ADVENTIST UNIVERSITY OF CENTRAL AFRICA

Msc in BIG DATA ANALYTICS

COURSE: BIG DATA ANALYTICS

Assignment (Project 2)

Project Name:

US Road Network Analysis Using Graph Database (Neo4j)

STUDENT IDENTIFICATION

NKURIKIYUMUKIZA Adrien | 101088

MUKANDAYISENGA Marie Josee | 101073

MPANO RUDAKENGA Rongin | 101032

Lecturer: **Temitope Oguntade**

Submitted on January 19, 2026

Table of Contents

1. Introduction

This project analyzes the US Road Network dataset using Neo4j, a graph database well suited for modeling and analyzing highly connected data. The road network is represented as a graph where:

- **Nodes (vertices)** represent road intersections
- **Relationships (edges)** represent roads between intersections
- Edge weights represent Euclidean distances between intersections

The full dataset contains **87,575 intersections** and **121,961 roads**, making it a large, sparse graph with an average degree of approximately **2.8**. This structure is ideal for demonstrating graph analytics concepts such as shortest paths, degree analysis, and centrality measures.

Using Neo4j and the Cypher query language, this project performs network size analysis, shortest path queries, degree-based connectivity analysis, and betweenness centrality evaluation. Visualization techniques are also applied to support interpretation and presentation of results. The project demonstrates how graph analytics can be effectively applied to large transportation datasets to support infrastructure planning and traffic optimization.

1.1. Problem Statement

Large-scale road networks generate complex, highly connected data that is difficult to analyze efficiently using traditional relational databases. Operations such as shortest path computation, identification of highly connected intersections, and detection of critical network nodes require extensive joins and high computational cost.

There is therefore a need for a **graph-based analytical approach** that can efficiently model and analyze road networks, enabling scalable analysis of connectivity, routing, and network importance within a Big Data context.

1.2. Project Objectives

The objectives of this project are to:

- Model the US road network using a graph database (Neo4j)
- Determine the total number of intersections and roads
- Compute shortest paths between selected intersections
- Identify highly connected intersections using degree analysis
- Analyze network importance using betweenness centrality
- Visualize key network metrics using charts and dashboards

1.3. Proposed Solution

The problem is addressed by modeling the US road network as a graph in Neo4j, where intersections are represented as nodes and roads as relationships. Cypher queries and graph algorithms are applied to analyze network size, connectivity, shortest paths, and centrality. Visualization tools such as Plotly are used to present insights clearly and support decision-making.

1.4. Data Model in Neo4j

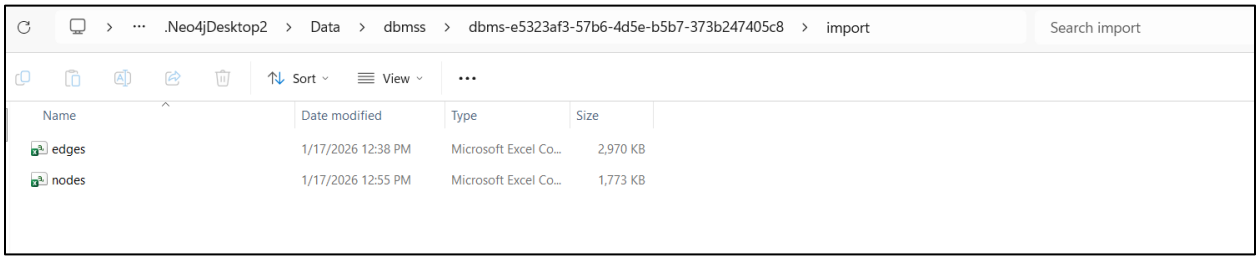
The dataset was modeled in Neo4j as follows:

- **Node label:** Intersection
  - ✓ Properties: **id, x, y**
- **Relationship type:** ROAD
  - ✓ Property: **distance**

2. Implementation of the Project

2.1. Prepare the dataset

Let's place the CSV files (**nodes.csv and edges.csv**) inside **Neo4j's import folder** to ensure that Neo4j can access them using **file:///filename.csv.** Instance name created **is USRoadNetwork and is usroadnetworkdb**



2.1. Create a Uniqueness Constraint

This above constraint enforces the uniqueness of each intersection ID in the graph, preventing duplicate nodes even if repeated records exist in the dataset.

2.1. Load intersections (nodes)

The intersections file is read row by row, cleaned to avoid duplicate IDs, and merged to create each intersection only once. As a result, **87,575** unique (**:Intersection**) nodes were created, each uniquely identified by its id and assigned x and y coordinate2.2. Import RAW Roads (Keep All Records)

2.2 Import RAW Roads (Keep All Records)

Each CSV row is mapped to a relationship, resulting in **121,961 road relationships**, including duplicates and reversed edges.

2.2. Load roads (relationships) and Intersetions

Q1. Find the total number of intersections and roads

```
// Verify total Intersections and Roads
MATCH (n:Intersection)
WITH count(n) AS totalIntersections
MATCH ()-[r:RAW_ROAD]->()
RETURN totalIntersections, count(r) AS totalRoads;
```

Table RAW

totalIntersections	totalRoads
87575	121961

Started streaming 1 record after 175 ms and completed after 464 ms.

The result shows that the graph contains **87,575** unique intersection nodes and **121,961** road relationships, confirming that all raw road records (including duplicates and reversed edges) were loaded while intersections remain uniquely represented.

Q1.1. Verify Average Degree

```
//Verify Average Degree
MATCH (i:Intersection)
RETURN avg(COUNT { (i)-[:ROAD]-() }) AS avgDegree;
```

Table RAW

avgDegree
2.774558949471861

The computed average degree is **approximately 2.8**, which means that, on average, each intersection is connected to about three roads. This indicates that the road network is sparse, aligning with the structural characteristics expected for a large-scale road network.

