DALHOUSIE UNIVERSITY

CSCI 5408 – Data Management, Warehousing, Analytics

Assignment 4

Work done by,

Name: Guturu Rama Mohan Vishnu

Banner ID: B00871849

Email: rm286720@dal.ca

DECLARATION

I, Guturu Rama Mohan Vishnu, declare that in assignment 3 of CSCI 5408 course, I declare that all the work done was done by myself and I have not collaborated with anyone for the assignment.

Problem #2

Task #1: Data Processing using Spark – MapReduce to perform count

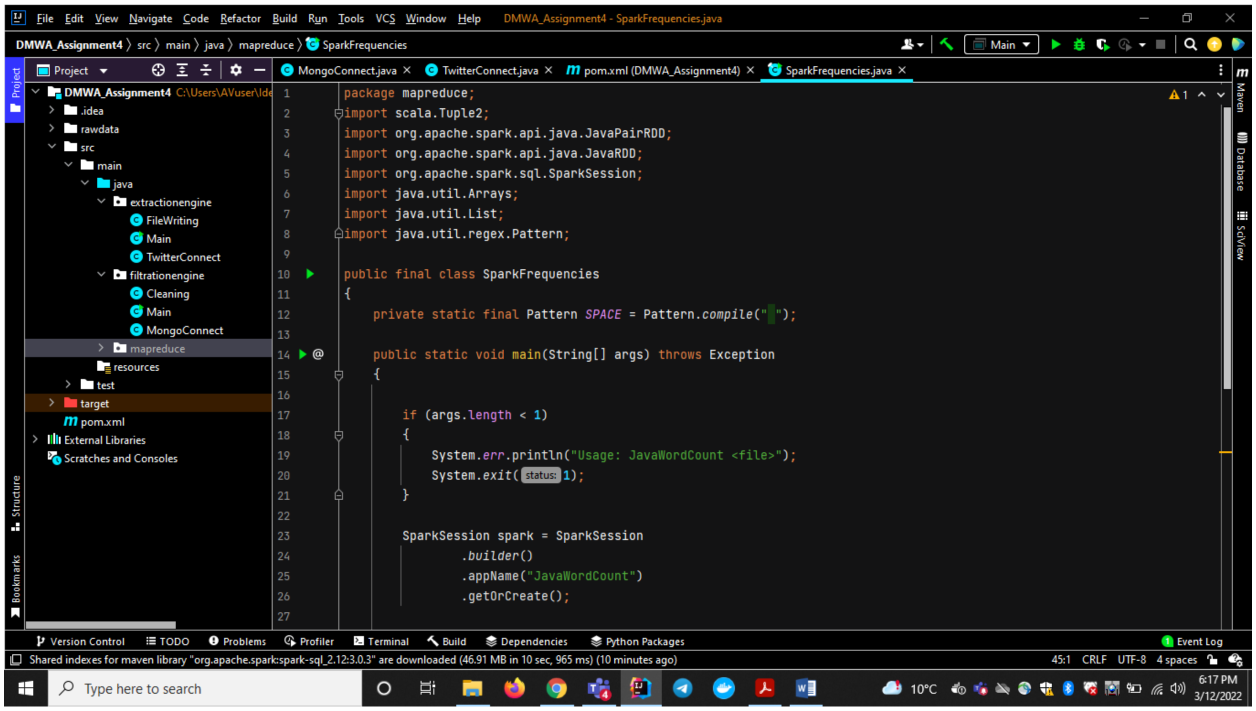
Step 1:

In order to explain how I completed this task, I am creating a flowchart and adding it to a folder in my drive. I am sharing the link of that drive in this pdf and the reason I couldn't add the flowchart here is that the image is long and it won't fit in a single page.

