**Universiteti i Prishtinës**

**Fakulteti Inxhinierisë Elektrike dhe Kompjuterike**



**Dokumentim teknik i projektit**

**Lënda: Big Data**

**Homework 4: Data Manipulation and Network Analysis**

**Emri profesorit Emri & mbiemri studentëve / email adresa**

| Prof. Vigan Raca | 1. Gyltene Sfishta | gyltene.sfishta[@student.uni-pr.edu](mailto:filan.fisteku@student.uni-pr.edu) |
| --- | --- | --- |
| 2. Klajdi Gashi | klajdi.gashi[@student.uni-pr.edu](mailto:filan.fisteku@student.uni-pr.edu) |
| 3. Kleda Gashi | kleda.gashi[@student.uni-pr.edu](mailto:filan.fisteku@student.uni-pr.edu) |
| 4. Myhedin Vuciterna | myhedin.vuciterna[@student.uni-pr.edu](mailto:filan.fisteku@student.uni-pr.edu) |
|  | 5. Rinesa Hoxha | rinesa.hoxha1[@student.uni-pr.edu](mailto:filan.fisteku@student.uni-pr.edu) |

Prishtinë, 2024

# Permbajtja

[**Abstrakti**](#_30j0zll) **3**

[**I. Hyrje**](#_1fob9te) **4**

[Apache Spark](#_9psfzu37c70o) 4

[Apache Hive](#_rh72ugmoli1u) 4

[PySpark](#_ng8cl7w3ul5i) 4

[Network Analysis (Teoria e Grafëve)](#_j8mkd4q9e475) 5

[Metrikat e Grafeve](#_upg1rvbc19pn) 5

[**II. Qëllimi i punimit**](#_3znysh7) **6**

[**III. Pjesa kryesore**](#_2et92p0) **6**

[**IV. Konkluzione**](#_tyjcwt) **9**

[**Referencat**](#_3dy6vkm) **9**

# **Abstrakti**

Në këtë dokumentim do të përshkruhet mënyra e konfigurimit të frameworks nga Appache per manipulimin me Big Data, si dhe analiza e rrjetave (teoria e grafeve).

Në këtë punim, do të shpjegohet përdorimi i Appache Spark përfshirë edhe komponentet për manipulim me të dhënat.

Njëkohësisht do shpjegohet migrimi i skemës dhe të dhënave nga njëra nga databazat relacionare të përdorura në Homework\_1 duke e konvertuar në platformën Spark, Hive dhe PySpark.

Përmes këtij punimi, synohet të prezantohet një përmbledhje e plotë, duke përfshirë teknologjitë e përdorura, procesin e zhvillimit dhe arritjet gjatë kësaj eksperience.

# **Hyrje**

Në një sistem të madh të të dhënave, zakonisht kemi të bëjmë me sasi të mëdha të dhënash që duhen përpunuar dhe analizuar. Të dhënat mund të jenë të strukturuara, të pa strukturuara ose gjysmë të strukturuara dhe mund të vijnë nga burime të ndryshme si sistemet e menaxhimit të bazave të të dhënave, shërbimet në internet, sensoret, etj. Për të përpunuar këto të dhëna, përdoren teknologji dhe platforma të ndryshme që ndihmojnë në menaxhimin dhe analizën e të dhënave të mëdha (Big Data).

#### Apache Spark

Apache Spark është një platformë e përpunimit të të dhënave të mëdha. Spark mund të përpunojë të dhëna në mënyrë të shpërndarë dhe të paralelizuar, duke ofruar performancë të lartë dhe mundësi për të ekzekutuar algoritme komplekse.

**Implementimi i Apache Spark:**

1. **Instalimi dhe konfigurimi:** Instalimi i Apache Spark është bërë në një kompjuter lokal për zhvillim dhe testim.
2. **Krijimi i aplikacioneve Spark:** Aplikacionet në Spark mund të shkruhen në gjuhë të ndryshme si Scala, Java, Python dhe R.
3. **Ekzekutimi i punëve:** Punët ekzekutohen në Spark Cluster, ku Spark shfrytëzon burimet e sistemit për të përpunuar të dhënat.

#### Apache Hive

Apache Hive është një sistem i magazinimit të të dhënave dhe pyetjeve që lejon analizën e të dhënave të mëdha të ruajtura në HDFS (Hadoop Distributed File System) duke përdorur një sintaksë të ngjashme me SQL.

