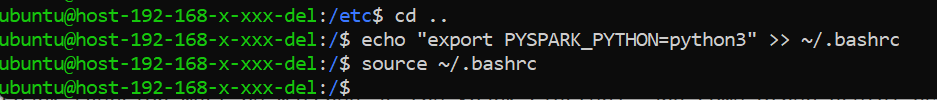
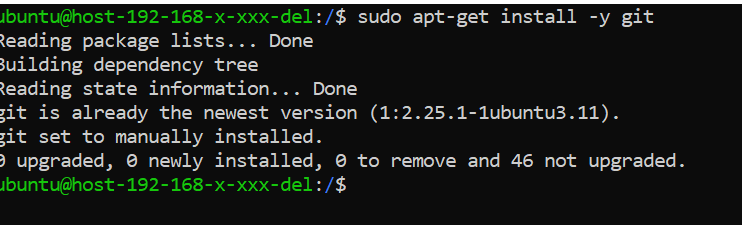
* A computer screen with white text

  Description automatically generatedmanually define a hostname for all the hosts on the de1 project. this will ensure connections of spark between difference instances:
* A black screen with white text

  Description automatically generatedHostname change screenshot
* Env variable so the workers know which Python to use..



* install git



* # install python dependencies, and start jupyterlab
* check the version:

pip 20.0.2 from /usr/lib/python3/dist-packages/pip (python 3.8)

A screen shot of a computer code

Description automatically generated

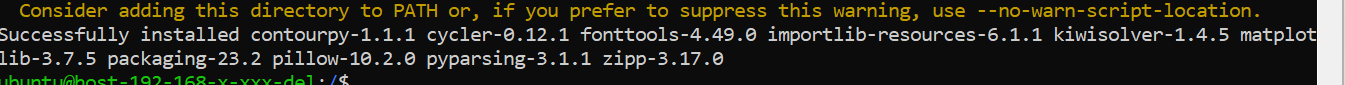
* # install pyspark (version must be matched as the Spark cluster), and some other useful deps

A screenshot of a computer program

Description automatically generatedCode : python3 -m pip install pyspark==3.5.0 –user

A screenshot of a computer program

Description automatically generatedCode: python3 -m pip install pandas –user

Code : python3 -m pip install matplotlib –user

* Git clone



* A screen shot of a computer screen

  Description automatically generated# install jupyterlab
* Start Jupyter Lab

A computer screen with text on it

Description automatically generatedCode : jupyter lab --ip=<vm\_local\_ip>

A computer screen shot of a computer program

Description automatically generated

A black screen with white text

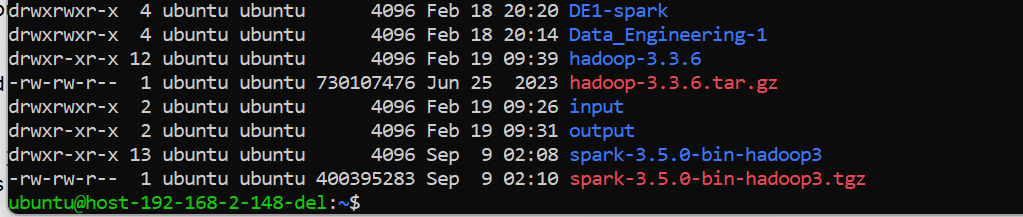
Description automatically generated--Hadoop Version

A computer screen shot of a program

Description automatically generatedHadoop -single setup

* A screen shot of a computer program

  Description automatically generatedSpark Download screenshot

🡪 Extract spark screenshot

* A computer screen with text

  Description automatically generatedSpark starting screenshot

A computer screen with blue text

Description automatically generated🡪 It is running confirmation.

