

KATHMANDU UNIVERSITY
SCHOOL OF ENGINEERING



DEPARTMENT OF GEOMATICS ENGINEERING
MINI PROJECT REPORT
SPATIAL DATABASE MANAGEMENT SYSTEM GEOM 318

Submitted By:

Bipul Chaudhary
Roll no.: 08
Dept. of GE

Submitted To:

Er. Ajay Thapa
Dept of GE

ACKNOWLEDGEMENT:

I would like to express my sincere gratitude towards the Department of Geomatics Engineering (DOGE). A special appreciation for our SPATIAL DATABASE MANAGEMENT SYSTEM faculty teacher ER. AJAY THAPA sir for his valuable contribution towards providing us with the knowledge and training of software like Postgres, Qgis and many other integrated tools. His supervision has led us to complete our mini project with such efficiency.

Thank you!

ABSTRACT:

This is a mini project report depicting the crucial principles of spatial queries and its application on various software. This project generates the shortest route between the origin(start) and the destination using spatial queries. Different software tools were used such as Postgres, Q, POSTGIS, JAVA. Data such as OSM generated data in both OSM2PO format and .PBF format were also crucial for the execution. The data obtained were used for the determination of shortest route through queries. pgRouting is an extension of the PostGIS/PostgreSQL geospatial database that provides geospatial routing functionality. PostgreSQL, often referred to as Postgres, is a powerful, open-source object-relational database system known for its robustness, feature set, and extensibility. PostGIS is a spatial database extender for PostgreSQL, turning it into a spatial database for geographic information systems (GIS). It adds support for geographic objects, allowing location queries to be run in SQL. SQL (Structured Query Language) is a standard programming language specifically designed for managing and manipulating relational databases. It is widely used for querying, updating, and managing data stored in a database. QGIS (Quantum GIS) is a powerful, free, and open-source geographic information system (GIS) application used for viewing, editing, and analyzing geospatial data. It supports a wide range of vector, raster, and database formats and functionalities, making it a popular choice for both professional and academic use. Spatial data, also known as geospatial data, is information about the physical location and shape of geometric objects. These objects can be points, lines, polygons, and other complex structures representing natural and man-made features on Earth. Spatial data is fundamental to geographic information systems (GIS) and is used in a wide range of applications, from mapping and navigation to environmental monitoring and urban planning

Contents

1	INTRODUCTION.....	1
1.1	BACKGROUND	1
1.2	OBJECTIVES.....	2
2	METHODOLOGIES.....	2
3	PERFORMED OPERATIONS:	4
3.1	DATA ACQUISITIONS:.....	4
4	CONCLUSION:	16
5	References	17

1 INTRODUCTION

1.1 BACKGROUND

pgRouting is an extension of the PostGIS/PostgreSQL geospatial database that provides geospatial routing functionality. It is designed to work with the PostgreSQL database and the PostGIS spatial extension, enabling complex routing algorithms to be executed directly within the database. (Puyam S. Singh, 2015). Here are some key features and capabilities of pgRouting:

- i. **Routing Algorithms:** pgRouting supports a variety of routing algorithms eg.: Dijkstra.
- ii. **Integration with postgis:** The spatial capabilities of postgis is integrated with pgrouting.
- iii. **Network Analysis:** It can perform network analysis operation such as finding the shortest route between two points.

Terminologies:

PostgreSQL: PostgreSQL, often referred to as Postgres, is a powerful, open-source object-relational database system known for its robustness, feature set, and extensibility. (Puyam S. Singh, 2015)

PostGIS: PostGIS is a spatial database extender for PostgreSQL, turning it into a spatial database for geographic information systems (GIS). It adds support for geographic objects, allowing location queries to be run in SQL. (Regina O. Obe, 2017)

QGIS: QGIS (Quantum GIS) is a powerful, free, and open-source geographic information system (GIS) application used for viewing, editing, and analyzing geospatial data. It supports a wide range of vector, raster, and database formats and functionalities, making it a popular choice for both professional and academic use (Muñoz, 2020).

SQL: SQL (Structured Query Language) is a standard programming language specifically designed for managing and manipulating relational databases. It is widely used for querying, updating, and managing data stored in a database (Muñoz, 2020).