2.3. Import CLEAN Roads (Deduplicated Network)

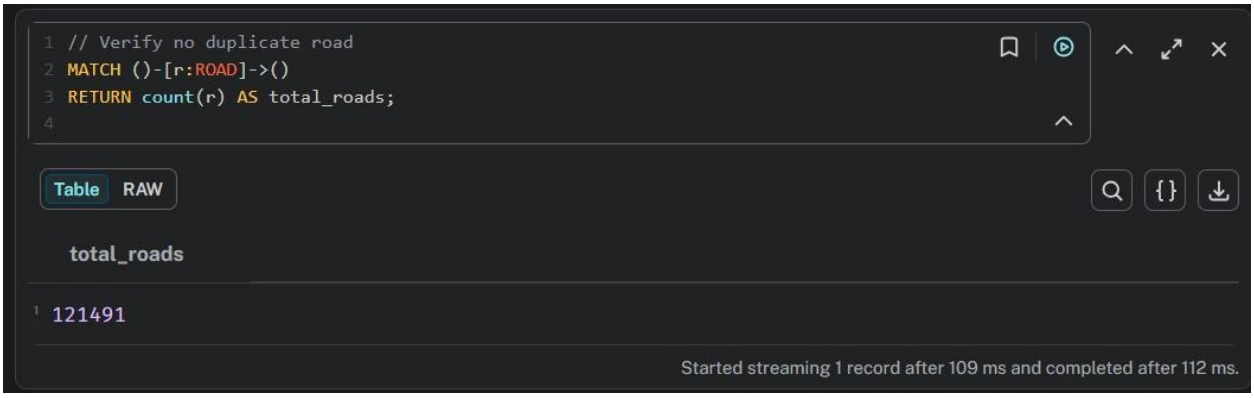
```
//Import CLEAN Roads (Deduplicated Network)
LOAD CSV WITH HEADERS FROM 'file:///roads.csv' AS row
WITH toInteger(row.source) AS s,
     toInteger(row.target) AS t,
     toFloat(row.distance) AS dist
WITH CASE WHEN s < t THEN s ELSE t END AS a,
     CASE WHEN s < t THEN t ELSE s END AS b,
     dist
MATCH (i1:Intersection {id:a})
MATCH (i2:Intersection {id:b})
MERGE (i1)-[r:ROAD]-(i2)
SET r.distance = dist;
```

Created 121,491 relationships, set 121,961 properties

Completed after 48,644 ms

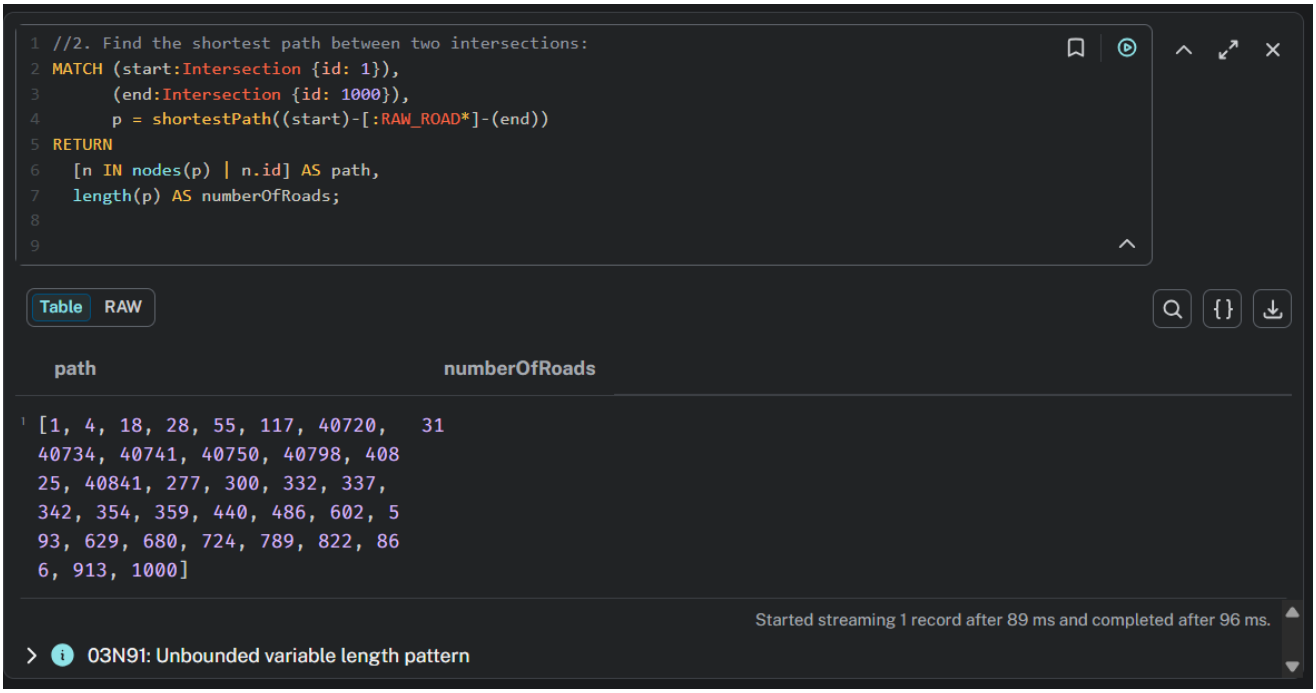
The process normalizes road endpoints and merges duplicate edges so that each physical road is represented only once, resulting in **121,491 unique road relationships,** while properties were assigned based on the original **121,961 records.**