Drive link for flowchart: https://drive.google.com/drive/folders/1RnxczW2VeqWuyzAlfIV3kX0UX3o-mGIN?usp=sharing

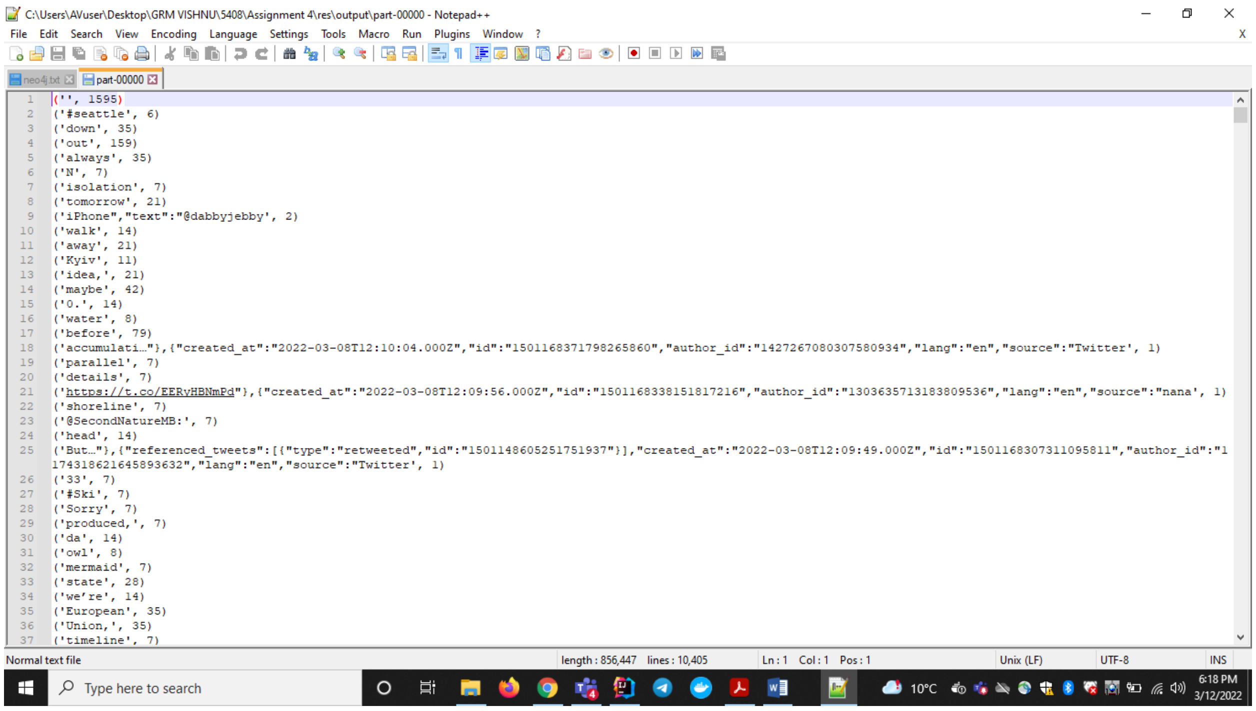
Git URL for java code: https://git.cs.dal.ca/rguturu/csci-5408-w2022-b00871849-gutururamamohanvishnu/-/tree/main/assignment-4

Also, I am adding the screenshots of the work and output alongside the flowchart.



```java
package mapreduce;
import scala.Tuple2;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaRDD;
import org.apache.spark.sql.SparkSession;
import java.util.Arrays;
import java.util.List;
import java.util.regex.Pattern;

public final class SparkFrequencies
{
    private static final Pattern SPACE = Pattern.compile(" ");

    public static void main(String[] args) throws Exception
    {

        if (args.length < 1)
        {
            System.err.println("Usage: JavaWordCount <file>");
            System.exit(1);
        }

        SparkSession spark = SparkSession
                .builder()
                .appName("JavaWordCount")
                .getOrCreate();

```

Shared indexes for maven library "org.apache.spark:spark-sql_2.12:3.0.3" are downloaded (46.91 MB in 10 sec, 965 ms) (10 minutes ago)


