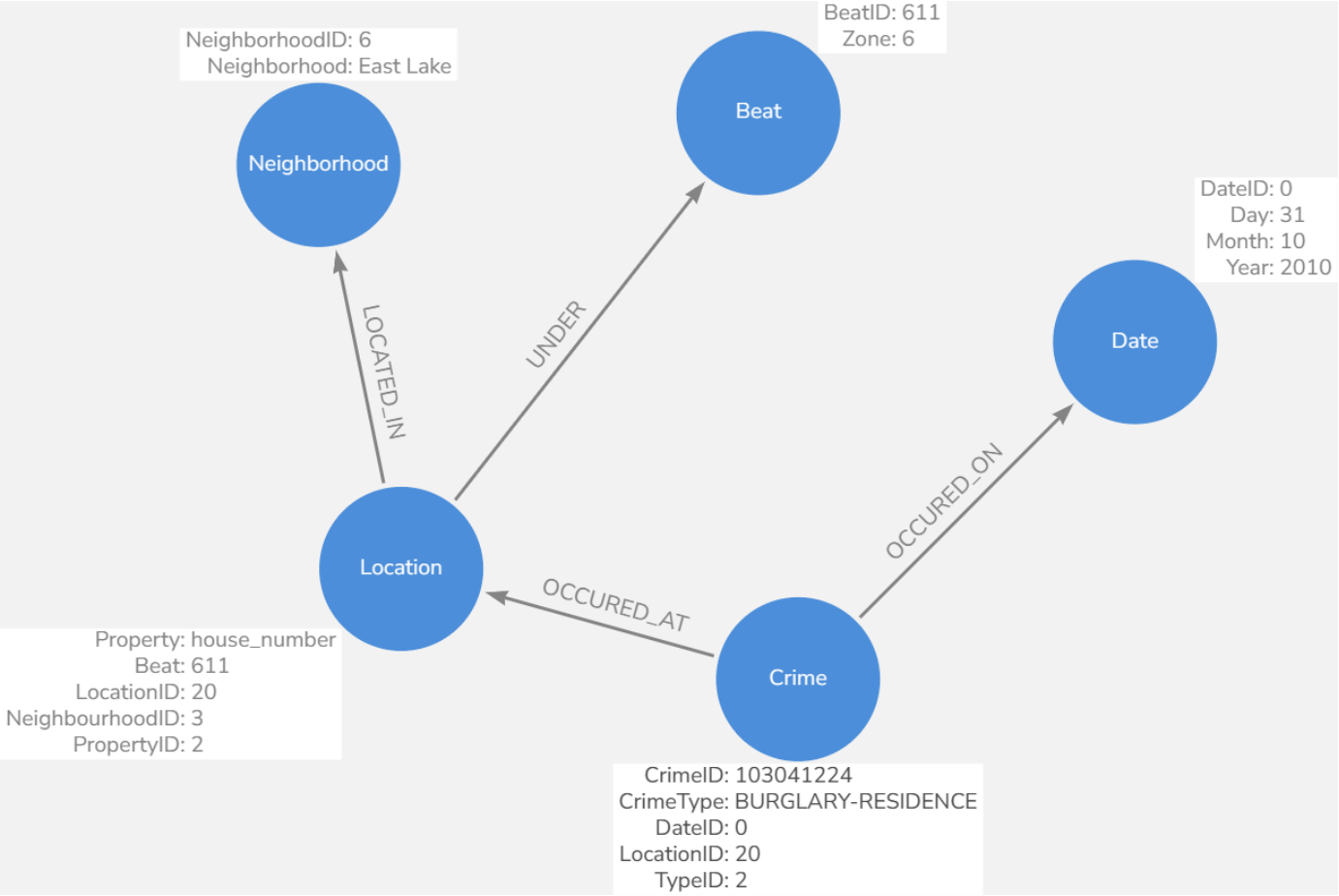


Figure 1: Graph from Arrow tool

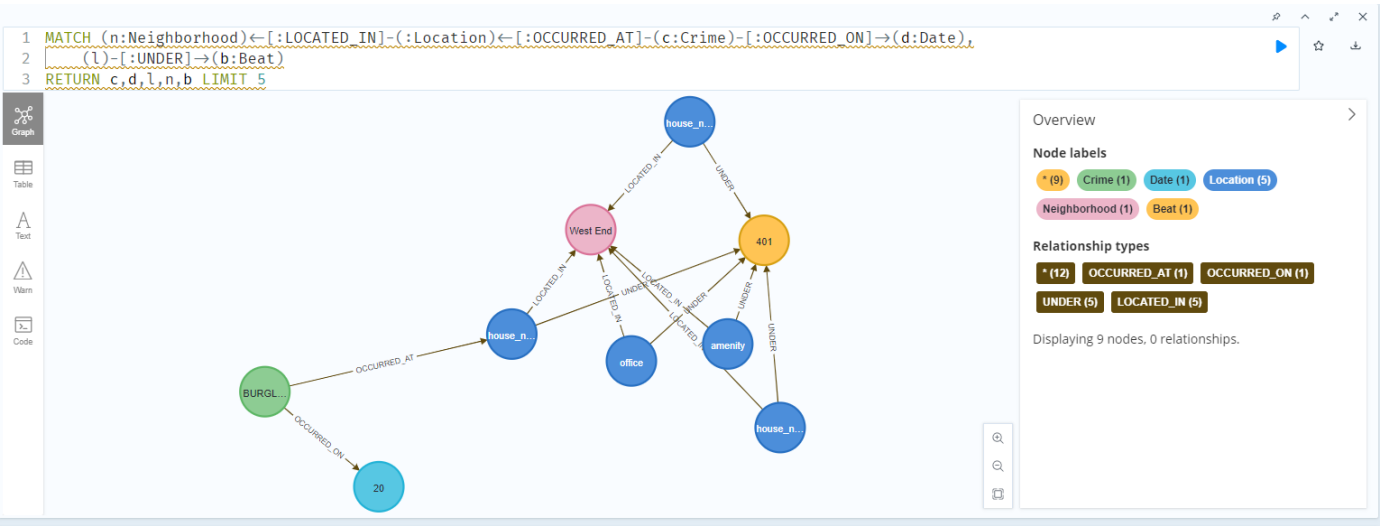
