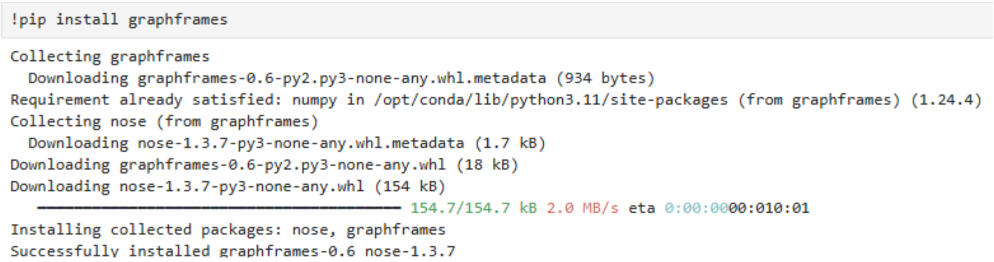
NAMA : SUKMA BAGUS WAHASDWIKA
NIM : 2241720223
KELAS : TI - 3D

BIG DATA – 11 Graph Processing

Install graphframes

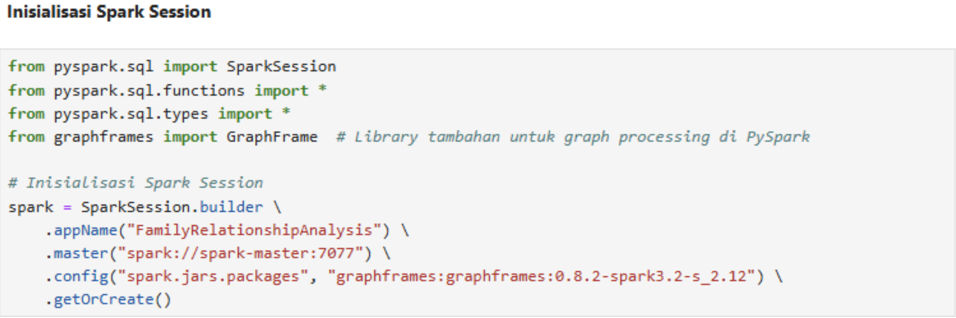


```
!pip install graphframes
```

```
Collecting graphframes
  Downloading graphframes-0.6-py2.py3-none-any.whl.metadata (934 bytes)
Requirement already satisfied: numpy in /opt/conda/lib/python3.11/site-packages (from graphframes) (1.24.4)
Collecting nose (from graphframes)
  Downloading nose-1.3.7-py3-none-any.whl.metadata (1.7 kB)
Downloading graphframes-0.6-py2.py3-none-any.whl (18 kB)
Downloading nose-1.3.7-py3-none-any.whl (154 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 154.7/154.7 kB 2.0 MB/s eta 0:00:0000:010:01
Installing collected packages: nose, graphframes
Successfully installed graphframes-0.6 nose-1.3.7
```

Praktikum graphframes

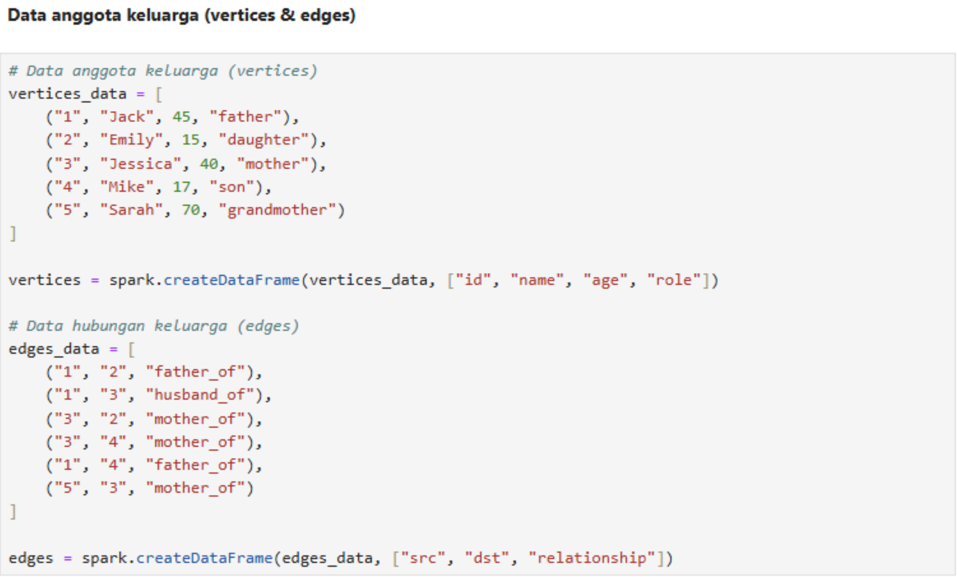
1. Inisialisasi Spark Session



Inisialisasi Spark Session

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import *
from pyspark.sql.types import *
from graphframes import GraphFrame  # Library tambahan untuk graph processing di PySpark