Task-A

A1.1: Code is : For Eng language count

#A1.1: Read the English transcripts with Spark, and count the number of lines.

spark\_context = spark\_session.sparkContext

book\_read\_en = spark\_context.textFile('hdfs://192.168.2.250:9000/europarl/europarl-v7.sv-en.en').cache().setName("English Transcripts RDD")

english\_line\_count = book\_read\_en.count()

print("Number of lines in English transcripts:", english\_line\_count)

Ans: Number of lines in English transcripts: 1862234

# A.1.2 Do the same with the other language (Svenska) (so that you have a separate lineage of RDDs for each).

spark\_context = spark\_session.sparkContext

book\_read\_sv = spark\_context.textFile('hdfs://192.168.2.250:9000/europarl/europarl-v7.sv-en.sv').cache().setName("English Transcripts RDD")

svenska\_line\_count = book\_read\_sv.count()

print("Number of lines in Svenska transcripts:", svenska\_line\_count)

Ans: Number of lines in Svenska transcripts: 1862234

# A.1.3 Verify that the line counts are the same for the two languages.(English & Svenska)

if english\_line\_count == svenska\_line\_count:

print("Line counts are the same for both languages.")

else:

print("Line counts are different for the two languages.")

Ans : Line counts are the same for both languages.

# A.1.4 Count the number of partitions.

# Count the number of partitions for English transcripts

english\_partitions = book\_read\_en.getNumPartitions()

# Count the number of partitions for the other language transcripts

other\_language\_partitions = book\_read\_sv.getNumPartitions()

print("Number of partitions for English transcripts:", english\_partitions)

print("Number of partitions for the Svenska language transcripts:", other\_language\_partitions)

Ans :

Number of partitions for English transcripts: 2

Number of partitions for the Svenska language transcripts: 3

A.2:

Hints:

from pyspark import SparkContext

# Create SparkContext

spark\_context = SparkContext.getOrCreate()

# Define preprocessing function to run driver applciation

def preprocess\_text(text):

# Lowercase the text

text = text.lower()

# Tokenize the text (split on space)

tokens = text.split()

return tokens

# Read English transcripts

english\_rdd = spark\_context.textFile('hdfs://path\_to\_english\_transcripts')

# Read transcripts in the other language (e.g., Swedish)

other\_language\_rdd = spark\_context.textFile('hdfs://path\_to\_other\_language\_transcripts')

# Apply preprocessing function to both RDDs

english\_preprocessed\_rdd = english\_rdd.map(preprocess\_text)

other\_language\_preprocessed\_rdd = other\_language\_rdd.map(preprocess\_text)

# Inspect 10 entries from each RDD

print("English Transcripts - 10 Entries after Preprocessing:")

print(english\_preprocessed\_rdd.take(10))

print("\nOther Language Transcripts - 10 Entries after Preprocessing:")

print(other\_language\_preprocessed\_rdd.take(10))

# Verify that line counts still match after preprocessing

english\_line\_count = english\_preprocessed\_rdd.count()

other\_language\_line\_count = other\_language\_preprocessed\_rdd.count()

print("\nNumber of lines in English transcripts after preprocessing:", english\_line\_count)

print("Number of lines in the other language transcripts after preprocessing:", other\_language\_line\_count)

# Stop SparkContext

spark\_context.stop()

A2 Answers:

#A.2.1 Pre-process the text from both RDDs by doing the following:

# ● Lowercase the text

# ● Tokenize the text (split on space)

# Define preprocessing function to run driver application

def preprocess\_text(text):

# Lowercase the text

text = text.lower()

# Tokenize the text (split on space)

tokens = text.split()

return tokens

# A.2.2 Inspect 10 entries from each of your RDDs to verify your pre-processing.

spark\_context = spark\_session.sparkContext

book\_read\_en = spark\_context.textFile('hdfs://192.168.2.250:9000/europarl/europarl-v7.sv-en.en').cache().setName("English Transcripts RDD")

book\_read\_sv = spark\_context.textFile('hdfs://192.168.2.250:9000/europarl/europarl-v7.sv-en.sv').cache().setName("Svenska Transcripts RDD")

# Apply preprocessing function to both RDDs

english\_preprocess\_rdd = book\_read\_en.map(preprocess\_text)

svenska\_preprocess\_rdd = book\_read\_sv.map(preprocess\_text)

# Inspect 10 entries from each RDD

print("English Transcripts - 10 Entries after preprocessing:")

print(english\_preprocess\_rdd.take(10))

print("Svenska Transcripts - 10 Entries after preprocessing:")

print(svenska\_preprocess\_rdd.take(10))

Results: English Transcripts - 10 Entries after preprocessing:

[['resumption', 'of', 'the', 'session'], ['i', 'declare', 'resumed', 'the', 'session', 'of', 'the', 'european', 'parliament', 'adjourned', 'on', 'friday', '17', 'december', '1999,', 'and', 'i', 'would', 'like', 'once', 'again', 'to', 'wish', 'you', 'a', 'happy', 'new', 'year', 'in', 'the', 'hope', 'that', 'you', 'enjoyed', 'a', 'pleasant', 'festive', 'period.'], ['although,', 'as', 'you', 'will', 'have', 'seen,', 'the', 'dreaded', "'millennium", "bug'", 'failed', 'to', 'materialise,', 'still', 'the', 'people', 'in', 'a', 'number', 'of', 'countries', 'suffered', 'a', 'series', 'of', 'natural', 'disasters', 'that', 'truly', 'were', 'dreadful.'], ['you', 'have', 'requested', 'a', 'debate', 'on', 'this', 'subject', 'in', 'the', 'course', 'of', 'the', 'next', 'few', 'days,', 'during', 'this', 'part-session.'], ['in', 'the', 'meantime,', 'i', 'should', 'like', 'to', 'observe', 'a', "minute'", 's', 'silence,', 'as', 'a', 'number', 'of', 'members', 'have', 'requested,', 'on', 'behalf', 'of', 'all', 'the', 'victims', 'concerned,', 'particularly', 'those', 'of', 'the', 'terrible', 'storms,', 'in', 'the', 'various', 'countries', 'of', 'the', 'european', 'union.'], ['please', 'rise,', 'then,', 'for', 'this', "minute'", 's', 'silence.'], ['(the', 'house', 'rose', 'and', 'observed', 'a', "minute'", 's', 'silence)'], ['madam', 'president,', 'on', 'a', 'point', 'of', 'order.'], ['you', 'will', 'be', 'aware', 'from', 'the', 'press', 'and', 'television', 'that', 'there', 'have', 'been', 'a', 'number', 'of', 'bomb', 'explosions', 'and', 'killings', 'in', 'sri', 'lanka.'], ['one', 'of', 'the', 'people', 'assassinated', 'very', 'recently', 'in', 'sri', 'lanka', 'was', 'mr', 'kumar', 'ponnambalam,', 'who', 'had', 'visited', 'the', 'european', 'parliament', 'just', 'a', 'few', 'months', 'ago.']]

Svenska Transcripts - 10 Entries after preprocessing:

[['återupptagande', 'av', 'sessionen'], ['jag', 'förklarar', 'europaparlamentets', 'session', 'återupptagen', 'efter', 'avbrottet', 'den', '17', 'december.', 'jag', 'vill', 'på', 'nytt', 'önska', 'er', 'ett', 'gott', 'nytt', 'år', 'och', 'jag', 'hoppas', 'att', 'ni', 'haft', 'en', 'trevlig', 'semester.'], ['som', 'ni', 'kunnat', 'konstatera', 'ägde', '"den', 'stora', 'år', '2000-buggen"', 'aldrig', 'rum.', 'däremot', 'har', 'invånarna', 'i', 'ett', 'antal', 'av', 'våra', 'medlemsländer', 'drabbats', 'av', 'naturkatastrofer', 'som', 'verkligen', 'varit', 'förskräckliga.'], ['ni', 'har', 'begärt', 'en', 'debatt', 'i', 'ämnet', 'under', 'sammanträdesperiodens', 'kommande', 'dagar.'], ['till', 'dess', 'vill', 'jag', 'att', 'vi,', 'som', 'ett', 'antal', 'kolleger', 'begärt,', 'håller', 'en', 'tyst', 'minut', 'för', 'offren', 'för', 'bl.a.', 'stormarna', 'i', 'de', 'länder', 'i', 'europeiska', 'unionen', 'som', 'drabbats.'], ['jag', 'ber', 'er', 'resa', 'er', 'för', 'en', 'tyst', 'minut.'], ['(parlamentet', 'höll', 'en', 'tyst', 'minut.)'], ['fru', 'talman!', 'det', 'gäller', 'en', 'ordningsfråga.'], ['ni', 'känner', 'till', 'från', 'media', 'att', 'det', 'skett', 'en', 'rad', 'bombexplosioner', 'och', 'mord', 'i', 'sri', 'lanka.'], ['en', 'av', 'de', 'personer', 'som', 'mycket', 'nyligen', 'mördades', 'i', 'sri', 'lanka', 'var', 'kumar', 'ponnambalam,', 'som', 'besökte', 'europaparlamentet', 'för', 'bara', 'några', 'månader', 'sedan.']]