C:\Users\AVuser\Desktop\GRM VISHNU\5408\Assignment 4\res\output\part-00000 - Notepad++

```
('', 1595)
('#seattle', 6)
('down', 35)
('out', 159)
('always', 35)
('N', 7)
('isolation', 7)
('tomorrow', 21)
('iPhone","text":"@dabbyjebby', 2)
('walk', 14)
('away', 21)
('Kyiv', 11)
('idea,', 21)
('maybe', 42)
('0.', 14)
('water', 8)
('before', 79)
('accumulati…"},{"created_at":"2022-03-08T12:10:04.000Z","id":"1501168371798265860","author_id":"1427267080307580934","lang":"en","source":"Twitter', 1)
('parallel', 7)
('details', 7)
('https://t.co/EERyHBNmPd"},{"created_at":"2022-03-08T12:09:56.000Z","id":"1501168338151817216","author_id":"1303635713183809536","lang":"en","source":"nana', 1)
('shoreline', 7)
('@SecondNatureMB:', 7)
('head', 14)
('But…"},{"referenced_tweets":[{"type":"retweeted","id":"1501148605251751937"}],"created_at":"2022-03-08T12:09:49.000Z","id":"1501168307311095811","author_id":"1174318621645893632","lang":"en","source":"Twitter', 1)
('33', 7)
('#Ski', 7)
('Sorry', 7)
('produced,', 7)
('da', 14)
('owl', 8)
('mermaid', 7)
('state', 28)
('we're', 14)
('European', 35)
('Union,', 35)
('timeline', 7)
```

Normal text file length : 856,447 lines : 10,405 Ln : 1 Col : 1 Pos : 1 Unix (LF) UTF-8 INS

Step 2:

By performing the MapReduce program, I have found the frequencies of the given words as below,

- flu – 1008
- snow – 777
- cold – 651

So, the highest frequency word is “flu” with 1008 appearances and lowest frequency word is “cold” with 651 appearances.

Task #2: Data Visualization using Graph Database – Neo4j for graph

Step 3:

Neo4j is a graph database management system and Cypher language is used to write the queries which build the graphs in Neo4j.

There are multiple keywords in Neo4j such as CREATE, MATCH, DELETE, RETURN etc.

To develop and write queries using Cypher in Neo4j, we need to get hold of it's terminology first, such as

- Nodes – These are like real world objects or entities
- Labels – They are defined for nodes separation
- Relationship – These define the connection between nodes
- Properties – Any node which has any kind of behavior is called as it's property

Below are the syntaxes for few commands of Cypher query language,

- CREATE (n: label {p: 'property'})
- DELETE (n: label {p: 'property'})
- MATCH (n: label {p: 'property'}) RETURN n
- MATCH (n1: label), (n2: label) WHERE n1.name = 'abc' AND n2.name = 'efg' CREATE (n1)-[r1:HAS_RELATIONSHIP]->(n2) RETURN TYPE (r1)

Step 4:

The below are the queries I executed in Cypher language in order to create a graph.

- MATCH (n) RETURN (n)
- CREATE (k1: Flu {name: 'Flu'})
- CREATE (p1: OtherName {name: 'Influenza'})
- CREATE (p2: Speciality {name: 'Disease'})
- CREATE (p3: Level {name: 'Low'})
- CREATE (p4: Level {name: 'Mild'})
- CREATE (p5: Level {name: 'Severe'})
- CREATE (p6: Type {name: 'Bird Flu'})
- CREATE (p7: Type {name: 'Stomach Flu'})
- CREATE (p8: Type {name: 'Crab Flu'})
- CREATE (p9: Consequence {name: 'Asthma'})
- CREATE (p10: Consequence {name: 'Death'})
- CREATE (p11: DetectedAt {name: 'China'})
- CREATE (p12: DetectedAt {name: 'Spain'})
- CREATE (p13: Cure {name: 'Flu vaccine'})
- CREATE (p14: Cure {name: 'Good ventilation'})
- CREATE (p15: Cure {name: 'Air filters'})
- CREATE (p16: Symptom {name: 'Fever'})
- CREATE (p17: Symptom {name: 'Headache'})
- CREATE (p18: Symptom {name: 'Runny nose'})
- CREATE (p19: Duration {name: '6 to 10 days'})
- CREATE (p20: Year {name: '1918'})
- CREATE (p21: CanAvoidBy {name: 'Wearing a mask'})