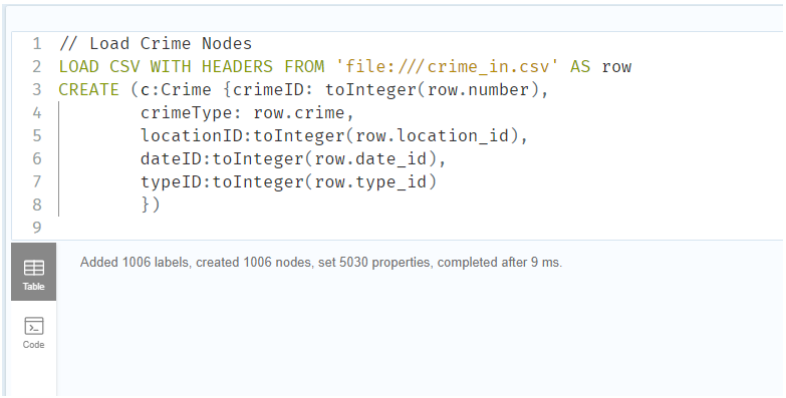


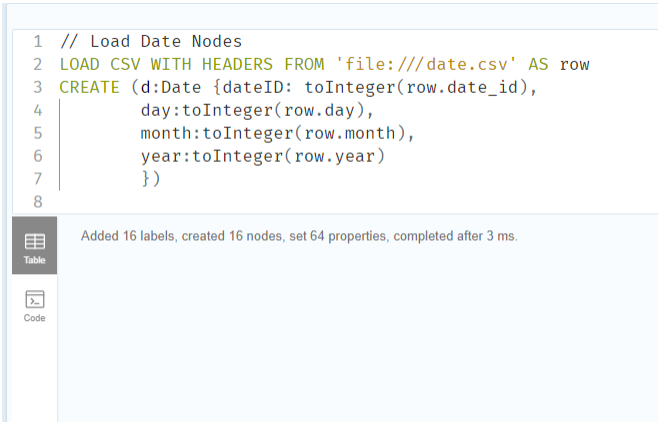
Figure 2: Graph from Neo4j

Graph database creation:

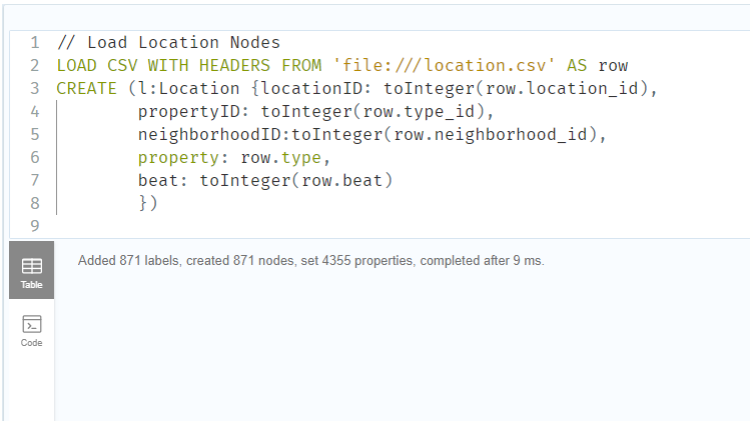
```
// Load Crime Nodes
LOAD CSV WITH HEADERS FROM 'file:///crime_in.csv' AS row
CREATE (c:Crime {crimeID: toInteger(row.number),
        crimeType: row.crime,
        locationID:toInteger(row.location_id),
        dateID:toInteger(row.date_id),
        typeID:toInteger(row.type_id)
        })
```



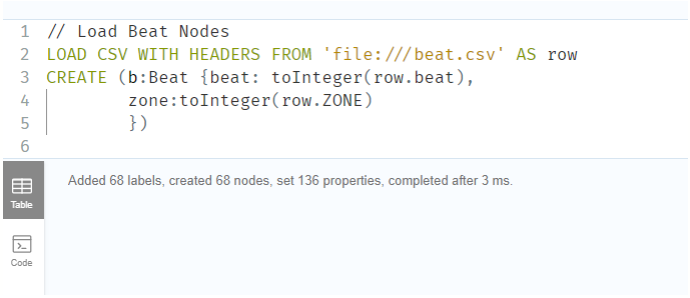

```
// Load Date Nodes
LOAD CSV WITH HEADERS FROM 'file:///date.csv' AS row
CREATE (d:Date {dateID: toInteger(row.date_id),
        day:toInteger(row.day),
        month:toInteger(row.month),
        year:toInteger(row.year)
        })
```



```
// Load Location Nodes
LOAD CSV WITH HEADERS FROM 'file:///location.csv' AS row
CREATE (l:Location {locationID: toInteger(row.location_id),
        propertyID: toInteger(row.type_id),
        neighborhoodID:toInteger(row.neighborhood_id),
        property: row.type,
        beat: toInteger(row.beat)
        })
```

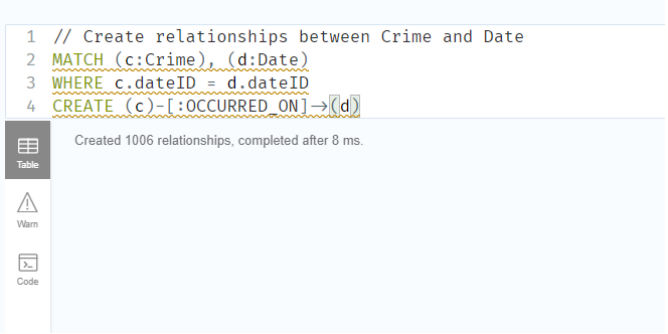



```
// Load Beat Nodes
LOAD CSV WITH HEADERS FROM 'file:///beat.csv' AS row
CREATE (b:Beat {beat: toInteger(row.beat),
        zone:toInteger(row.ZONE)
        })
```