2.3. Verify CLEAN Road Count (with no duplicates for the reversed edges)



2.4. Find Shortest Path

Q2. Find the shortest path between two intersections (using selected ID interval 1-1000)

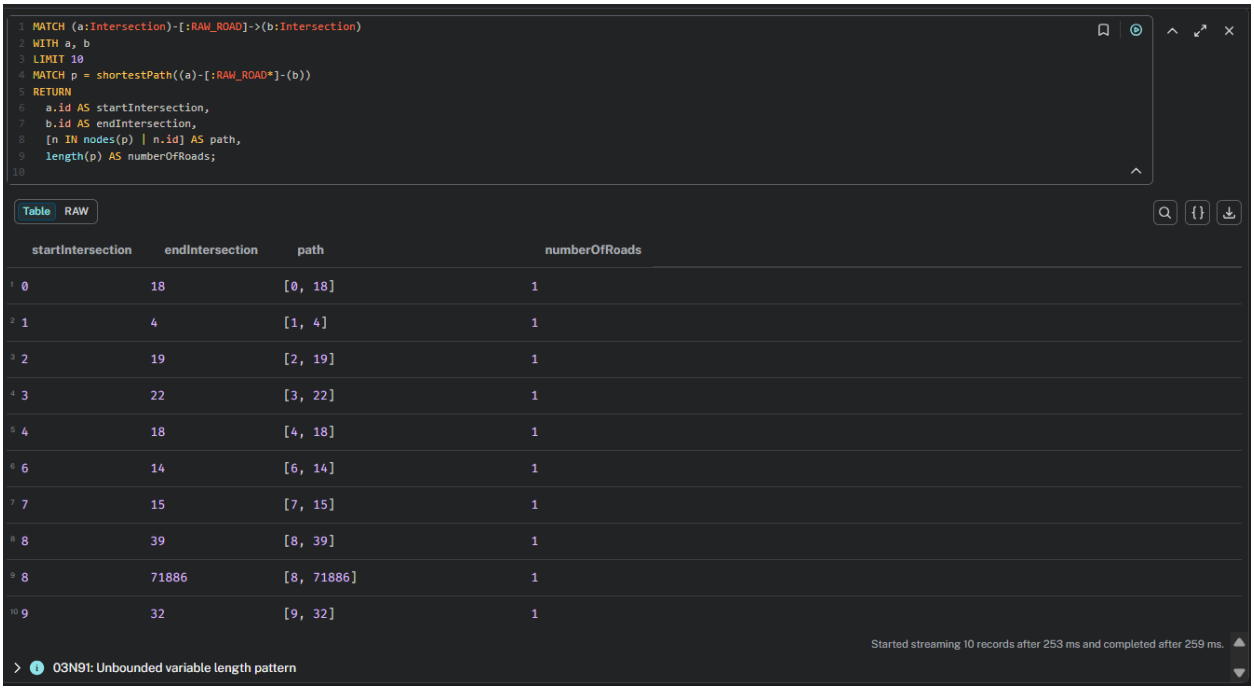


The shortest path was computed using **Neo4j's** built-in **shortestPath()** function on the available road relationships. Since the Graph Data Science plugin was not enabled, the path minimizes the number of road segments rather than distance. We used subset of intersections (IDs 1–1000) as example

2. Shortest Path Between Two Intersections (ID Dynamically selected)

In the below picture, we used Neo4j's **shortestPath()** function on the **RAW_ROAD** relationships to dynamically find the shortest path between connected intersections. Instead of hard-coding IDs, the query automatically selects connected pairs, ensuring a valid path exists.

The results show paths of length **1**, meaning each pair of intersections is directly connected by a single road. This confirms that the road network is correctly loaded and that intersection connectivity is properly represented in the graph.



```
MATCH (a:Intersection)-[:RAW_ROAD]->(b:Intersection)
WITH a, b
LIMIT 10
MATCH p = shortestPath((a)-[:RAW_ROAD*]-(b))
RETURN
  a.id AS startIntersection,
  b.id AS endIntersection,
  [n IN nodes(p) | n.id] AS path,
  length(p) AS numberOfRoads;
```