**Implementimi i Apache Hive:**

1. **Instalimi dhe konfigurimi:** Instalimi i Hive përfshin konfigurimin e Hive Metastore dhe lidhjen e tij me një bazë të dhënash.
2. **Krijimi i tabelave Hive:** Tabelat krijohen duke përdorur HiveQL (SQL për Hive).
3. **Ekzekutimi i query:** Ekzekutohen në Hive për të marrë informacione nga të dhënat e mëdha të ruajtura në HDFS.

#### PySpark

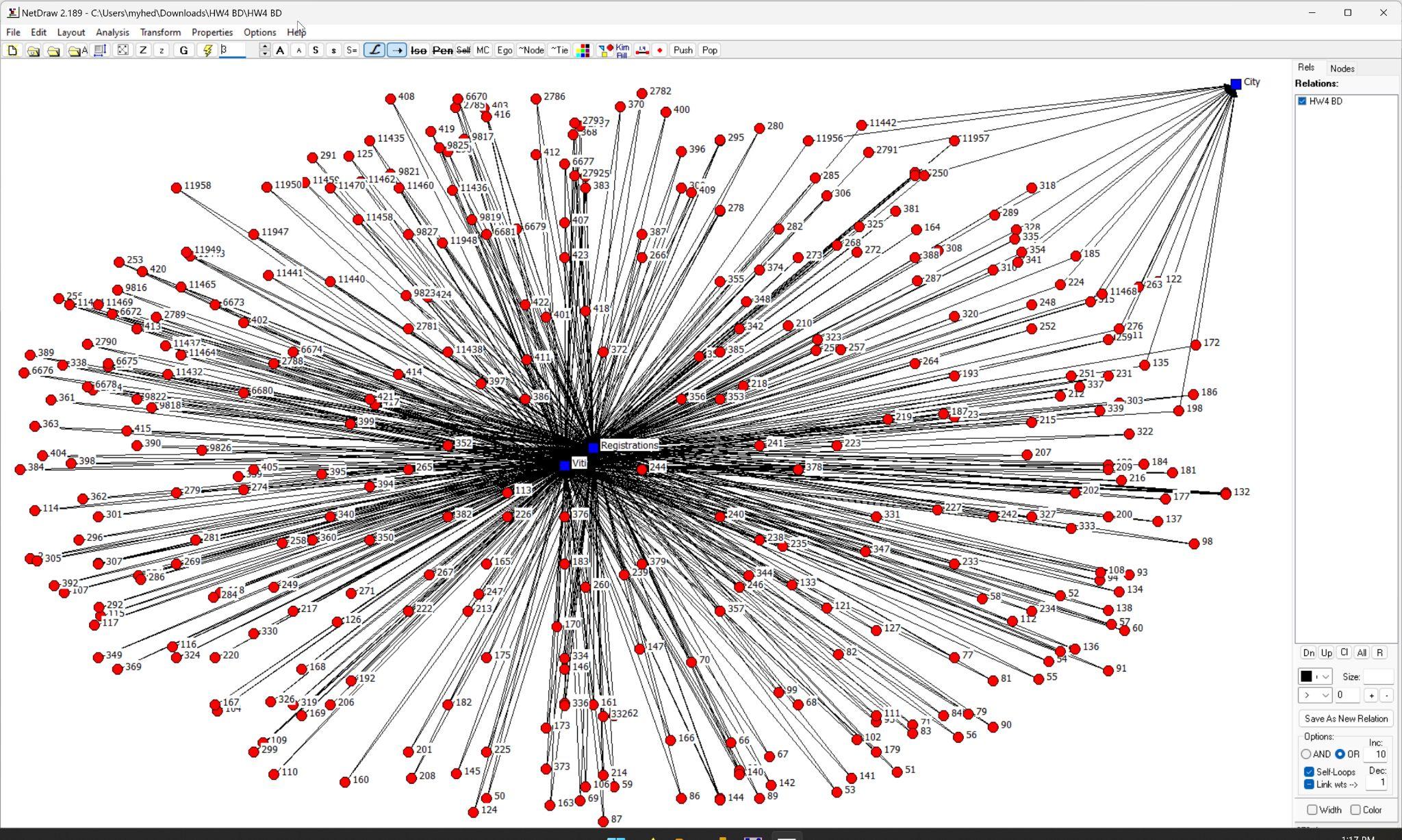
PySpark është API Python për Apache Spark, e cila lejon përdorimin e Spark nga programet Python. PySpark kombinon fuqinë e Spark me lehtësinë e përdorimit të Python.

#### Network Analysis (Teoria e Grafëve)

Analiza e rrjetit (Network Analysis) përdor teorinë e grafëve për të analizuar dhe modeluar struktura komplekse të lidhjeve midis njësive (në grafë, njësitë quhen nyje dhe lidhjet quhen brinjë).