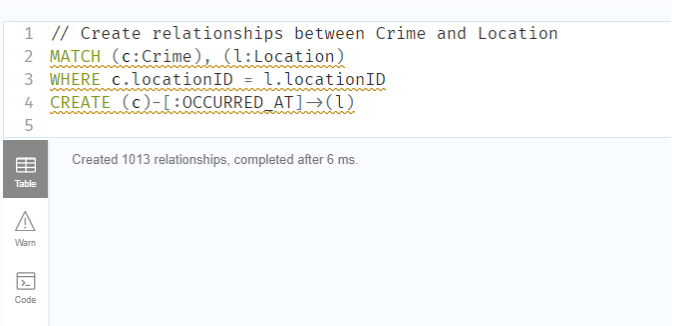


```
// Load Neighborhood nodes
LOAD CSV WITH HEADERS FROM 'file:///neighborhood.csv' AS row
CREATE (n:Neighborhood {neighborhoodID: toInteger(row.neighborhood_id),
      neighborhood:row.neighborhood
      })
```

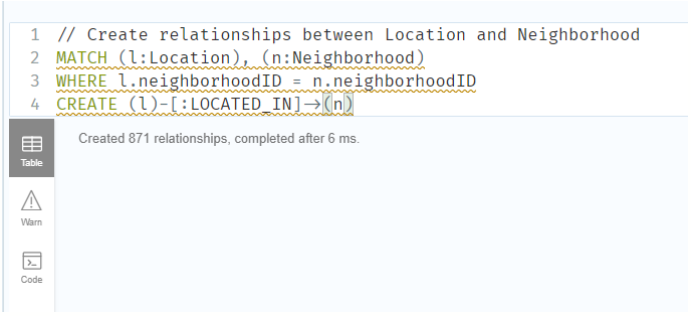

```
// Create relationships between Crime and Date
MATCH (c:Crime), (d:Date)
WHERE c.dateID = d.dateID
CREATE (c)-[:OCCURRED_ON]->(d)
```



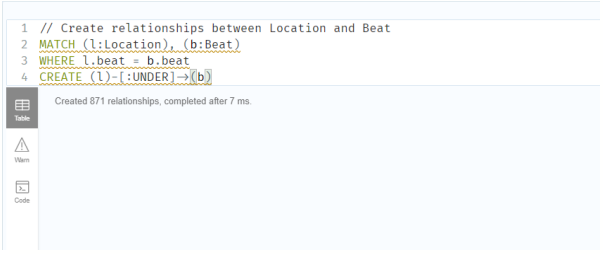

```
// Create relationships between Crime and Location
MATCH (c:Crime), (l:Location)
WHERE c.locationID = l.locationID
CREATE (c)-[:OCCURRED_AT]->(l)
```



```
// Create relationships between Location and Neighborhood
MATCH (l:Location), (n:Neighborhood)
WHERE l.neighborhoodID = n.neighborhoodID
CREATE (l)-[:LOCATED_IN]->(n)
```




```
// Create relationships between Location and Beat
MATCH (l:Location), (b:Beat)
WHERE l.beat = b.beat
CREATE (l)-[:UNDER]->(b)
```



Querries:

// How many crimes are recorded for a given crime type in a specified neighbourhood for a particular perio
d?

```
MATCH (c:Crime)-[:OCCURRED_AT]->(l:Location)-[:LOCATED_IN]->(n:Neighborhood),
      (c)-[:OCCURRED_ON]->(d:Date)
WHERE c.crimeType = 'LARCENY-NON VEHICLE'
  AND n.neighborhood='Downtown'
  AND d.day = 31
  AND d.month = 10
RETURN count(c) AS crimeCount
```