- CREATE (p22: Season {name: 'Winter'})
- CREATE (p23: NumberOfPeopleInfected {name: '500 million'})
- CREATE (p24: NumberOfDeaths {name: '50 million'})
- CREATE (k2: Snow {name: 'Snow'})
- CREATE (p25: FormedAs {name: 'Snow flakes'})
- CREATE (p26: FormedAs {name: 'Snow squalls'})
- CREATE (p27: FormedAs {name: 'Snow showers'})
- CREATE (p28: Color {name: 'White'})
- CREATE (p29: Temperature {name: '-10C to -40C'})
- CREATE (p30: ConvertsInTo {name: 'Ice'})
- CREATE (p31: Size {name: '0cm to 20cm'})
- CREATE (p32: Humidity {name: '40% to 100%'})
- CREATE (p33: Height {name: 'Upto 6 feet'})
- CREATE (k3: Cold {name: 'Cold'})
- CREATE (p34: Temperature {name: '10C or less'})
- CREATE (p35: HealthRisks {name: 'Hypothermia'})
- CREATE (p36: HealthRisks {name: 'Frostbite'})
- CREATE (p37: WindSpeed {name: '10 miles per hour or more'})
- CREATE (p38: Precautions {name: 'Dress warmly'})
- MATCH (k1: Flu), (p1: OtherName) WHERE k1.name='Flu' AND p1.name='Influenza' CREATE (k1)-[r:HAS_A_NAME]->(p1) RETURN type(r)
- MATCH (k1: Flu), (p2: Speciality) WHERE k1.name='Flu' AND p2.name='Disease' CREATE (k1)-[r:IS_A]->(p2) RETURN type(r)

- MATCH (k1: Flu), (p3: Level) WHERE k1.name='Flu' AND p3.name='Low' CREATE (k1)-[r:SEVERITY_LEVEL]->(p3) RETURN type(r)
- MATCH (k1: Flu), (p4: Level) WHERE k1.name='Flu' AND p4.name='Mild' CREATE (k1)-[r:SEVERITY_LEVEL]->(p4) RETURN type(r)
- MATCH (k1: Flu), (p5: Level) WHERE k1.name='Flu' AND p5.name='Severe' CREATE (k1)-[r:SEVERITY_LEVEL]->(p5) RETURN type(r)
- MATCH (k1: Flu), (p6: Type) WHERE k1.name='Flu' AND p6.name='Bird Flu' CREATE (k1)-[r:HAS_TYPE]->(p6) RETURN type(r)
- MATCH (k1: Flu), (p7: Type) WHERE k1.name='Flu' AND p7.name='Stomach Flu' CREATE (k1)-[r:HAS_TYPE]->(p7) RETURN type(r)
- MATCH (k1: Flu), (p8: Type) WHERE k1.name='Flu' AND p8.name='Crab Flu' CREATE (k1)-[r:HAS_TYPE]->(p8) RETURN type(r)
- MATCH (k1: Flu), (p9: Consequence) WHERE k1.name='Flu' AND p9.name='Asthma' CREATE (k1)-[r:HAS_CONSEQUENCE_OF]->(p9) RETURN type(r)
- MATCH (k1: Flu), (p10: Consequence) WHERE k1.name='Flu' AND p10.name='Death' CREATE (k1)-[r:HAS_CONSEQUENCE_OF]->(p10) RETURN type(r)
- MATCH (k1: Flu), (p11: DetectedAt) WHERE k1.name='Flu' AND p11.name='China' CREATE (k1)-[r:WAS_FOUND_AT]->(p11) RETURN type(r)

