-11-01'),
             (402, 618541295, '2021-11-03'),
             (402, 449300413, '2021-11-06'),
             (602, 442900777, '2021-11-02')]

    ciclistas_df  = spark_session.createDataFrame(ciclistas,['cedula', 'nombre_ciclista','provincia'])
    rutas_df = spark_session.createDataFrame(rutas,['codigo_de_ruta', 'nombre_ruta','kms'])
    actividad_df = spark_session.createDataFrame(actividad, ['codigo_de_ruta', 'cedula', 'fecha'])

    joined_df = join_dfs(ciclistas_df, actividad_df, rutas_df, 'cedula', 'codigo_de_ruta', 'inner', 'inner')
    correct_df = spark_session.createDataFrame([(442900777, 'Francisco Vargas', 'Heredia', 602, '2021-11-02', 'Puerto Viejo', 6.63),
                                                (618541295, 'Olga Martinez', 'Puntarenas', 402, '2021-11-03', 'Alajuela - San Rafael', 9.43),
                                                (449300413, 'Jose Fernandez', 'Heredia', 402, '2021-11-06', 'Alajuela - San Rafael', 9.43),
                                                (336996814, 'Javier Rodriguez', 'Cartago', 901, '2021-11-01', 'Puntarenas - Esparza', 6.09)],
    ['cedula', 'nombre_ciclista', 'provincia', 'codigo_de_ruta', 'fecha', 'nombre_ruta', 'kms'])
    assert joined_df.collect() == correct_df.collect()
# Test 2
```

G. Cómo se podrían ver algunas partes del archivo PDF con las indicaciones:

Archivos necesarios para ejecutar el código:

- actividad
- build_image
- ciclistas
- conftest
- Dockerfile
- functions
- main
- run_image
- rutas
- test_unit_tests

Instrucciones para el código principal:

Descargar y descomprimir el zip en un directorio local.
Abra el command prompt de su sistema y busque la dirección en la que guardó la carpeta. Debe nombrarla "tarea_1".
Luego, debe construir la imagen. Utilice el comando *docker* <***build --tag tarea_1 .***> En caso de que no funcione pruebe con <***bash build_image.sh***> o bien con <***sh***> o <***./***>.
Ahora debe correr la imagen con el comando <***docker run -p 8888:8888 -i -t tarea_1 /bin/bash***> o en su defecto <***run_image.sh***>.
Dentro del contenedor ejecute el comando <***spark-submit main.py ciclistas.csv rutas.csv actividad.csv***>.

Instrucciones para las pruebas unitarias:

1. Dentro del contenedor ejecute <***pytest -s***>. Si desea omitir el contenido de las pruebas utilice <***pytest***>.