//Find the neighbourhoods that share the same crime types, organise in decending order of the number of c ommon crime types.

```
MATCH (n1:Neighborhood)<-[:LOCATED_IN]-(:Location)<-[:OCCURRED_AT]-(c1:Crime)
WITH n1, COLLECT(DISTINCT c1.crimeType) AS crimeTypes
MATCH (n2:Neighborhood)<-[:LOCATED_IN]-(:Location)<-[:OCCURRED_AT]-(c2:Crime)
WHERE n2 <> n1
WITH n1, n2, COLLECT(DISTINCT c2.crimeType) AS commonCrimeTypes, crimeTypes
RETURN n1.neighborhood AS neighborhood1, n2.neighborhood AS neighborhood2, SIZE([x IN common
CrimeTypes WHERE x IN crimeTypes]) AS numCommonCrimeTypes
ORDER BY numCommonCrimeTypes DESC
```

```
1  //Find the neighbourhoods that share the same crime types, organise in decending order of the number of common crime types.
2
3  MATCH (n1:Neighborhood)<-[:LOCATED_IN]-(:Location)<-[:OCCURRED_AT]-(c1:Crime)
4  WITH n1, COLLECT(DISTINCT c1.crimeType) AS crimeTypes
5  MATCH (n2:Neighborhood)<-[:LOCATED_IN]-(:Location)<-[:OCCURRED_AT]-(c2:Crime)
6  WHERE n2 <> n1
7  WITH n1, n2, COLLECT(DISTINCT c2.crimeType) AS commonCrimeTypes, crimeTypes
8  RETURN n1.neighborhood AS neighborhood1, n2.neighborhood AS neighborhood2, SIZE([x IN commonCrimeTypes WHERE x IN crimeTypes]) AS
   numCommonCrimeTypes
9  ORDER BY numCommonCrimeTypes DESC
```

Table | Text | Code

	neighborhood1	neighborhood2	numCommonCrimeTypes
1	"West End"	"Old Fourth Ward"	7
2	"West End"	"Inman Park"	7
3	"West End"	"Center Hill"	7
4	"West End"	"Edgewood"	7
5	"West End"	"Grove Park"	7
6	"West End"	"Pittsburgh"	7
7			

Started streaming 10712 records in less than 1 ms and completed in less than 1 ms, displaying first 1000 rows.

// Return the top 5 neighbourhoods for a specified crime for a specified duration.

```
MATCH (n:Neighborhood)<-[:LOCATED_IN]-(:Location)<-[:OCCURRED_AT]-(c:Crime)-
[:OCCURRED_ON]->(d:Date)
WHERE c.crimeType = 'AGG ASSAULT'
  AND d.day >=10
  AND d.day <=28
RETURN n.neighborhood AS neighborhood, COUNT(c) AS numCrimes
ORDER BY numCrimes DESC
LIMIT 5
```

```
1  // Return the top 5 neighbourhoods for a specified crime for a specified duration.
2
3  MATCH (n:Neighborhood)←[:LOCATED_IN]-(:Location)←[:OCCURRED_AT]-(c:Crime)-[:OCCURRED_ON]→(d:Date)
4  WHERE c.crimeType = 'AGG ASSAULT'
5    AND d.day ⩾10
6    AND d.day ⩽28
7  RETURN n.neighborhood AS neighborhood, COUNT(c) AS numCrimes
8  ORDER BY numCrimes DESC
9  LIMIT 5
10
```

Table | Text | Code

	neighborhood	numCrimes
1	"Downtown"	5
2	"Pittsburgh"	5
3	"English Avenue"	3
4	"Midtown"	3
5	"West End"	3

Started streaming 5 records after 1 ms and completed after 4 ms.