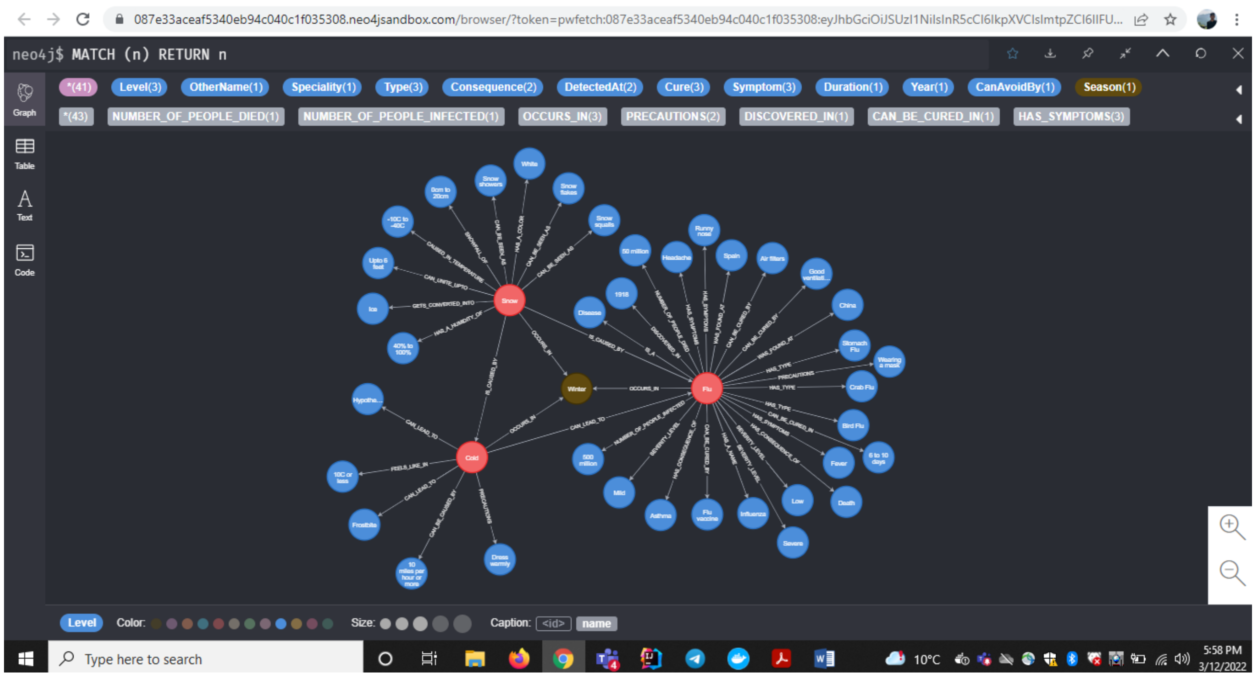
- MATCH (k1: Flu), (p12: DetectedAt) WHERE k1.name='Flu' AND p12.name='Spain' CREATE (k1)-[r:WAS_FOUND_AT]->(p12) RETURN type(r)
- MATCH (k1: Flu), (p13: Cure) WHERE k1.name='Flu' AND p13.name='Flu vaccine' CREATE (k1)-[r:CAN_BE_CURED_BY]->(p13) RETURN type(r)
- MATCH (k1: Flu), (p14: Cure) WHERE k1.name='Flu' AND p14.name='Good ventilation' CREATE (k1)-[r:CAN_BE_CURED_BY]->(p14) RETURN type(r)
- MATCH (k1: Flu), (p15: Cure) WHERE k1.name='Flu' AND p15.name='Air filters' CREATE (k1)-[r:CAN_BE_CURED_BY]->(p15) RETURN type(r)
- MATCH (k1: Flu), (p16: Symptom) WHERE k1.name='Flu' AND p16.name='Fever' CREATE (k1)-[r:HAS_SYMPTOMS]->(p16) RETURN type(r)
- MATCH (k1: Flu), (p17: Symptom) WHERE k1.name='Flu' AND p17.name='Headache' CREATE (k1)-[r:HAS_SYMPTOMS]->(p17) RETURN type(r)
- MATCH (k1: Flu), (p18: Symptom) WHERE k1.name='Flu' AND p18.name='Runny nose' CREATE (k1)-[r:HAS_SYMPTOMS]->(p18) RETURN type(r)
- MATCH (k1: Flu), (p19: Duration) WHERE k1.name='Flu' AND p19.name='6 to 10 days' CREATE (k1)-[r:CAN_BE_CURED_IN]->(p19) RETURN type(r)

- MATCH (k1: Flu), (p20: Year) WHERE k1.name='Flu' AND p20.name='1918' CREATE (k1)-[r:DISCOVERED_IN]->(p20) RETURN type(r)
- MATCH (k1: Flu), (p21: CanAvoidBy) WHERE k1.name='Flu' AND p21.name='Wearing a mask' CREATE (k1)-[r:PRECAUTIONS]->(p21) RETURN type(r)
- MATCH (k1: Flu), (p22: Season) WHERE k1.name='Flu' AND p22.name='Winter' CREATE (k1)-[r:OCCURS_IN]->(p22) RETURN type(r)
- MATCH (k1: Flu), (p23: NumberOfPeopleInfected) WHERE k1.name='Flu' AND p23.name='500 million' CREATE (k1)-[r:NUMBER_OF_PEOPLE_INFECTED]->(p23) RETURN type(r)
- MATCH (k1: Flu), (p24: NumberOfDeaths) WHERE k1.name='Flu' AND p24.name='50 million' CREATE (k1)-[r:NUMBER_OF_PEOPLE_DIED]->(p24) RETURN type(r)
- MATCH (k2: Snow), (p22: Season) WHERE k2.name='Snow' AND p22.name='Winter' CREATE (k2)-[r:OCCURS_IN]->(p22) RETURN type(r)
- MATCH (k2: Snow), (p25: FormedAs) WHERE k2.name='Snow' AND p25.name='Snow flakes' CREATE (k2)-[r:CAN_BE_SEEN_AS]->(p25) RETURN type(r)
- MATCH (k2: Snow), (p26: FormedAs) WHERE k2.name='Snow' AND p26.name='Snow squalls' CREATE (k2)-[r:CAN_BE_SEEN_AS]->(p26) RETURN type(r)