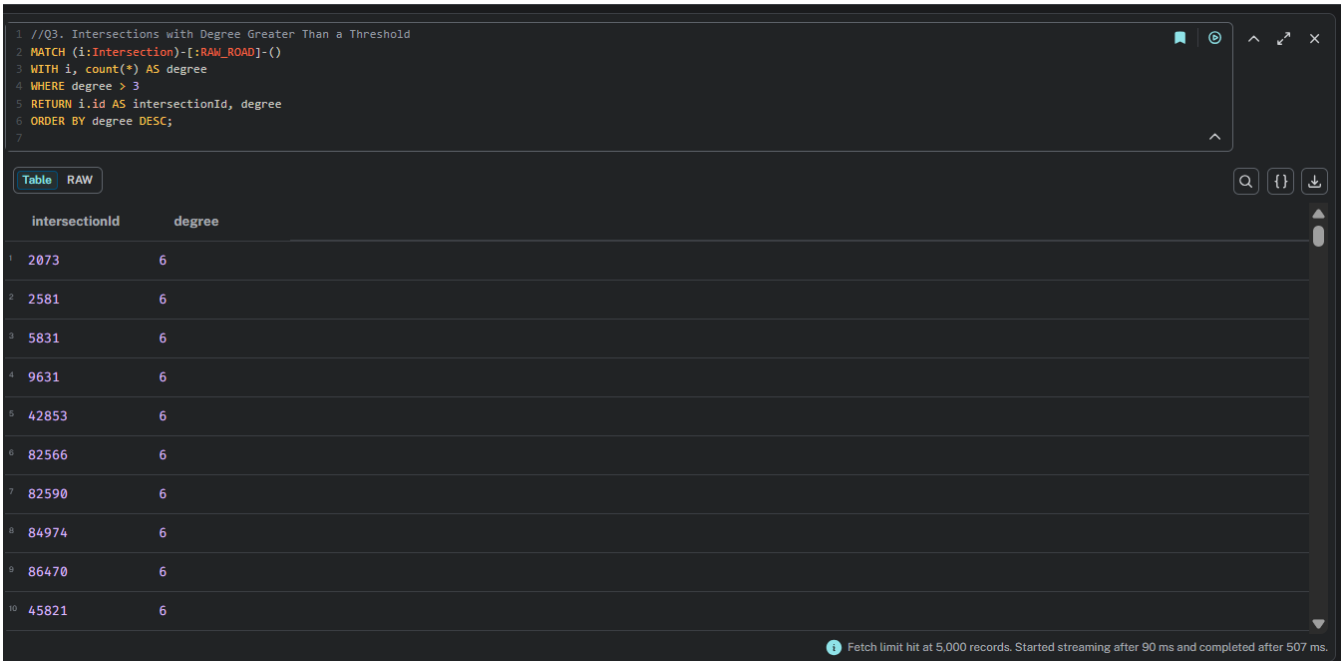
Table RAW

	startIntersection	endIntersection	path	numberOfRoads
1	0	18	[0, 18]	1
2	1	4	[1, 4]	1
3	2	19	[2, 19]	1
4	3	22	[3, 22]	1
5	4	18	[4, 18]	1
6	6	14	[6, 14]	1
7	7	15	[7, 15]	1
8	8	39	[8, 39]	1
9	8	71886	[8, 71886]	1
10	9	32	[9, 32]	1

Started streaming 10 records after 253 ms and completed after 259 ms.

03N91: Unbounded variable length pattern

Q3. Intersections with Degree Greater Than a Threshold



```
//Q3. Intersections with Degree Greater Than a Threshold
MATCH (i:Intersection)-[:RAW_ROAD]-()
WITH i, count(*) AS degree
WHERE degree > 3
RETURN i.id AS intersectionId, degree
ORDER BY degree DESC;
```

Table RAW

	intersectionId	degree
1	2073	6
2	2581	6
3	5831	6
4	9631	6
5	42853	6
6	82566	6
7	82590	6
8	84974	6
9	86470	6
10	45821	6

Fetch limit hit at 5,000 records. Started streaming after 90 ms and completed after 507 ms.

The results show intersections with a degree greater than 3, meaning they are connected to many roads. All listed intersections have a **degree of 6**, identifying them as **highly connected hubs** in the road network. These intersections are important for connectivity and likely represent major junctions where multiple roads meet.

Q4. Analyze the centrality of intersections using the Betweenness Centrality algorithm

Q5. Create a simple dashboard using Plotly (or other options) to display key metrics

To create a dashboard, let first check total number of Roads and Interconnections

Q5.1 Total Roads

```
neo4j$ //Q5.1. Total Roads MATCH ()-[r:ROAD]-() RETURN count(r) AS totalRoads;
```

Table RAW

totalRoads
242982

Started streaming 1 record after 154 ms and completed after 406 ms.

The value **242,982** appears because each physical road is stored as two directional relationships, resulting in double the count of unique roads.

Q5.2. Total Intersections

```
MATCH (i:Intersection)
RETURN count(i) AS totalIntersections;
```

Table RAW

totalIntersections
87575

Started streaming 1 record after 22 ms and completed after 23 ms.

The result shows that the graph contains **87,575 intersection nodes**, confirming that all intersections loaded and stored uniquely with no **duplicate (:Intersection)** nodes present.

Q6. Degree Distribution

Roads connect to each intersection

```
// Q6: Degree Distribution
MATCH (i:Intersection)
WITH COUNT { (i)-[:ROAD]-() } AS degree
RETURN degree, count(*) AS frequency
ORDER BY degree;
```

Table RAW

degree	frequency
1	2234
2	35486
3	29941
4	19652
5	227
6	35

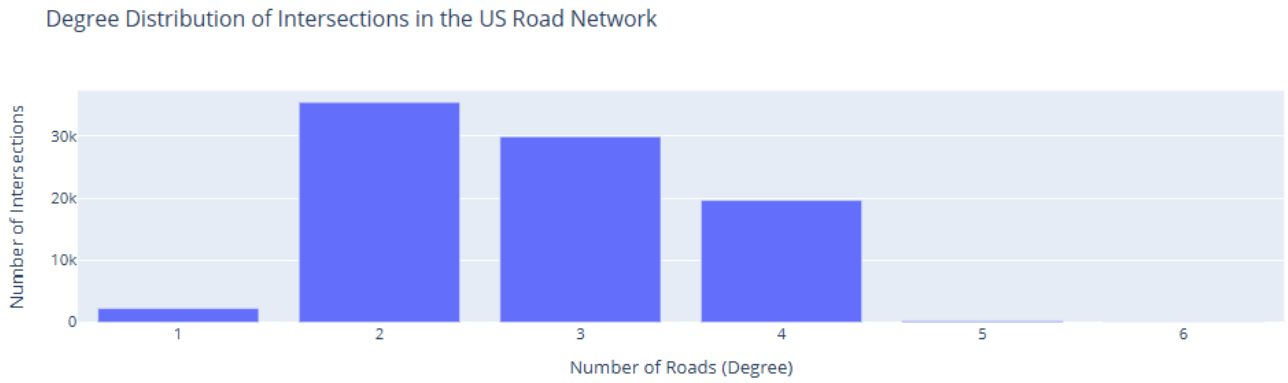
Started streaming 6 records after 141 ms and completed after 299 ms.