// Find the types of crimes for each property type.

```
MATCH (l:Location)<-[:OCCURRED_AT]-(c:Crime)
RETURN l.property AS propertyType, COLLECT(DISTINCT c.crimeType) AS crimeTypes
```

```
// Find the types of crimes for each property type.

MATCH (l:Location)<-[:OCCURRED_AT]-(c:Crime)
RETURN l.property AS propertyType, COLLECT(DISTINCT c.crimeType) AS crimeTypes
```

Table | Text | Code

	propertyType	crimeTypes
1	"office"	["AUTO THEFT", "LARCENY-FROM VEHICLE", "LARCENY-NON VEHICLE", "BURGLARY-NONRES"]
2	"shop"	["LARCENY-FROM VEHICLE", "LARCENY-NON VEHICLE", "AUTO THEFT", "ROBBERY-COMMERCIAL", "BURGLARY-RESIDENCE", "AGG ASSAULT"]
3	"house_number"	["LARCENY-NON VEHICLE", "ROBBERY-PEDESTRIAN", "AGG ASSAULT", "LARCENY-FROM VEHICLE", "AUTO THEFT", "BURGLARY-RESIDENCE", "BURGLARY-NONRES", "HOMICIDE", "ROBBERY-
4	"building"	["LARCENY-FROM VEHICLE", "LARCENY-NON VEHICLE", "AUTO THEFT", "ROBBERY-PEDESTRIAN", "BURGLARY-RESIDENCE", "BURGLARY-NONRES"]
5	"amenity"	["AGG ASSAULT", "LARCENY-FROM VEHICLE", "AUTO THEFT", "LARCENY-NON VEHICLE", "BURGLARY-RESIDENCE", "ROBBERY-COMMERCIAL", "ROBBERY-PEDESTRIAN", "BURGLARY-NONRE
6	"tourism"	["RAPE", "LARCENY-FROM VEHICLE", "LARCENY-NON VEHICLE", "AGG ASSAULT", "ROBBERY-PEDESTRIAN"]

// Which month of a specified year has the highest crime rate? Return one record each for each beat. (also n eed to return one record for each zone.)

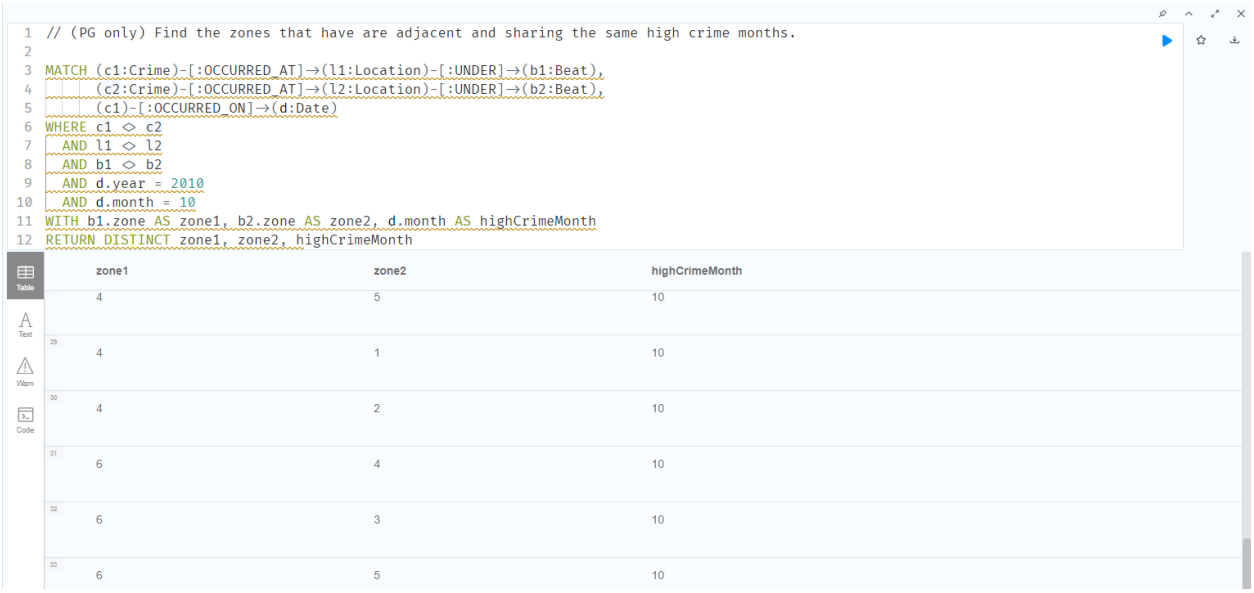
```
MATCH (c:Crime)-[:OCCURRED_ON]->(d:Date), (l:Location)-[:UNDER]->(b:Beat)
WHERE d.year = 2010
WITH d.month AS month, b.beat AS beat, b.zone AS zone, COUNT(c) AS numCrimes
ORDER BY beat, zone, numCrimes DESC
WITH beat, zone, COLLECT(month)[0] AS highestCrimeMonth
RETURN beat, zone, highestCrimeMonth
```

```
1  // Which month of a specified year has the highest crime rate? Return one record each for each beat. (PG students also need to
   return one record for each zone.)
2
3  MATCH (c:Crime)-[:OCCURRED_ON]→(d:Date), (l:Location)-[:UNDER]→(b:Beat)
4  WHERE d.year = 2010
5  WITH d.month AS month, b.beat AS beat, b.zone AS zone, COUNT(c) AS numCrimes
6  ORDER BY beat, zone, numCrimes DESC
7  WITH beat, zone, COLLECT(month)[0] AS highestCrimeMonth
8  RETURN beat, zone, highestCrimeMonth
9
```