**Implementimi i Network Analysis:**

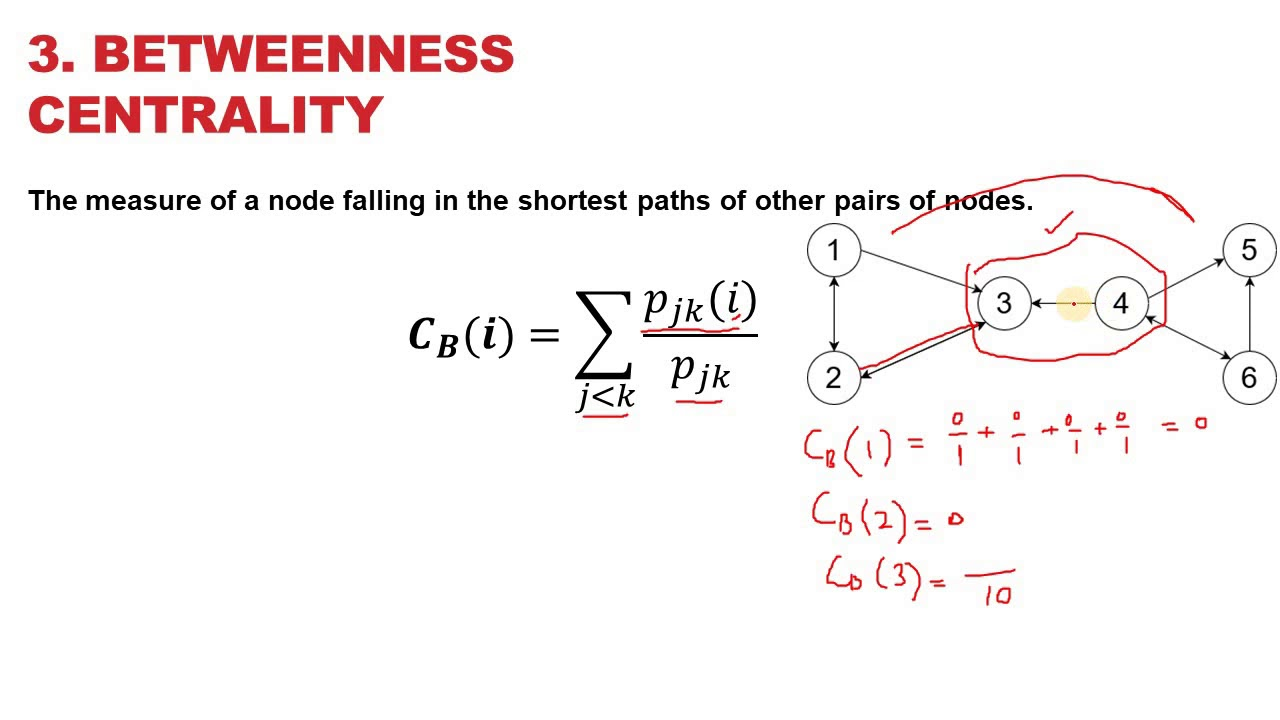
1. **Modelimi i të dhënave si grafë:** Të dhënat modelohen si grafe, ku çdo njësi përfaqësohet nga një nyje dhe çdo lidhje përfaqësohet nga një brinjë.
2. **Analiza e metrikave të grafëve:** Duke përdorur metrika të ndryshme, analizohet struktura dhe karakteristikat e grafit.