The degree distribution shows that the majority of intersections have **2 or 3 connecting roads**, which is typical for road networks. Intersections with **4 connections** are less common, and those with **5 or 6 connections** are very rare, indicating that only a small number of highly connected hubs exist. Overall, this confirms that the network is **sparse**, with most intersections having low connectivity and only a few acting as major junctions.

Degree Distribution Bar Chart

Degree Distribution Bar Chart

[13]:
```
# Cypher query (Degree Distribution)
query = """
MATCH (i:Intersection)
WITH COUNT { (i)-[:ROAD]-() } AS degree
RETURN degree, count(*) AS frequency
ORDER BY degree
"""

# ---------------------------------
# Run query using graph (no session)
# ---------------------------------
df = graph.run(query).to_data_frame()

# ---------------------------------
# Plotly bar chart
# ---------------------------------
fig = px.bar(
    df,
    x="degree",
    y="frequency",
    labels={
        "degree": "Number of Roads (Degree)",
        "frequency": "Number of Intersections"
    },
    title="Degree Distribution of Intersections in the US Road Network"
)

fig.update_layout(
    xaxis=dict(tickmode="linear"),
    bargap=0.2
)

fig.show()
```

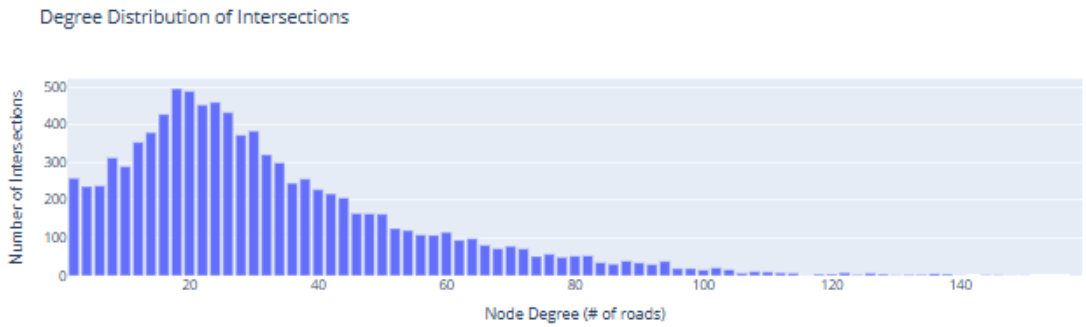


The bar chart shows that most intersections in the network have **2 or 3 connecting roads**, indicating typical road junctions. Intersections with **4 connections** are less common, and those with **5 or 6 connections** are very rare. Overall, this confirms that the road network is **sparse**, with only a small number of highly connected intersections acting as hubs.

Distribution of Intersections

```
[38]: # Query degree distribution
with driver.session(database="road-network-db") as session:
    results = session.run("""
        MATCH (i:Intersection)-[r]-()
        WITH i, count(r) AS degree
        RETURN degree, count(*) AS frequency
        ORDER BY degree
    """)
    df = pd.DataFrame([dict(record) for record in results])

print(df)  # Shows degree vs frequency table

# Plot degree distribution
fig = px.bar(df, x="degree", y="frequency",
             labels={"degree": "Node Degree (# of roads)", "frequency": "Number of Intersections"},
             title="Degree Distribution of Intersections")
fig.show()
```

```
    degree  frequency
0        2        258
1        4        236
2        6        238
3        8        312
4       10        289
..     ...        ...
70     144          2
71     146          2
72     148          1
73     150          1
74     158          1

[75 rows x 2 columns]
```



Q7. Top 10 Most Connected Intersections

This analysis identifies the **top 10 most connected intersections** in the road network based on their degree, which represents the number of roads connected to each intersection. All listed intersections have a **degree of 6**, indicating they are the most highly connected nodes in the network. These intersections function as **local hubs**, and their high connectivity suggests they play an important role in maintaining efficient traffic flow and overall network connectivity.

```
1 //Q7. Top 10 Intersections by Degree
2 MATCH (i:Intersection)-[:ROAD]-()
3 WITH i, count(*) AS degree
4 RETURN i.id AS intersectionId, degree
5 ORDER BY degree DESC
6 LIMIT 10;
7
```

Table RAW

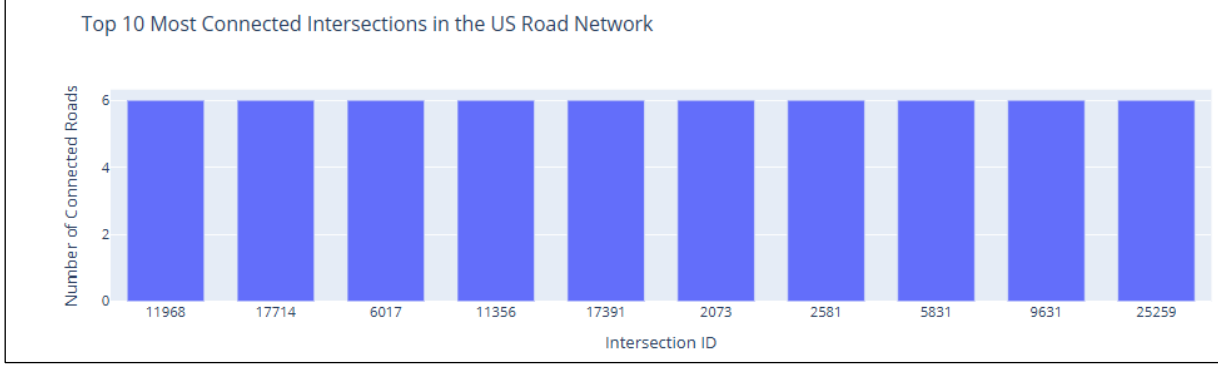
	intersectionId	degree
1	11968	6
2	17714	6
3	6017	6
4	11356	6
5	17391	6
6	2073	6
7	2581	6
8	5831	6
9	9631	6
10	25259	6