#A.2.3 Verify that the line counts still match after the pre-processing.

#Line count of English Transcripts after preprocess.

english\_prep\_line\_count = english\_preprocess\_rdd.count()

print("After preprocess the number of lines in English transcripts:", english\_prep\_line\_count)

#Line count of Svenska Transcripts after preprocess.

svenska\_prep\_line\_count = svenska\_preprocess\_rdd.count()

print("After preprocess the number of lines in Svenska transcripts:", svenska\_prep\_line\_count)

#Verify both languages line counts are still matching or not.

if english\_prep\_line\_count == svenska\_prep\_line\_count:

print("Line counts are the same for both languages after preprocessing.")

else:

print("Line counts are different for the two languages after preprocessing.")

Results:

After preprocess the number of lines in English transcripts: 1862234

After preprocess the number of lines in Svenska transcripts: 1862234

Line counts are the same for both languages after preprocessing.

#A.3.1 Use Spark to compute the 10 most frequently according words in the English language corpus. Repeat for the other language.

#sorted(book\_read\_en.map(lambda x: x.split(" ")).filter(lambda x: len(x) > 0).flatMap(lambda x: x).filter(lambda x: len(x) > 0).keyBy(lambda x: x[0].lower()).map(lambda x: (x[0], 1)).reduceByKey(add).collect())

en\_words\_sp = book\_read\_en.flatMap(lambda x: x.split(" "))

en\_words\_fl = en\_words\_sp.filter(lambda x: len(x) > 0)

en\_words\_map1 = en\_words\_fl.map(lambda x: x.lower())

en\_words\_map2 = en\_words\_map1.map(lambda x: (x, 1))

en\_words\_rdc = en\_words\_map2.reduceByKey(lambda x, y: x + y)

en\_words\_sorted = en\_words\_rdc.sortBy(lambda x: x[1], ascending=False)

en\_wordsFQ\_10 = en\_words\_sorted.take(10)

print("Top 10 most frequent English Language words are below:")

for key, count in en\_wordsFQ\_10:

print(f"{key} : {count}")

#Other(Svenska) Language top 10 words collection

sv\_words\_sp = book\_read\_sv.flatMap(lambda x: x.split(" "))

sv\_words\_fl = sv\_words\_sp.filter(lambda x: len(x) > 0)

sv\_words\_map1 = sv\_words\_fl.map(lambda x: x.lower())

sv\_words\_map2 = sv\_words\_map1.map(lambda x: (x, 1))

sv\_words\_rdc = sv\_words\_map2.reduceByKey(lambda x, y: x + y)

sv\_words\_sorted = sv\_words\_rdc.sortBy(lambda x: x[1], ascending=False)

sv\_wordsFQ\_10 = sv\_words\_sorted.take(10)

print("Top 10 most frequent English Language words are below:")

for key, count in sv\_wordsFQ\_10:

print(f"{key} : {count}")

Answer:

Top 10 most frequent English Language words are below:

the : 3498375

of : 1659758

to : 1539760

and : 1288401

in : 1085993

that : 797516

a : 773522

is : 758050

for : 534242

we : 522849

Top 10 most frequent Swedish Language words are below:

att : 1706293

och : 1344830

i : 1050774

det : 924866

som : 913276

för : 908680

av : 738068

är : 694381

en : 620310

vi : 539797

#A.3.2 Verify that your results are reasonable.

en\_count\_array = [count for key, count in en\_wordsFQ\_10]

sv\_count\_array = [count for key, count in sv\_wordsFQ\_10]

#matching both array values

matches\_data = (en\_count\_array == sv\_count\_array)

#logical check

if matches\_data:

print("Count of top 10 most frequent words match between English and Swedish.")

else:

print("Count of top 10 most frequent words do not match between English and Swedish.")

Result: Count of top 10 most frequent words do not match between English and Swedish.