# Inisialisasi Spark Session
spark = SparkSession.builder \
    .appName("FamilyRelationshipAnalysis") \
    .master("spark://spark-master:7077") \
    .config("spark.jars.packages", "graphframes:graphframes:0.8.2-spark3.2-s_2.12") \
    .getOrCreate()
```

2. Data Anggota Keluarga (Vertices & Edges)



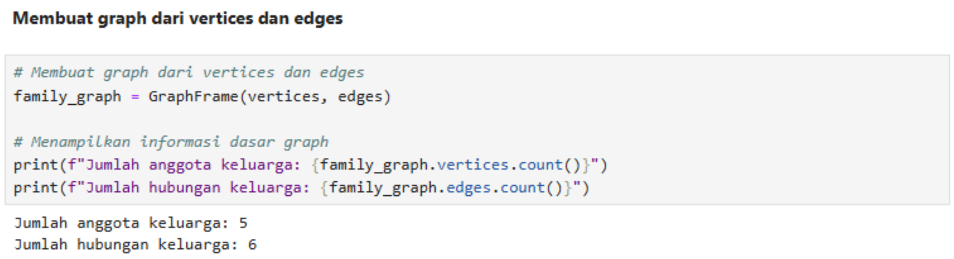
Data anggota keluarga (vertices & edges)

```
# Data anggota keluarga (vertices)
vertices_data = [
    ("1", "Jack", 45, "father"),
    ("2", "Emily", 15, "daughter"),
    ("3", "Jessica", 40, "mother"),
    ("4", "Mike", 17, "son"),
    ("5", "Sarah", 70, "grandmother")
]

vertices = spark.createDataFrame(vertices_data, ["id", "name", "age", "role"])

# Data hubungan keluarga (edges)
edges_data = [
    ("1", "2", "father_of"),
    ("1", "3", "husband_of"),
    ("3", "2", "mother_of"),
    ("3", "4", "mother_of"),
    ("1", "4", "father_of"),
    ("5", "3", "mother_of")
]

edges = spark.createDataFrame(edges_data, ["src", "dst", "relationship"])
```

NAMA : SUKMA BAGUS WAHASDWIKA
NIM : 2241720223
KELAS : TI - 3D

BIG DATA – 11 Graph Processing

3. Membuah Graph dari Vertices dan Edges

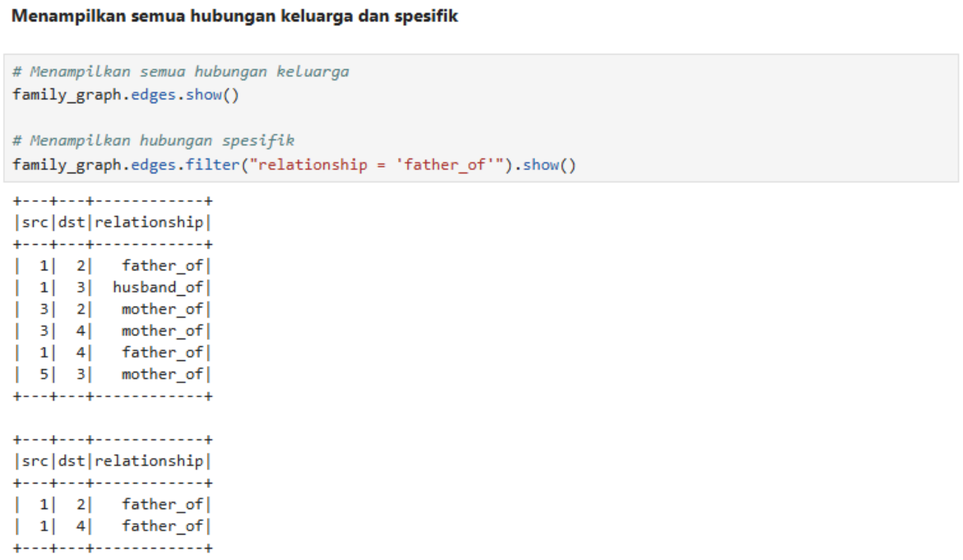


Membuat graph dari vertices dan edges