The Top 10 Most Connected Intersections visualization

7. Top 10 Most Connected Intersections:

[15]:

```
# Run Cypher query dynamically
# ---------------------------------
query = """
MATCH (i:Intersection)-[:ROAD]-()
WITH i, count(*) AS degree
RETURN i.id AS intersectionId, degree
ORDER BY degree DESC
LIMIT 10
"""

df = graph.run(query).to_data_frame()

# ---------------------------------
# Plotly bar chart
# ---------------------------------
fig = px.bar(
    df,
    x="intersectionId",
    y="degree",
    labels={
        "intersectionId": "Intersection ID",
        "degree": "Number of Connected Roads"
    },
    title="Top 10 Most Connected Intersections in the US Road Network"
)

fig.update_layout(
    xaxis=dict(type="category"),
    bargap=0.3
)

fig.show()
```



The bar chart shows the **top 10 most connected intersections** in the road network, and all of them have a **degree of 6**, meaning each is connected to six roads. This indicates that these intersections act as **local hubs**, providing multiple routing options and playing an important role in maintaining efficient traffic flow and overall network connectivity.

Q8. Intersection Categories by Degree

In this section, intersections are categorized based on their degree, which represents the number of roads connected to each intersection. By grouping intersections into low, medium, and high connectivity categories, this analysis provides a clear overview of how connectivity is distributed across the road network and highlights the presence of simple junctions versus more critical hub intersections

```
// Q8: Intersection Categories by Degree
MATCH (i:Intersection)
WITH COUNT { (i)-[:ROAD]-() } AS degree
WITH
  CASE
    WHEN degree <= 2 THEN 'Low Connectivity'
    WHEN degree <= 4 THEN 'Medium Connectivity'
    ELSE 'High Connectivity'
  END AS category
RETURN category, count(*) AS numberOfIntersections
ORDER BY numberOfIntersections DESC;
```

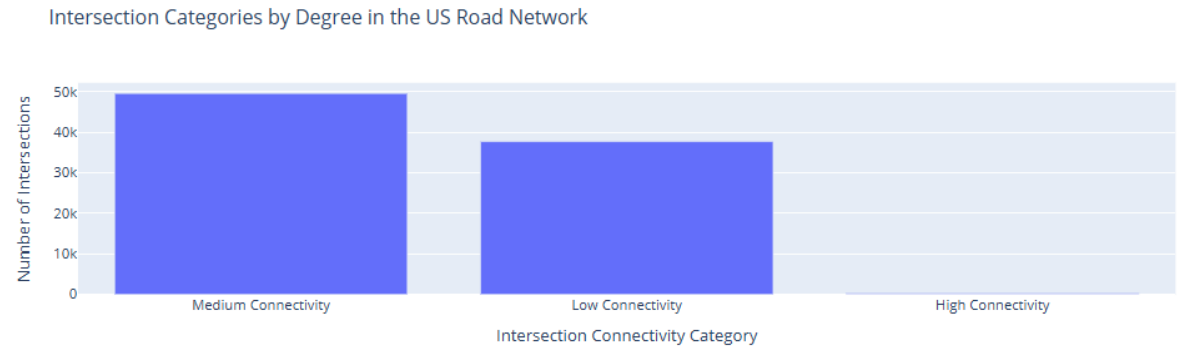
Table RAW

	category	numberOfIntersec
1	"Medium Connectivity"	49593
2	"Low Connectivity"	37720
3	"High Connectivity"	262

Started streaming 3 records after 16,393 ms and completed after 17,961 ms.