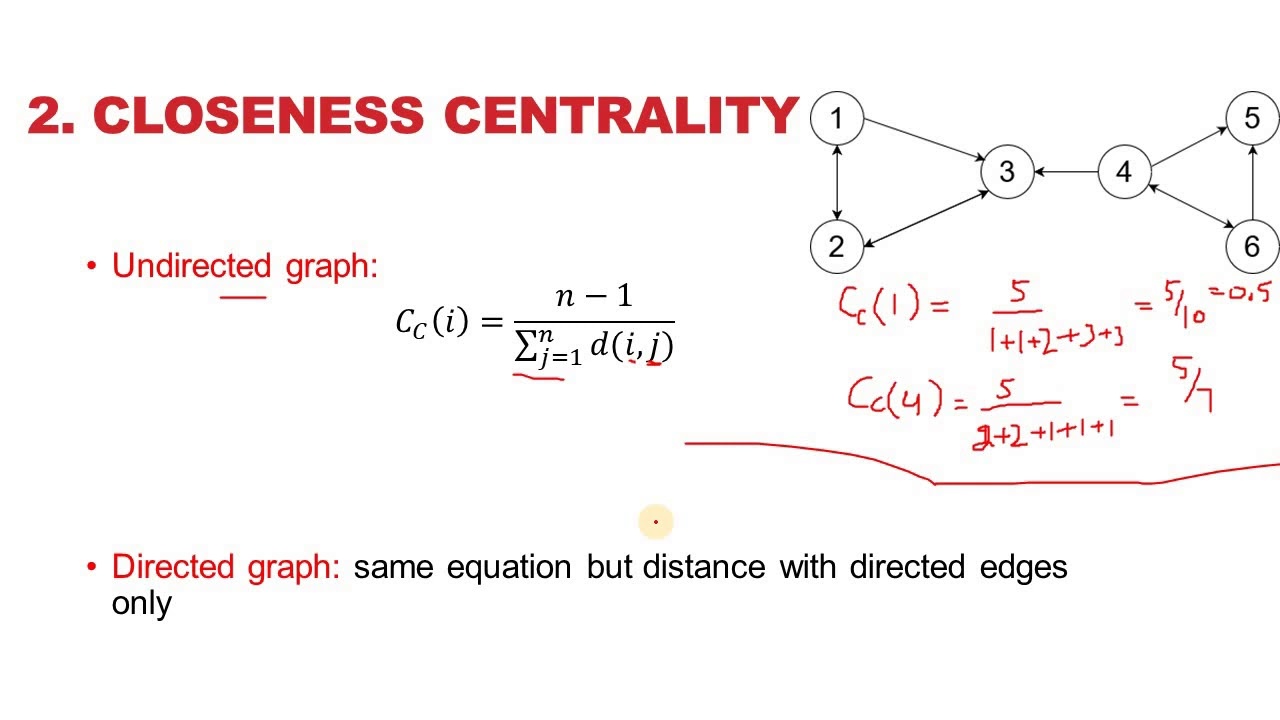
*Fig 1. Layout nga teoria e grafeve*

##### Metrikat e Grafeve

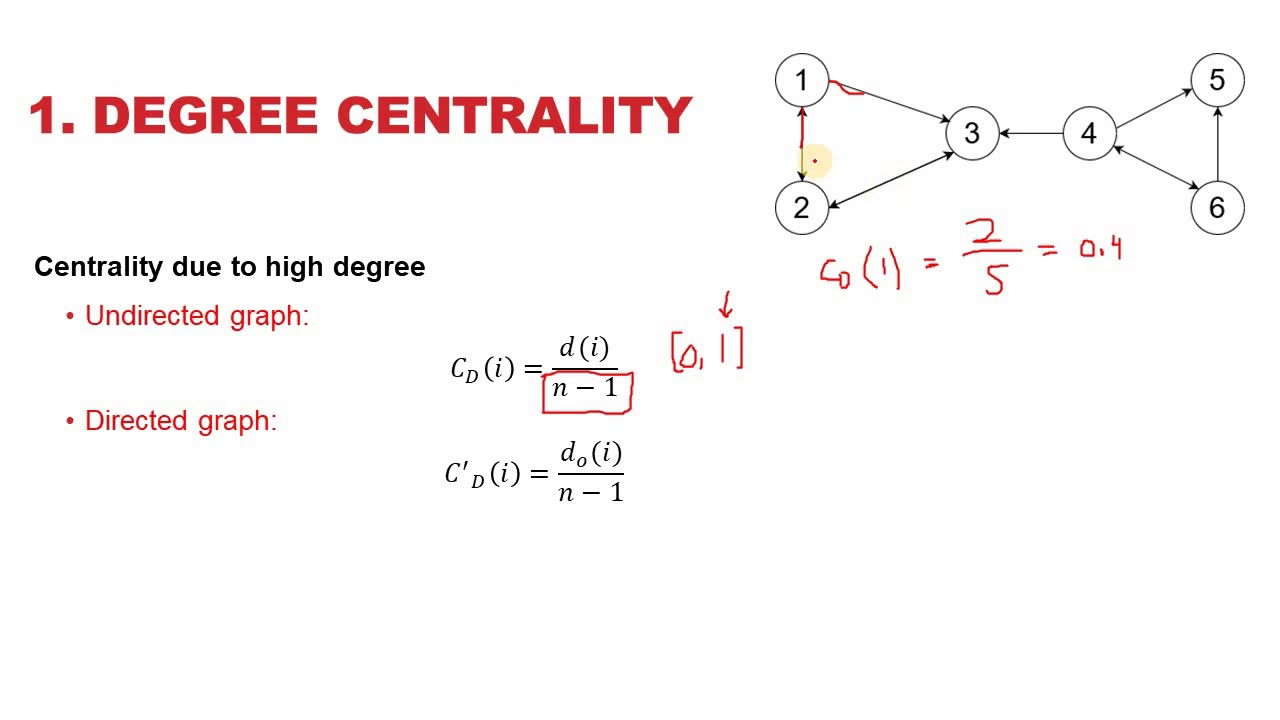
1. **Betweenness Centrality (Qendra e Ndërmjetësimit):**
   * Mat sa shpesh një nyje ndodhet në rrugët më të shkurtra midis dy nyjeve të tjera.
   * Formula për betweenness centrality:

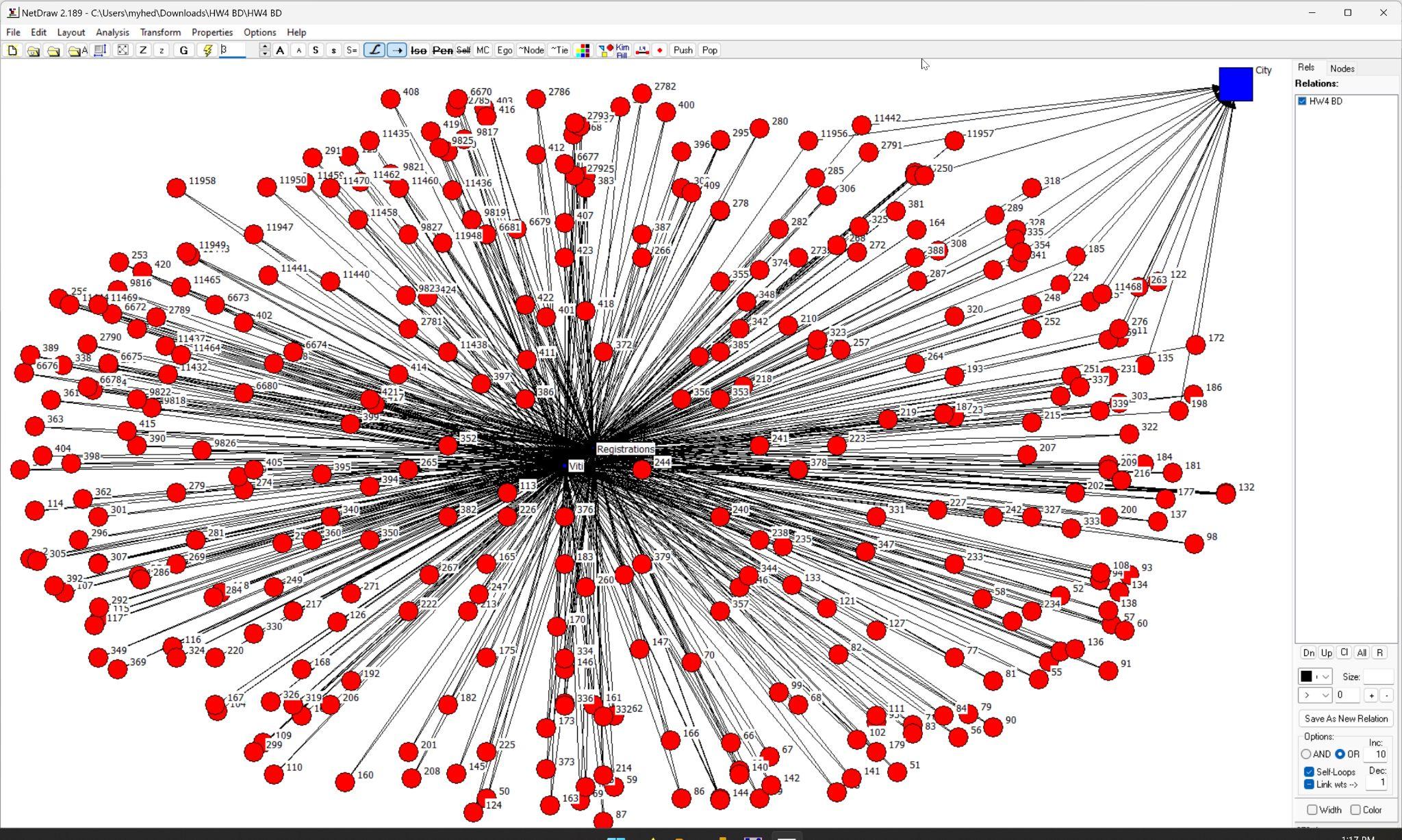


1. **Closeness Centrality (Qendra e Afërsisë):**
   * Mat afërsinë e një nyje me të gjitha nyjet e tjera në graf.
   * Formula për closeness centrality:



1. **Degree Centrality (Qendra e Gradës):**
   * Mat numrin e lidhjeve të një nyje.
   * Formula për degree centrality:





*Fig 2. Nyjet pas matjeve te centralitetit*

# **Qëllimi i punimit**

Qëllimi i punimit ka qenë aplikimi i njohurive rreth konfigurimit të bazave të ndryshme të të dhënave si dhe manipulimi me ato të dhëna në raste kur janë specifikuar kërkesa paraprake.