```
# Membuat graph dari vertices dan edges
family_graph = GraphFrame(vertices, edges)

# Menampilkan informasi dasar graph
print(f"Jumlah anggota keluarga: {family_graph.vertices.count()}")
print(f"Jumlah hubungan keluarga: {family_graph.edges.count()}")
```

```
Jumlah anggota keluarga: 5
Jumlah hubungan keluarga: 6
```

4. Menampilkan Semua Hubungan Keluarga dan Hubungan Spesifik

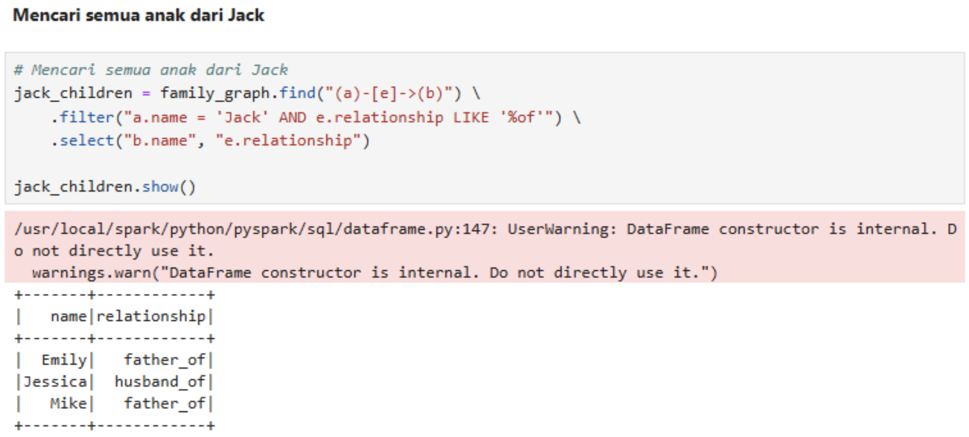


Menampilkan semua hubungan keluarga dan spesifik

```
# Menampilkan semua hubungan keluarga
family_graph.edges.show()

# Menampilkan hubungan spesifik
family_graph.edges.filter("relationship = 'father_of'").show()
```

```
+---+---+------------+
|src|dst|relationship|
+---+---+------------+
|  1|  2|   father_of|
|  1|  3|  husband_of|
|  3|  2|   mother_of|
|  3|  4|   mother_of|
|  1|  4|   father_of|
|  5|  3|   mother_of|
+---+---+------------+

+---+---+------------+
|src|dst|relationship|
+---+---+------------+
|  1|  2|   father_of|
|  1|  4|   father_of|
+---+---+------------+
```

5. Mencari Semua Anak dari Jack



Mencari semua anak dari Jack

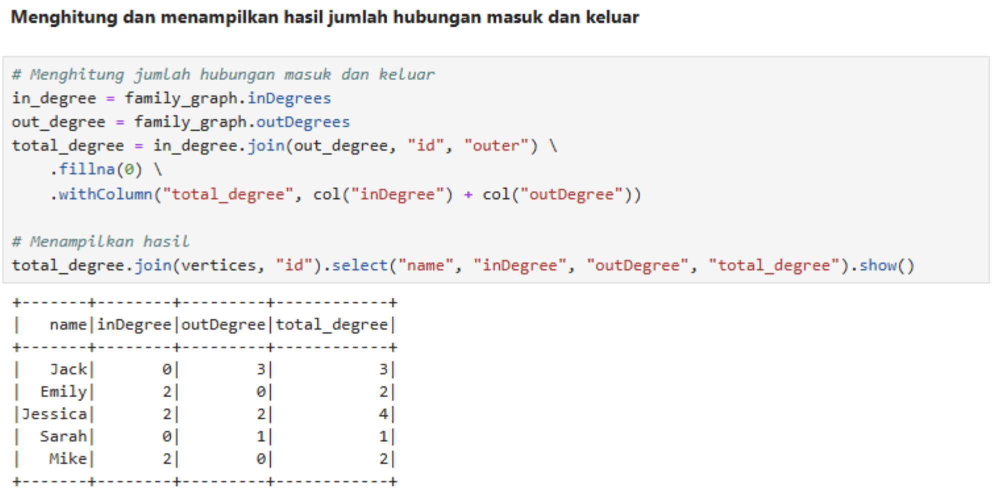
```
# Mencari semua anak dari Jack
jack_children = family_graph.find("(a)-[e]->(b)") \
    .filter("a.name = 'Jack' AND e.relationship LIKE '%of'") \
    .select("b.name", "e.relationship")