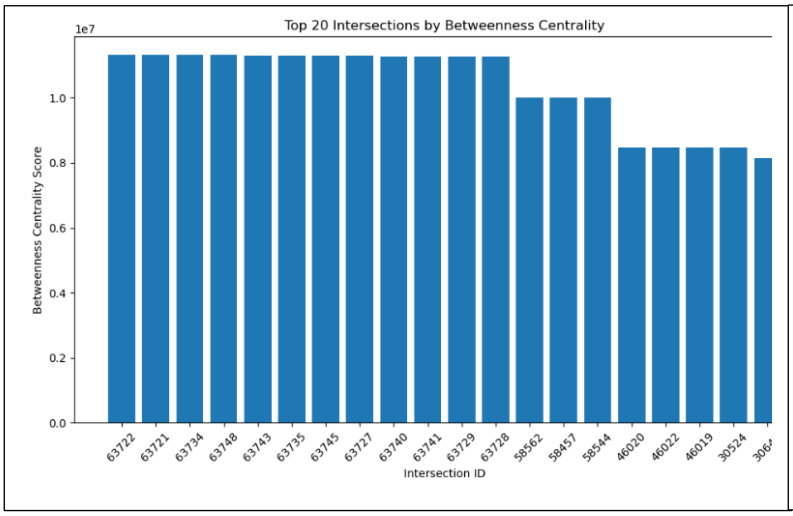
8. Intersection Categories by Degree

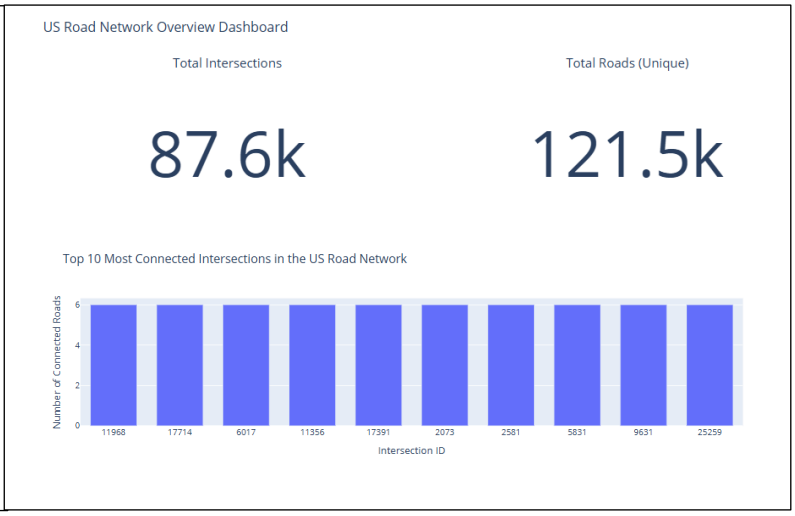
```
# Run query dynamically from db
query = """
MATCH (i:Intersection)
WITH COUNT { (i)-[:ROAD]-() } AS degree
WITH
  CASE
    WHEN degree <= 2 THEN 'Low Connectivity'
    WHEN degree <= 4 THEN 'Medium Connectivity'
    ELSE 'High Connectivity'
  END AS category
RETURN category, count(*) AS numberOfIntersections
ORDER BY numberOfIntersections DESC
"""

df = graph.run(query).to_data_frame()

# Bar chart
fig = px.bar(
    df,
    x="category",
    y="numberOfIntersections",
    labels={
        "category": "Intersection Connectivity Category",
        "numberOfIntersections": "Number of Intersections"
    },
    title="Intersection Categories by Degree in the US Road Network"
)

fig.show()
```

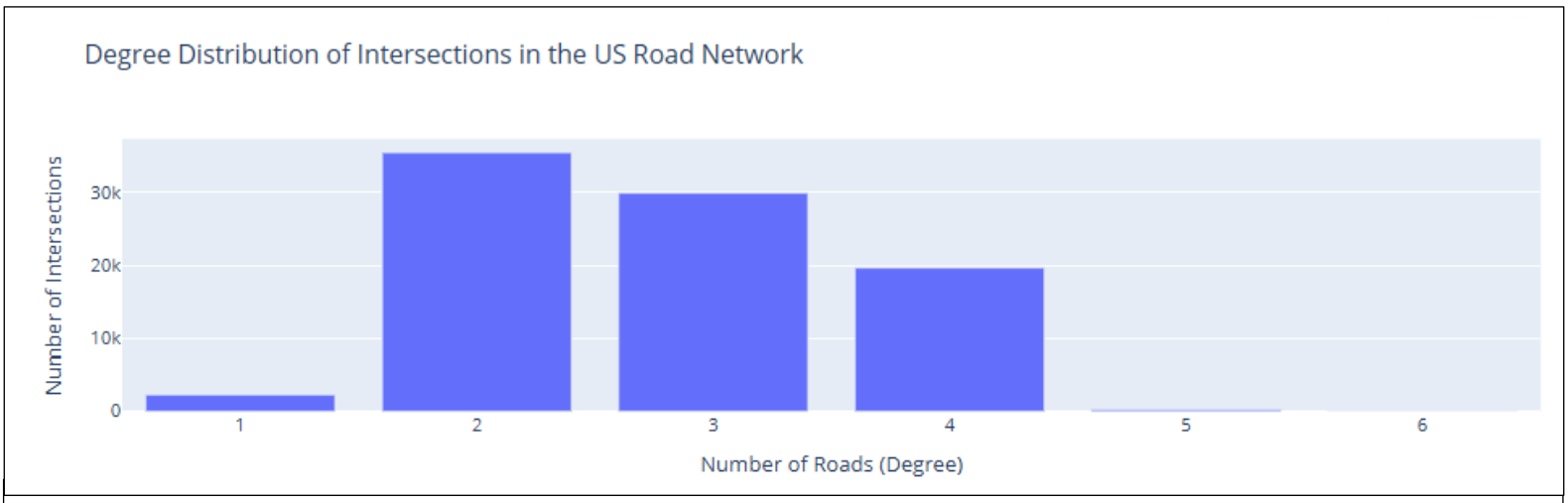


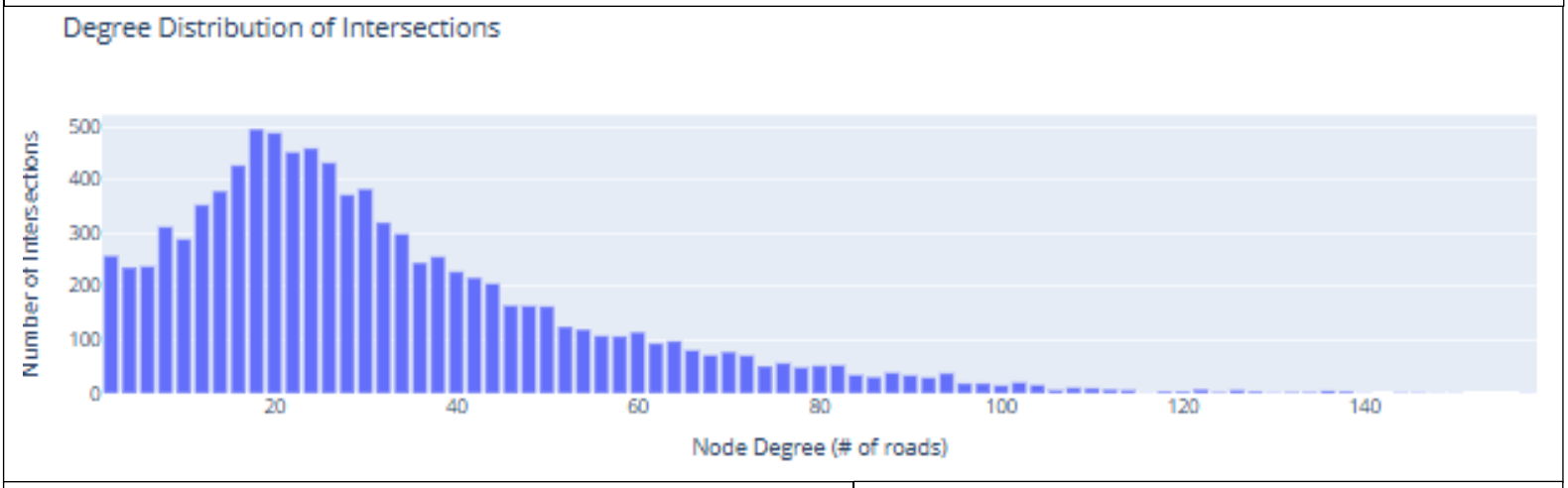
The results show that most intersections fall into the **low** and **medium connectivity** categories, while only a small number have **high connectivity**. This indicates that the road network is largely composed of simple intersections, with few highly connected hubs that play a more critical role in connectivity.