Teknologjitë e përdorura për arritjen e rezultateve të kërkuara janë **Appache Spark**, **Hive dhe PySpark.**

# **Pjesa kryesore**

Fillimisht janë ngarkuar disa files CSV në DataFrame të Spark, të krijohet një pamje e përkohshme për këto DataFrame dhe të ekzekutohen query-të SQL mbi këto pamje.

1. **Inicimi i Spark Session:**

Ky hap krijon një sesion Spark që do të përdoret për të ekzekutuar operacionet Spark si dhe është aktivizuar mbështetja për Hive.

spark = SparkSession.builder \

.appName("Mondial Dataset Import") \

.enableHiveSupport() \

.getOrCreate()

1. **Ngarkimi i skedarëve CSV në DataFrame:**

Ky hap ngarkon files CSV të specifikuar ("river.csv", "geo\_river.csv", "country.csv", "ismember.csv").

river\_df = spark.read.csv("river.csv", header=True, inferSchema=True)

geo\_river\_df = spark.read.csv("geo\_river.csv", header=True, inferSchema=True)

country\_df = spark.read.csv("country.csv", header=True, inferSchema=True)

ismember\_df = spark.read.csv("ismember.csv", header=True, inferSchema=True)

1. **Krijimi i pamjeve të përkohshme për query SQL:**

Ky hap krijon pamje të përkohshme për secilin DataFrame, të cilat mund të përdoren në pyetjet SQL të ekzekutuara përmes Spark SQL.

river\_df.createOrReplaceTempView("river")

geo\_river\_df.createOrReplaceTempView("geo\_river")

country\_df.createOrReplaceTempView("country")

ismember\_df.createOrReplaceTempView("ismember")

1. **Përkufizimi i query SQL:**

query = """

SELECT DISTINCT r.name AS river\_name, c.name as country,

(SELECT COUNT(\*) FROM geo\_river WHERE country = C.code) as count

FROM river r

JOIN geo\_river gr ON r.name = gr.river

JOIN country C ON GR.country = C.code

LEFT JOIN (

SELECT country

FROM ismember

WHERE organization = 'NATO'