jack_children.show()
```

```
/usr/local/spark/python/pyspark/sql/dataframe.py:147: UserWarning: DataFrame constructor is internal. D
o not directly use it.
  warnings.warn("DataFrame constructor is internal. Do not directly use it.")
+-------+------------+
|   name|relationship|
+-------+------------+
|  Emily|   father_of|
|Jessica|  husband_of|
|   Mike|   father_of|
+-------+------------+
```

NAMA : SUKMA BAGUS WAHASDWIKA
NIM : 2241720223
KELAS : TI - 3D

BIG DATA – 11 Graph Processing

6. Menghitung Dan Menampilkan Hasil Jumlah Hubungan Masukan Keluar

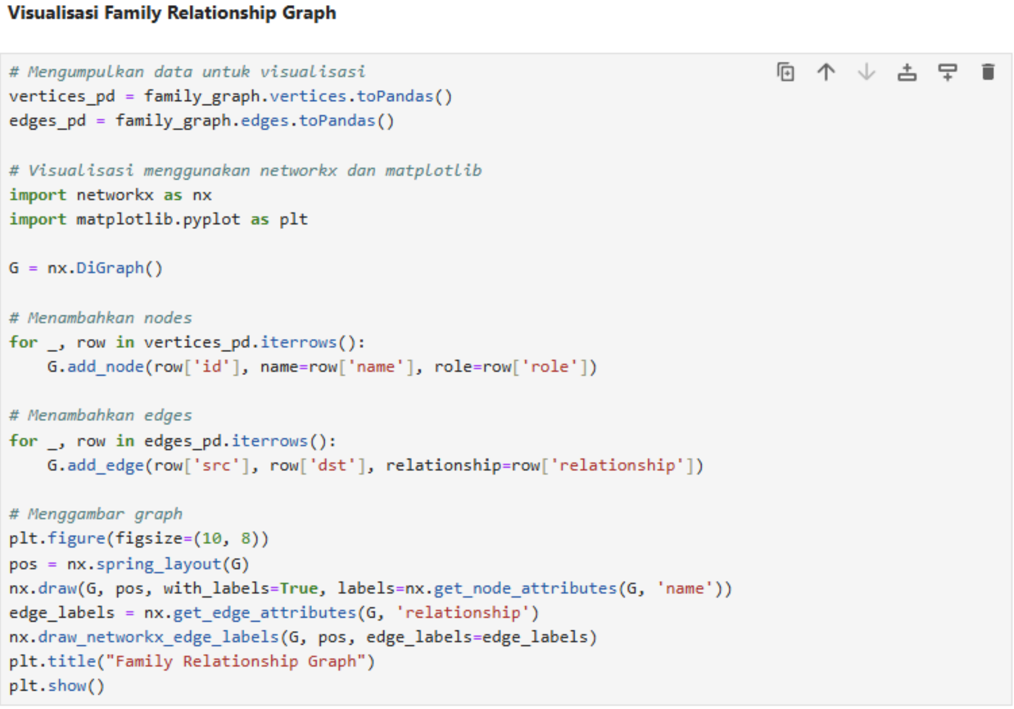


Menghitung dan menampilkan hasil jumlah hubungan masuk dan keluar

```
# Menghitung jumlah hubungan masuk dan keluar
in_degree = family_graph.inDegrees
out_degree = family_graph.outDegrees
total_degree = in_degree.join(out_degree, "id", "outer") \
    .fillna(0) \
    .withColumn("total_degree", col("inDegree") + col("outDegree"))