- MATCH (k2: Snow), (p27: FormedAs) WHERE k2.name='Snow' AND p27.name='Snow showers' CREATE (k2)-[r:CAN_BE_SEEN_AS]->(p27) RETURN type(r)
- MATCH (k2: Snow), (p28: Color) WHERE k2.name='Snow' AND p28.name='White' CREATE (k2)-[r:HAS_A_COLOR]->(p28) RETURN type(r)
- MATCH (k2: Snow), (p29: Temperature) WHERE k2.name='Snow' AND p29.name='-10C to -40C' CREATE (k2)-[r:CAUSED_IN_TEMPERATURE]->(p29) RETURN type(r)
- MATCH (k2: Snow), (p30: ConvertsInTo) WHERE k2.name='Snow' AND p30.name='Ice' CREATE (k2)-[r:GETS_CONVERTED_INTO]->(p30) RETURN type(r)
- MATCH (k2: Snow), (p31: Size) WHERE k2.name='Snow' AND p31.name='0cm to 20cm' CREATE (k2)-[r:SNOWFALL_OF]->(p31) RETURN type(r)
- MATCH (k2: Snow), (p32: Humidity) WHERE k2.name='Snow' AND p32.name='40% to 100%' CREATE (k2)-[r:HAS_A_HUMIDITY_OF]->(p32) RETURN type(r)
- MATCH (k2: Snow), (p33: Height) WHERE k2.name='Snow' AND p33.name='Upto 6 feet' CREATE (k2)-[r:CAN_UNITE_UPTO]->(p33) RETURN type(r)
- MATCH (k3: Cold), (p22: Season) WHERE k3.name='Cold' AND p22.name='Winter' CREATE (k3)-[r:OCCURS_IN]->(p22) RETURN type(r)