) M ON C.code = M.country

WHERE M.country IS NULL

AND R.sea IS NOT NULL

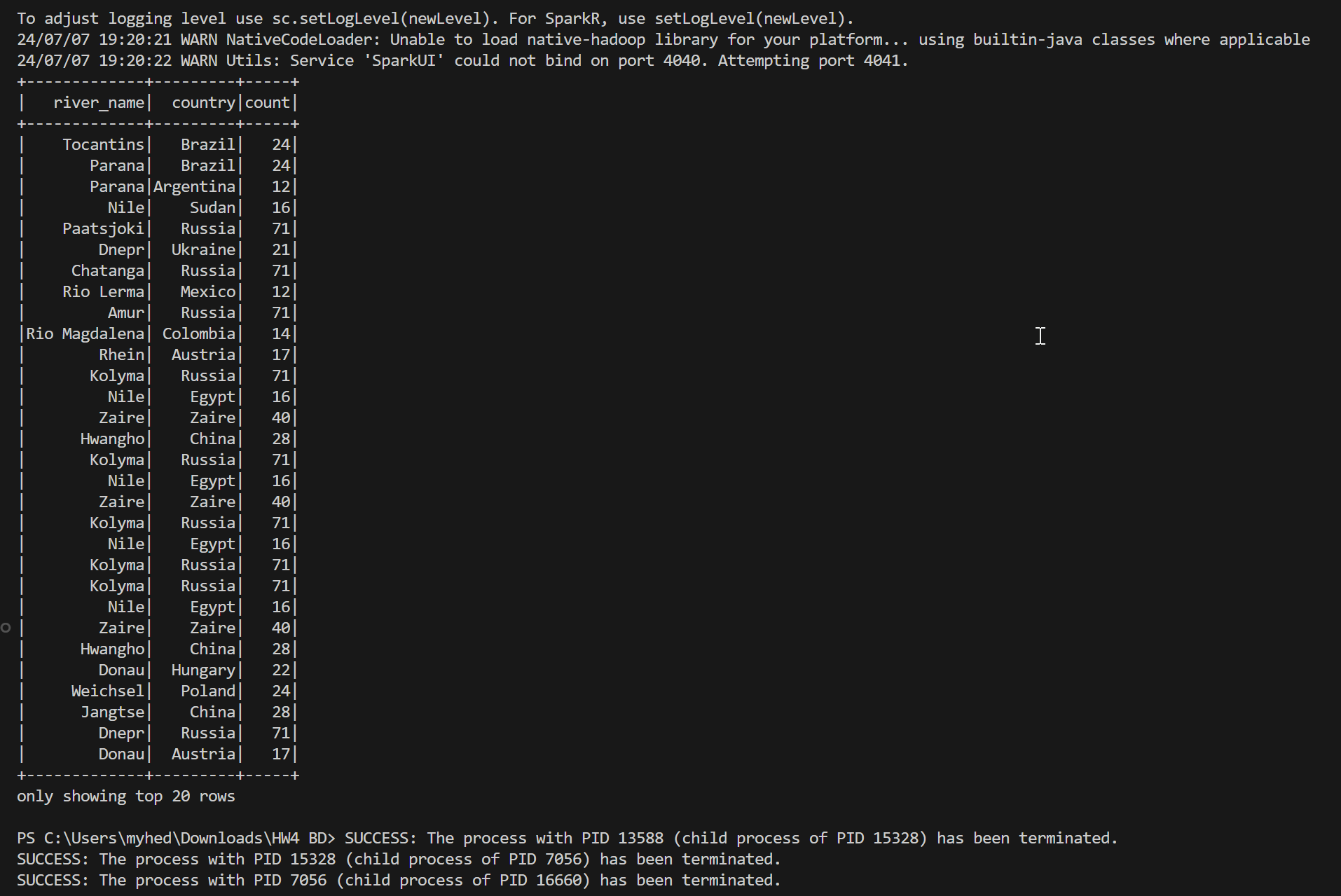
AND (

SELECT COUNT(\*) FROM geo\_river WHERE country = C.code

) > 10;

"""

Pastaj është bërë ekzekutimi i query-ve SQL dhe tregimi i rezultateve.

****

*Fig 3. Ekzekutimi i tabelave te Mondial nga HW1*

Në vazhdim, është ngarkuar një dataset nga një file CSV në një DataFrame të Spark dhe janë kryejr disa operacione bazike mbi të për të treguar shembuj të përdorimit të Spark për përpunimin e të dhënave. Këtu janë hapat që ndjek kodi dhe qëllimi i secilit hap:

1. **Inicimi i Spark Session**

Ky hap krijon një sesion Spark që do të përdoret për të ekzekutuar operacionet Spark.

spark = SparkSession.builder \

.appName("Large Dataset Import") \

.enableHiveSupport() \

.getOrCreate()

1. **Përcaktimi i shtegut të dataset-it:**

Ky hap ruan shtegun e skedarit CSV që përmban dataset-in e madh që do të ngarkohet.

dataset\_path = "new\_retail\_data.csv"

1. **Ngarkimi i dataset-it në një Spark DataFrame:**

Ky hap ngarkon dataset-in nga skedari CSV në një DataFrame të Spark. Parametri header=True tregon që rreshti i parë i CSV-së përmban emrat e kolonave dhe inferSchema=True tregon që Spark duhet të supozojë tipet e të dhënave bazuar në vlerat në dataset.

large\_dataset\_df = spark.read.csv(dataset\_path, header=True, inferSchema=True)