# Menampilkan hasil
total_degree.join(vertices, "id").select("name", "inDegree", "outDegree", "total_degree").show()
```

```
+-------+--------+---------+------------+
|   name|inDegree|outDegree|total_degree|
+-------+--------+---------+------------+
|   Jack|       0|        3|           3|
|  Emily|       2|        0|           2|
|Jessica|       2|        2|           4|
|  Sarah|       0|        1|           1|
|   Mike|       2|        0|           2|
+-------+--------+---------+------------+
```

7. Visualisasi Family Relationship Graph



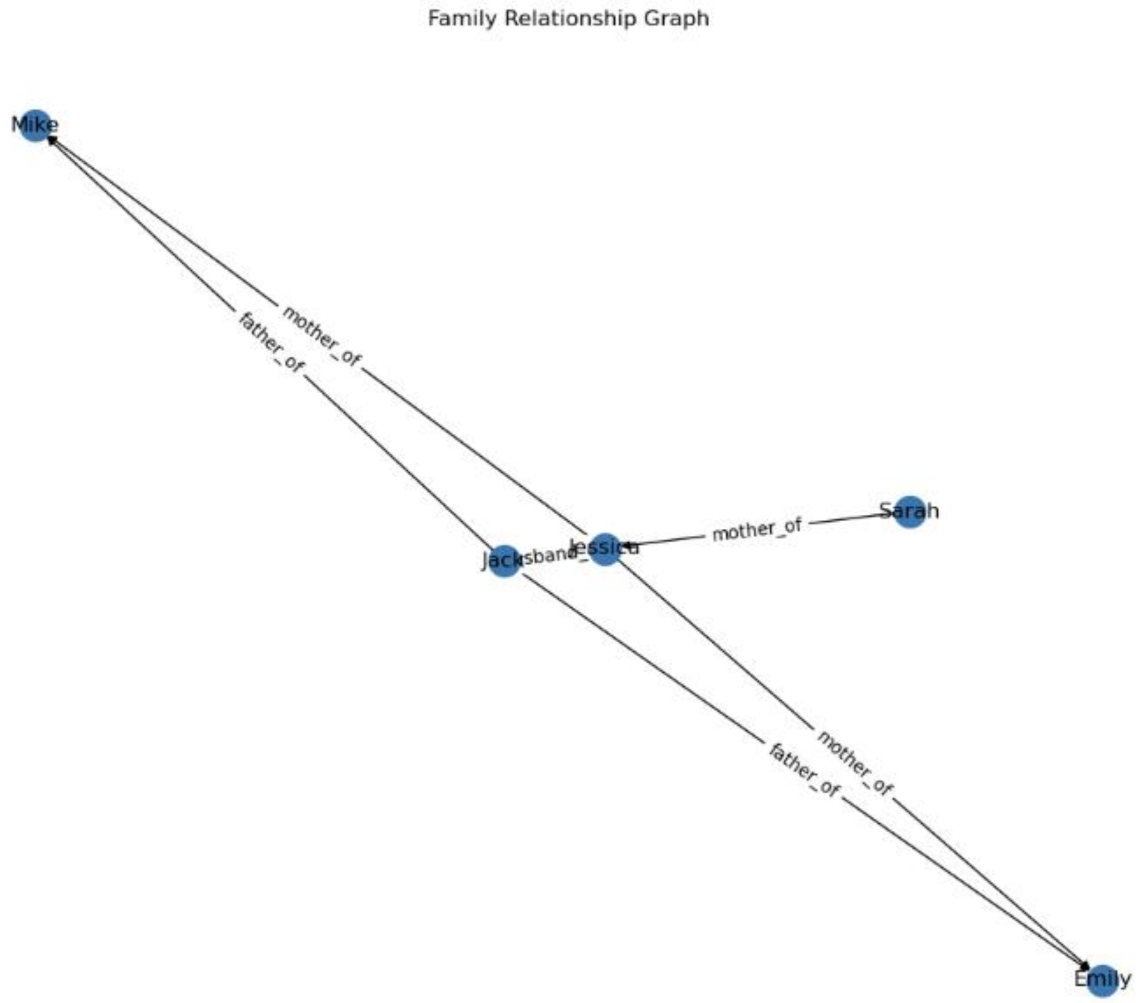
Visualisasi Family Relationship Graph

```
# Mengumpulkan data untuk visualisasi
vertices_pd = family_graph.vertices.toPandas()
edges_pd = family_graph.edges.toPandas()

# Visualisasi menggunakan networkx dan matplotlib
import networkx as nx
import matplotlib.pyplot as plt

G = nx.DiGraph()

# Menambahkan nodes
for _, row in vertices_pd.iterrows():
    G.add_node(row['id'], name=row['name'], role=row['role'])

# Menambahkan edges
for _, row in edges_pd.iterrows():
    G.add_edge(row['src'], row['dst'], relationship=row['relationship'])

# Menggambar graph
plt.figure(figsize=(10, 8))
pos = nx.spring_layout(G)
nx.draw(G, pos, with_labels=True, labels=nx.get_node_attributes(G, 'name'))
edge_labels = nx.get_edge_attributes(G, 'relationship')
nx.draw_networkx_edge_labels(G, pos, edge_labels=edge_labels)
plt.title("Family Relationship Graph")
plt.show()
```

NAMA : SUKMA BAGUS WAHASDWIKA
NIM : 2241720223
KELAS : TI - 3D

BIG DATA – 11 Graph Processing

Output Visualisasi Graph:


Family Relationship Graph
Mike
mother_of
father_of
Sarah
mother_of
Jack
sband_
Jessica
mother_of
father_of
Emily