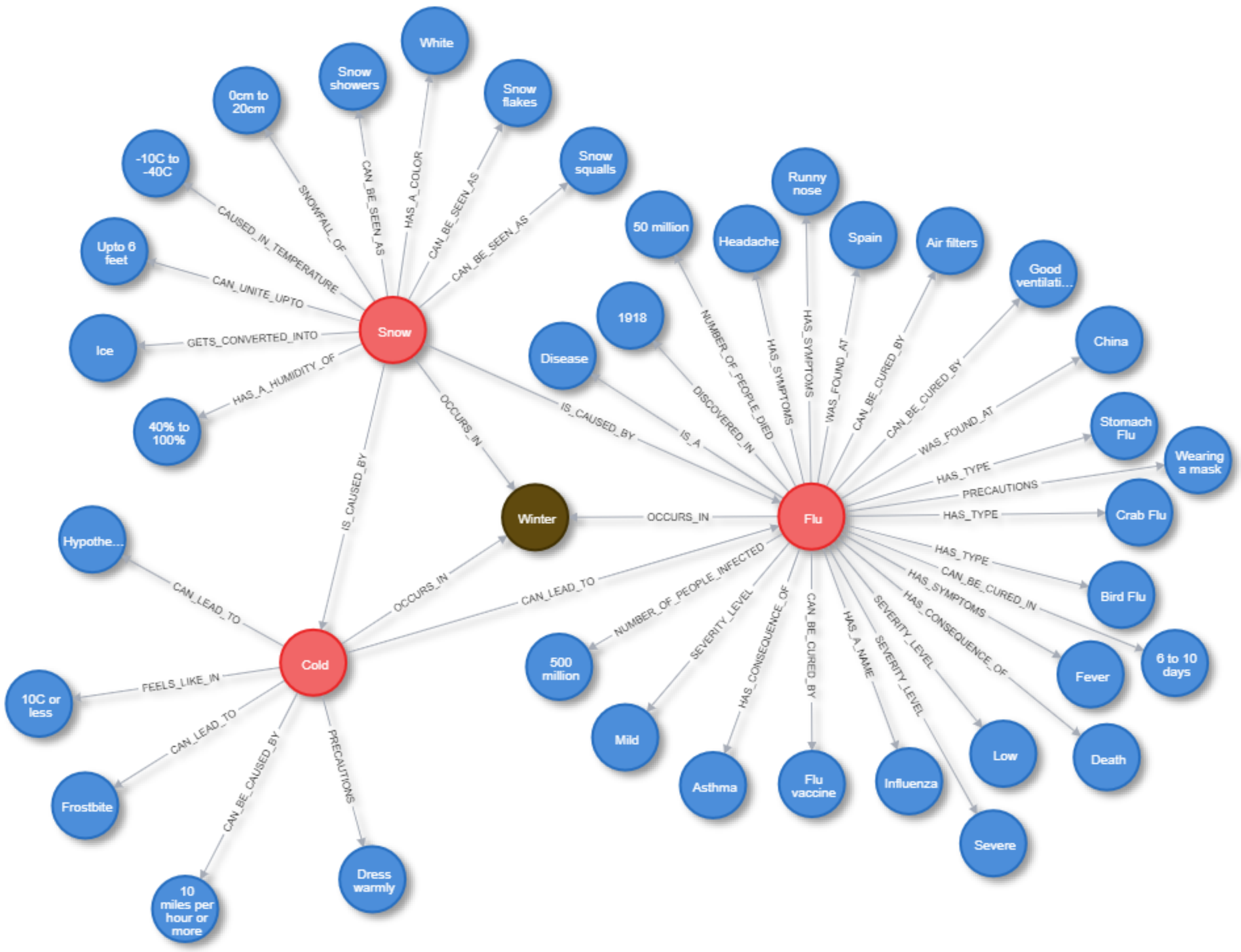
- MATCH (k3: Cold), (p34: Temperature) WHERE k3.name='Cold' AND p34.name='10C or less' CREATE (k3)-[r:FEELS_LIKE_IN]->(p34) RETURN type(r)
- MATCH (k3: Cold), (p35: HealthRisks) WHERE k3.name='Cold' AND p35.name='Hypothermia' CREATE (k3)-[r:CAN_LEAD_TO]->(p35) RETURN type(r)
- MATCH (k3: Cold), (p36: HealthRisks) WHERE k3.name='Cold' AND p36.name='Frostbite' CREATE (k3)-[r:CAN_LEAD_TO]->(p36) RETURN type(r)
- MATCH (k3: Cold), (p37: WindSpeed) WHERE k3.name='Cold' AND p37.name='10 miles per hour or more' CREATE (k3)-[r:CAN_BE_CAUSED_BY]->(p37) RETURN type(r)
- MATCH (k3: Cold), (p38: Precautions) WHERE k3.name='Cold' AND p38.name='Dress warmly' CREATE (k3)-[r:PRECAUTIONS]->(p38) RETURN type(r)
- MATCH (k3: Cold), (k2: Snow) WHERE k3.name='Cold' AND k2.name='Snow' CREATE (k3)<-[r:IS_CAUSED_BY]-(k2) RETURN type(r)
- MATCH (k1: Flu), (k2: Snow) WHERE k1.name='Flu' AND k2.name='Snow' CREATE (k1)<-[r:IS_CAUSED_BY]-(k2) RETURN type(r)
- MATCH (k1: Flu), (k3: Cold) WHERE k1.name='Flu' AND k3.name='Cold' CREATE (k1)<-[r:CAN_LEAD_TO]-(k3) RETURN type(r)

Screenshot of the work I have done in Neo4j web application for this assignment.



```
neo4j$ MATCH (n) RETURN n
```

Original image downloaded from Neo4j.



References:

[1] "*Clauses - Neo4j Cypher Manual*," Neo4j Graph Database Platform [Online]. Available at: https://neo4j.com/docs/cypher-manual/current/clauses/ [Accessed: March 12, 2022].