Table | Text | Warn | Code

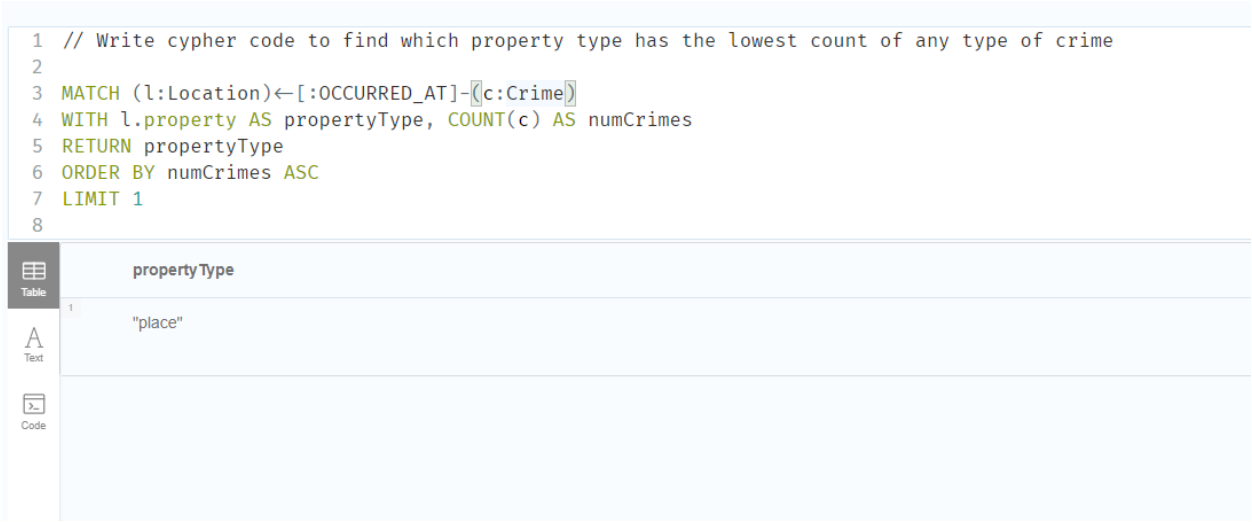
	beat	zone	highestCrimeMonth
1	101	1	10
2	102	1	10
3	103	1	10
4	104	1	10
5	106	1	10

```
//  Find the zones that have are adjacent and sharing the same high crime months.
MATCH (c1:Crime)-[:OCCURRED_AT]->(l1:Location)-[:UNDER]->(b1:Beat),
      (c2:Crime)-[:OCCURRED_AT]->(l2:Location)-[:UNDER]->(b2:Beat),
      (c1)-[:OCCURRED_ON]->(d:Date)
WHERE c1 <> c2
  AND l1 <> l2
  AND b1 <> b2
  AND d.year = 2010
  AND d.month = 10
WITH b1.zone AS zone1, b2.zone AS zone2, d.month AS highCrimeMonth
RETURN DISTINCT zone1, zone2, highCrimeMonth
```