Spatial Data: Spatial data, also known as geospatial data, is information about the physical location and shape of geometric objects. These objects can be points, lines, polygons, and other complex structures representing natural and man-made features on Earth. Spatial data is fundamental to geographic information systems (GIS) and is used in a wide range of applications, from mapping and navigation to environmental monitoring and urban planning (Ramesh Janipella, Spatial Data, 2019).

Network Analysis: Network analysis of spatial data involves examining the relationships and interactions between spatial entities within a geographic space. This type of analysis is crucial in various fields, such as transportation planning, urban development, environmental management, and public safety. It often requires integrating geographic information systems (GIS) with network analysis tools to derive meaningful insights from spatial networks (Smiraglia, 2001).

1.2 OBJECTIVES

Primary Objective:

The primary objective of this project is:

- i. To find out the shortest distance between the origin and destination.
- i. To analyze the spatial queries for the shortest route determination.

Secondary Objective:

The secondary objective of the project includes:

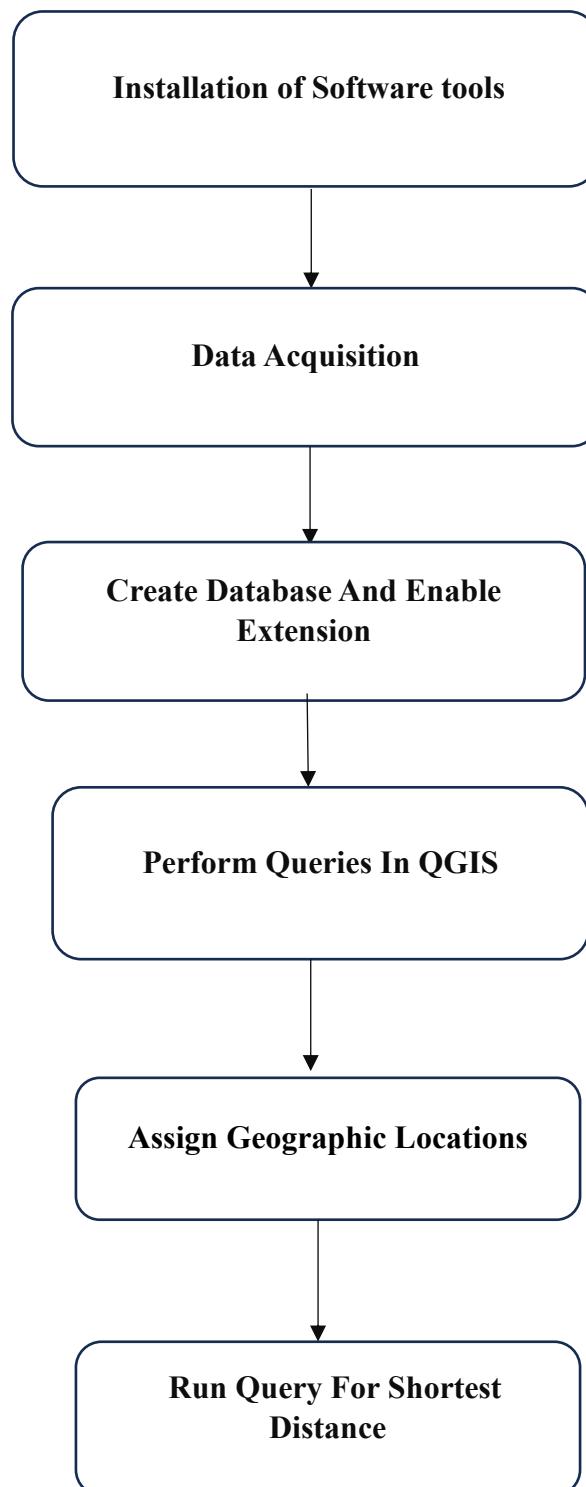
- i. To become familiar with the Postgres, Postgis, Qgis tools.
- ii. To be able to work with OSM data.