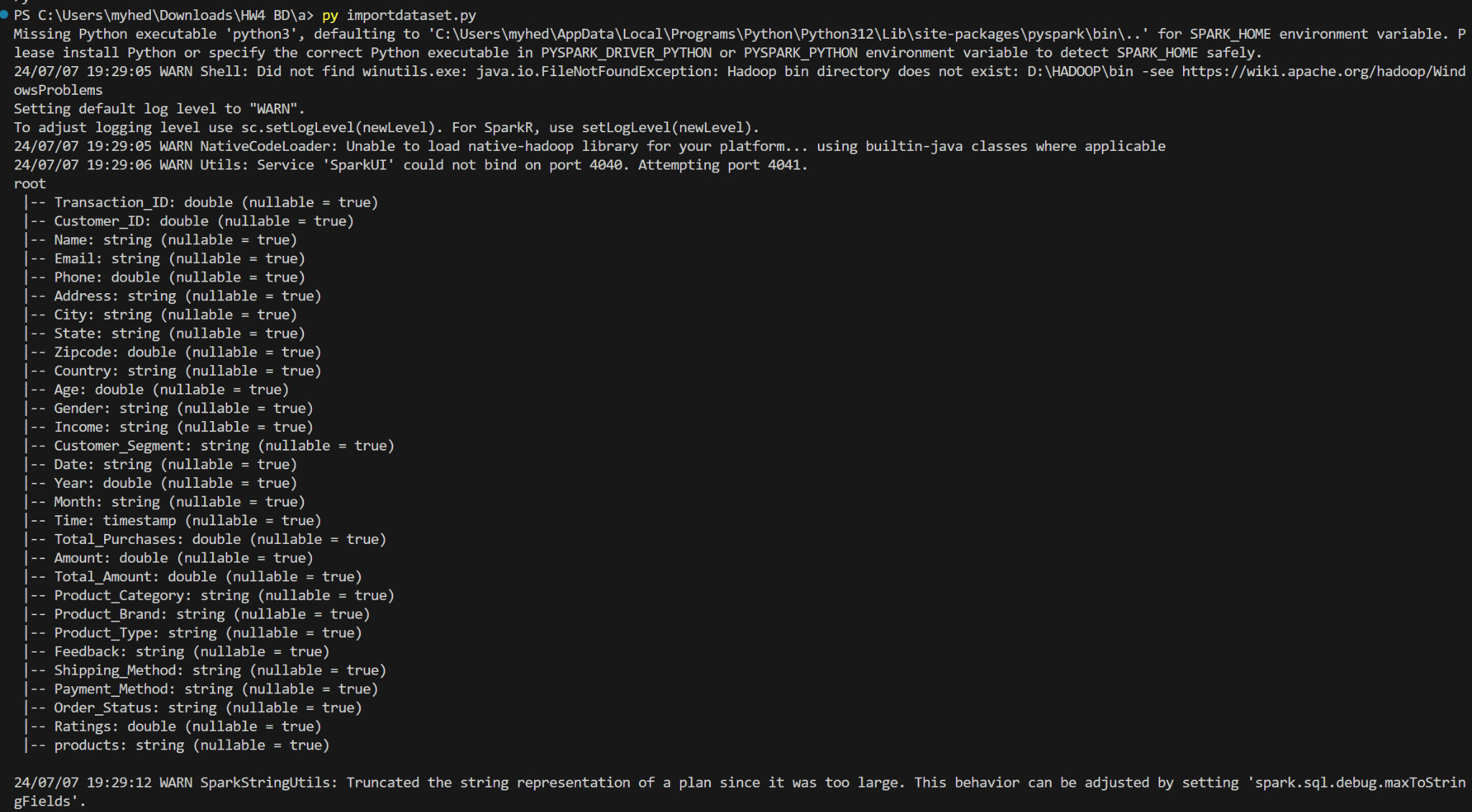
1. **Tregimi i skemës së DataFrame-it:**

Ky hap tregon skemën (strukturën) e DataFrame-it, duke shfaqur emrat e kolonave dhe tipet e të dhënave. Pastaj shfaq 5 rreshtat e parë të dataset-it për të dhënë një përmbledhje të përmbajtjes së tij si dhe numëron dhe shfaq numrin total të rreshtave në dataset. Në fund kryhet një përshkrim statistikor i dataset-it, duke shfaqur statistika si mesatarja, devijimi standard, minimumi dhe maksimumi për secilën kolonë.

large\_dataset\_df.printSchema()  
large\_dataset\_df.show(5)  
row\_count = large\_dataset\_df.count()

print(f"Total number of rows: {row\_count}")  
large\_dataset\_df.describe().show()  
spark.stop()

Në përmbledhje, ky kod ngarkon një dataset të madh nga një file CSV, tregon strukturën dhe disa vlera të dataset-it, kryen disa operacione bazike analitike dhe më pas ndërpret sesionin Spark.



*Fig 4. Ngarkimi i Dataset-it*

# **Konkluzione**

Pas përfundimit të këtij projekti dhe pas përdorimit të një sërë metodash për zhvillimin e tij, janë arritur disa perfundime të rëndësishme. Përdorimi i një game të gjerë metodash ka kontribuar në pasurimin e përmbajtjes së projektit dhe në sigurimin e një analize të tij.

Përdorimi i metodave hulumtuese dhe shpjeguese ka lejuar ekipin të kuptojë thellësisht konceptet dhe teknologjitë që janë përdorur në projekt.

# **Referencat**

Të gjitha informatat e përdorura në këtë dokumentim janë autentike dhe bazuar në ligjërata dhe ushtrime.

* [Appache Spark](https://spark.apache.org/)
* [Hive](https://hive.apache.org/)