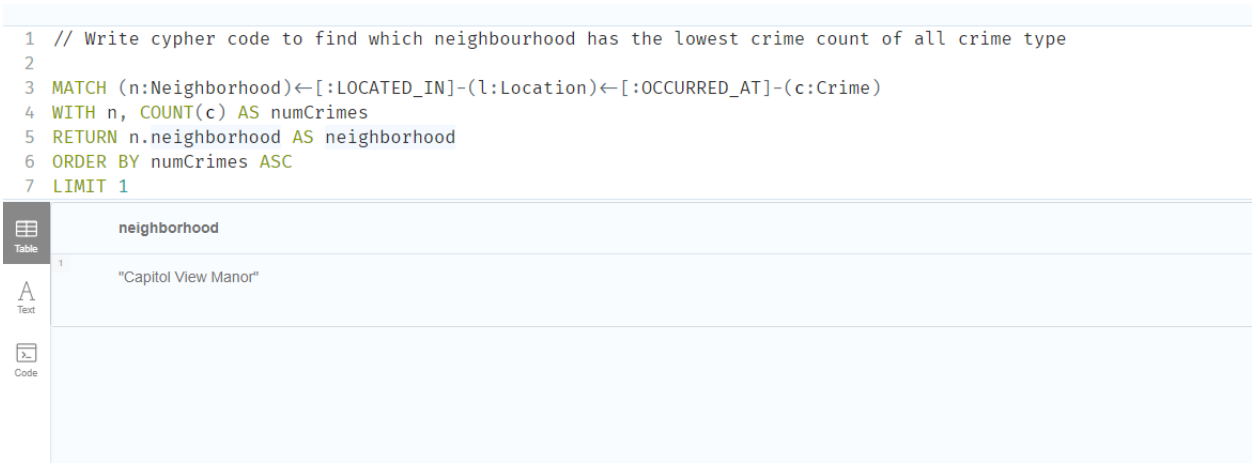
// Write cypher code to find which property type has the lowest count of any type of crime

```
MATCH (l:Location)<-[:OCCURRED_AT]-(c:Crime)
WITH l.property AS propertyType, COUNT(c) AS numCrimes
RETURN propertyType
ORDER BY numCrimes ASC
LIMIT 1
```



// Write cypher code to find which neighbourhood has the lowest crime count of all crime type

```
MATCH (n:Neighborhood)<-[:LOCATED_IN]-(l:Location)<-[:OCCURRED_AT]-(c:Crime)
WITH n, COUNT(c) AS numCrimes
RETURN n.neighborhood AS neighborhood
ORDER BY numCrimes ASC
LIMIT 1
```



The Atlanta dataset's crime data lends itself well to analysis using the graphic database.

1. Complex linkages between crimes, suspects, victims, and locations can be handled effectively using graph databases.

2. Investigating relationships and locating crime trends, hotspots, or repeated behaviors are made easier with the help of graph database traversal and pattern matching. 3. Scalability and good performance are features of graph databases. This is crucial for handling big quantities of criminal data and running intricate searches across various related entities.

4. Because graph databases are schema less, it is simple to adapt them to new attributes and growing criminal data models.

5. Context-rich investigations in graph databases offer more information on gang affiliations, social ties, and networks between suspects and victims.

6. Logical reasoning and semantic reasoning assist reveal hidden relationships and foresee probable links to criminal activity.

7. A chart database combines information from various sources and offers a consolidated view of crime data from various domains, including case reports, arrests, and court records. 8. Multiple agencies or departments can provide and analyze criminal data together thanks to the chart database's collaboration and data sharing features.

9. Chart databases can be used to determine resource allocations, criminal tendencies, and preemptive law enforcement tactics.

10. The chart database makes use of the chart structure to visualize crime networks and enable geographical analysis to comprehend crime trends in Atlanta.

YouTube Video link:

https://youtu.be/peEenVdcj7g