2 METHODOLOGIES

The methodologies involved for performing pgrouting are as follows:

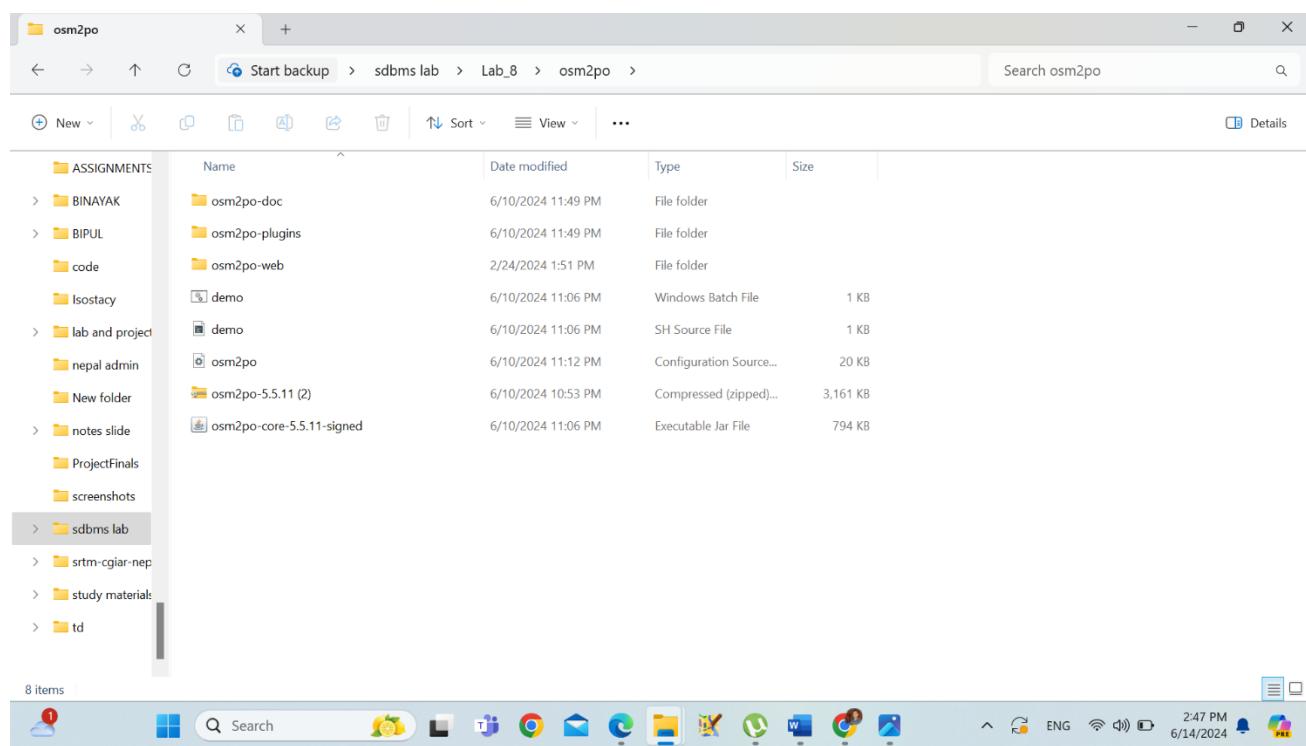
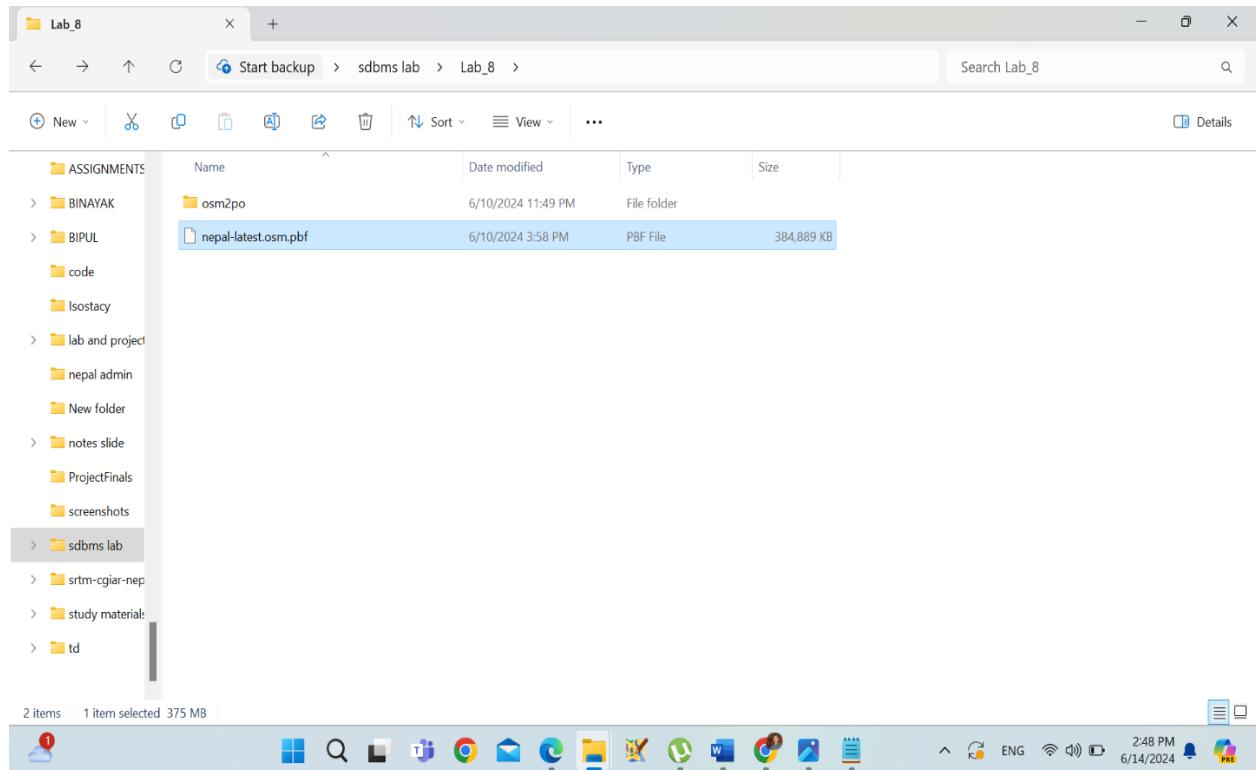
1. Installation:
 - To perform pgrouting we need to first install Postgres and Postgis in our devices.
 - i. Installation of PostgreSQL: <https://www.postgresql.org/download/>
 - ii. PostGIS: https://postgis.net/documentation/getting_started/install_windows/
 - iii. QGIS: <https://www.qgis.org/en/site/forusers/download.html>
 - iv. JAVA: <https://www.java.com/en/download/>
2. Data Acquisition:
 - Different spatial data acquisition are to be done before conducting the process:
 - i. OSM2PO: <https://osm2po.de/releases/>
 - ii. OSM data in .PBF format: <https://download.geofabrik.de/asia.html>
3. Create a Database and Enable extension
4. Perform spatial queries.
5. In QGIS, DB manager write a query to assign the location for the origin and destination.
6. Perform the query for the origin location.
7. Perform the query for the destination location.
8. Finally perform the overall query to find out the shortest path between the origin and destination (Puyam S. Singh, 2015) (Smiraglia, 2001).