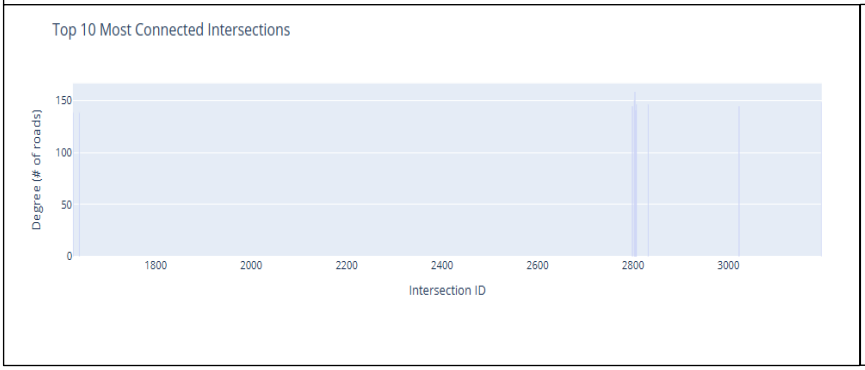
3. Dashboard Summarizing findings

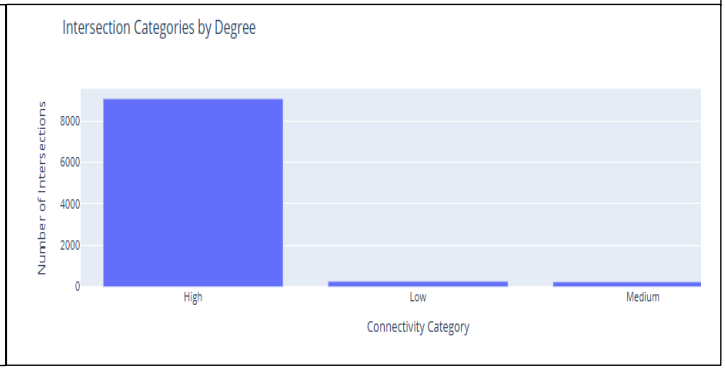












4. Results Interpretation and Insights

The visualizations provide a comprehensive overview of the structure and behavior of the U.S. road network. The overview dashboard confirms that the network contains **87,575 intersections** and **121,961 unique roads,** confirming that the road network is sparse, with an average node degree of approximately **2.8.**

The **degree distribution charts** show that most intersections are connected to **two or three roads**, with very few intersections having a high number of connections. This indicates that the network is dominated by simple junctions, while only a small fraction of intersections function as highly connected hubs. The long-tailed distribution observed in the detailed degree histogram further confirms this pattern, which is typical of real-world transportation networks.

The **Top 10 intersections by degree** analysis reveals that the most connected intersections have a degree of six, highlighting their role as local connectivity hubs. Although these intersections are few, they are important for maintaining efficient local traffic flow.

The **betweenness centrality visualization** identifies intersections that lie on many shortest paths between other intersections. These nodes act as critical bridges within the network. Even if such intersections are not the most connected by degree, their position makes them strategically important, as disruptions at these points could significantly affect overall connectivity and traffic movement.

Finally, the **intersection categories by degree** chart shows that the majority of intersections fall into the **low connectivity** category, with relatively few in the medium and high connectivity groups. This further confirms that the network is sparse and relies on a limited number of critical nodes to support broader connectivity.

5 . Lessons Learned

Through this analysis, I learned that graph-based modeling is highly effective for analyzing large transportation networks. Degree alone does not fully capture the importance of an intersection; centrality measures such as betweenness are essential for identifying true bottlenecks and critical routes. I also learned the importance of sampling techniques when applying computationally intensive algorithms to large graphs, as they enable meaningful insights without overwhelming system resources. Additionally, visual dashboards proved to be a powerful tool for summarizing complex graph metrics in an interpretable and decision-friendly way.

6. Recommendations

Based on the findings, intersections with **high betweenness centrality** should be prioritized for monitoring, maintenance, and traffic management, as failures at these points would have a disproportionate impact on network performance. Traffic planning efforts should also consider strengthening alternative routes around these critical intersections to improve resilience. For future work, incorporating real-time traffic data, road capacity, or temporal patterns would further enhance the analysis and provide deeper insights into congestion and network reliability.

Annex

- ✓ Neo4j database
- ✓ Jupyter Notebook (HTML format)
- ✓ Project report

End