FLOW DIAGRAM:



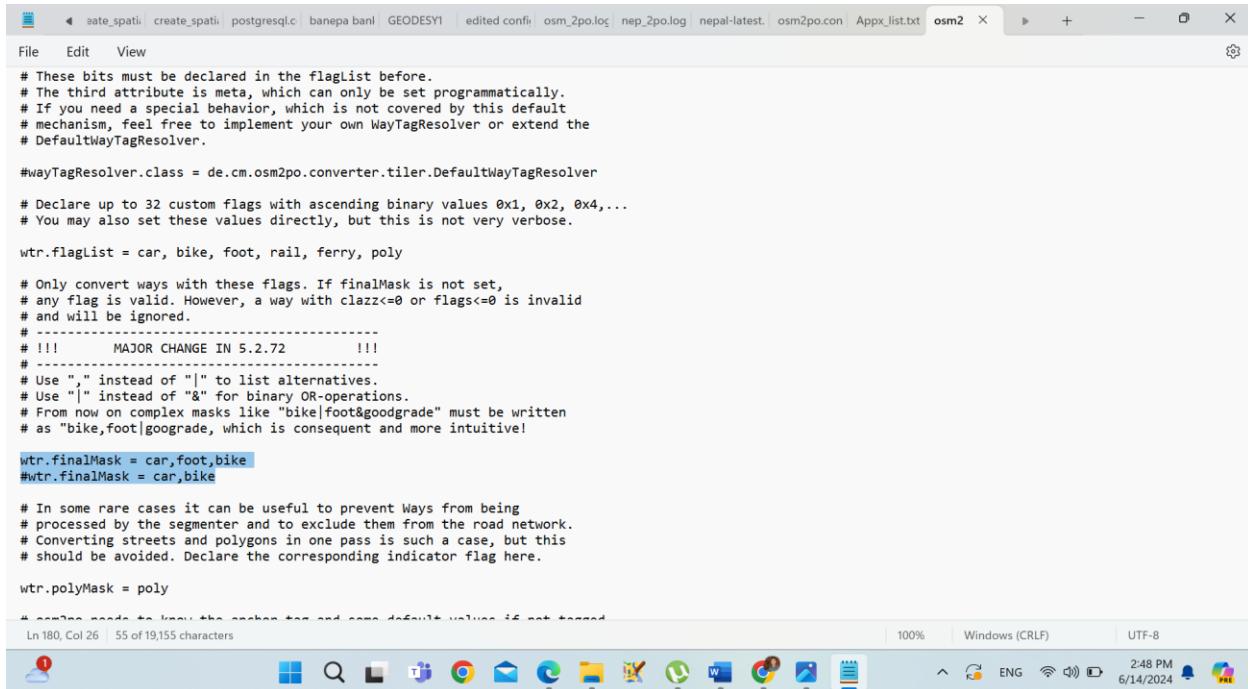
3 PERFORMED OPERATIONS:

3.1 DATA ACQUISITIONS:



Editing the configuration file in the notepad:

Opened the configuration file from OSM2P0 in the note pad and the edited the line 179 and 330 as mentioned.



```
# These bits must be declared in the flagList before.
# The third attribute is meta, which can only be set programmatically.
# If you need a special behavior, which is not covered by this default
# mechanism, feel free to implement your own WayTagResolver or extend the
# DefaultWayTagResolver.

#wayTagResolver.class = de.cm.osm2po.converter.tiler.DefaultWayTagResolver

# Declare up to 32 custom flags with ascending binary values 0x1, 0x2, 0x4,...
# You may also set these values directly, but this is not very verbose.

wtr.flagList = car, bike, foot, rail, ferry, poly

# Only convert ways with these flags. If finalMask is not set,
# any flag is valid. However, a way with clazz<=0 or flags<=0 is invalid
# and will be ignored.

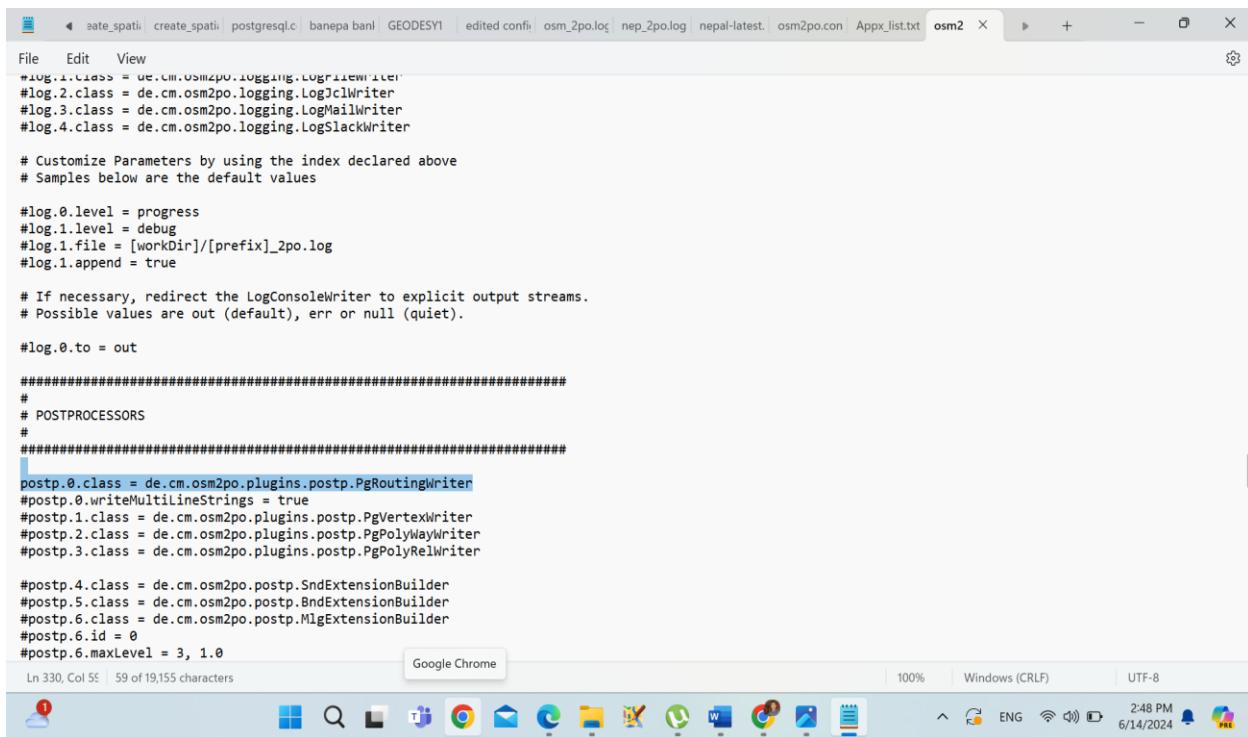
# -----
# !!!      MAJOR CHANGE IN 5.2.72      !!!
# -----
# Use ',' instead of "||" to list alternatives.
# Use "[" instead of "&" for binary OR-operations.
# From now on complex masks like "bike|foot&goodgrade" must be written
# as "bike,foot|goodgrade", which is consequent and more intuitive!

wtr.finalMask = car,foot,bike
#wtr.finalMask = car,bike

# In some rare cases it can be useful to prevent Ways from being
# processed by the segmenter and to exclude them from the road network.
# Converting streets and polygons in one pass is such a case, but this
# should be avoided. Declare the corresponding indicator flag here.

wtr.polyMask = poly

# osm2po needs to know the anchor tag and some default values if not tagged
```



```
#log.1.class = ue.cm.osm2po.logging.LogFileWriter
#log.2.class = de.cm.osm2po.logging.LogJclWriter
#log.3.class = de.cm.osm2po.logging.LogMailWriter
#log.4.class = de.cm.osm2po.logging.LogSlackWriter

# Customize Parameters by using the index declared above
# Samples below are the default values

#log.0.level = progress
#log.1.level = debug
#log.1.file = [workDir]/[prefix]_2po.log
#log.1.append = true

# If necessary, redirect the LogConsoleWriter to explicit output streams.
# Possible values are out (default), err or null (quiet).

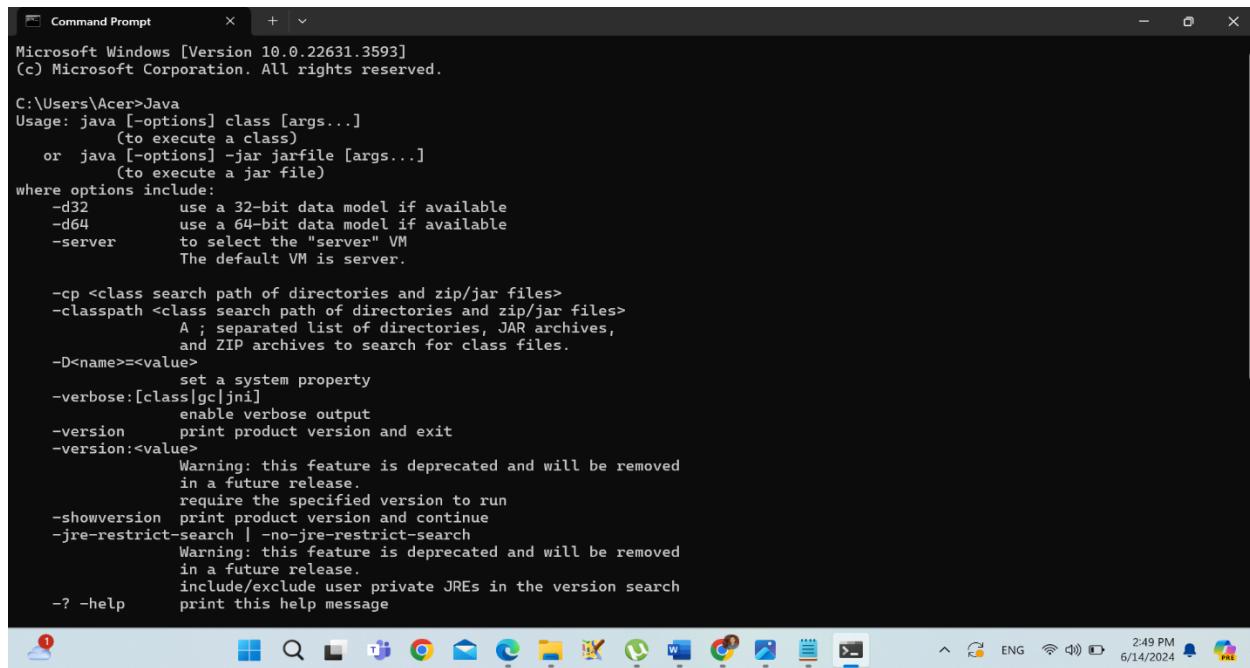
#log.0.to = out

#####
#
# POSTPROCESSORS
#
#####

postp.0.class = de.cm.osm2po.plugins.postp.PgRoutingWriter
#postp.0.writeMultilineStrings = true
#postp.1.class = de.cm.osm2po.plugins.postp.PgVertexWriter
#postp.2.class = de.cm.osm2po.plugins.postp.PgPolyWayWriter
#postp.3.class = de.cm.osm2po.plugins.postp.PgPolyRelWriter

#postp.4.class = de.cm.osm2po.postp.SndExtensionBuilder
#postp.5.class = de.cm.osm2po.postp.BndExtensionBuilder
#postp.6.class = de.cm.osm2po.postp.MlgExtensionBuilder
#postp.6.id = 0
#postp.6.maxLevel = 3, 1.0
```

Checking if java is installed or not:



```
Microsoft Windows [Version 10.0.22631.3593]
(c) Microsoft Corporation. All rights reserved.

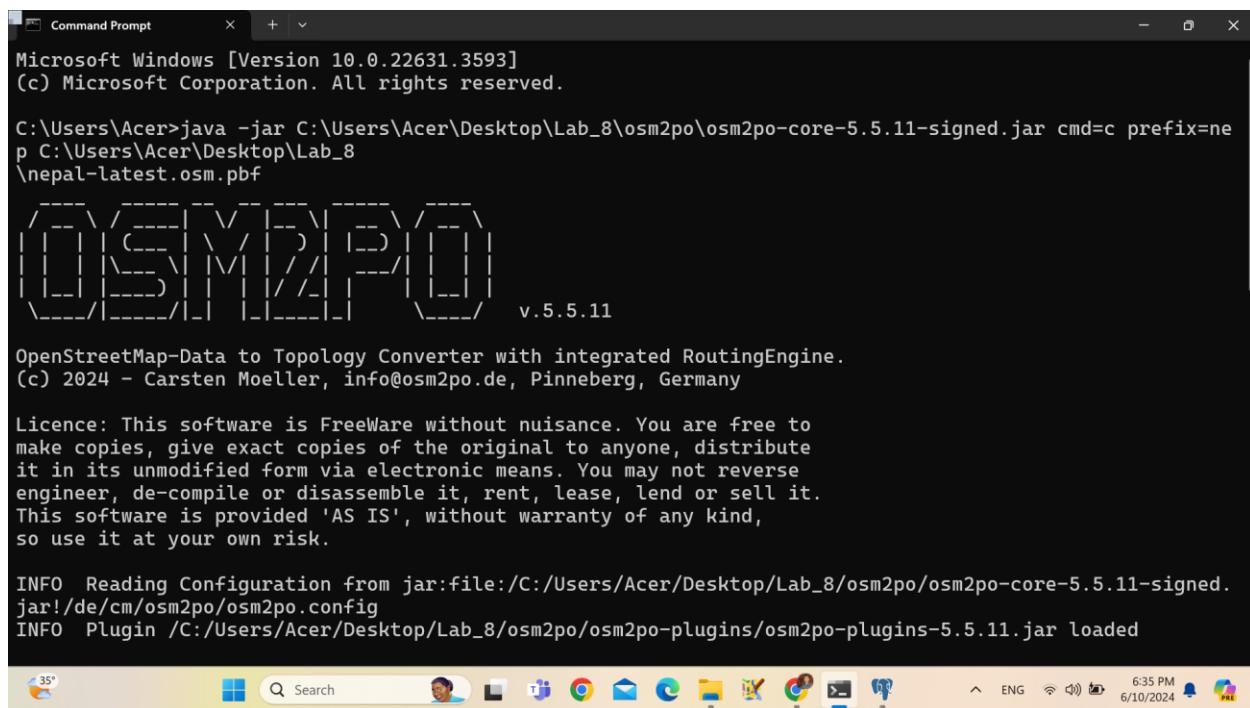
C:\Users\Acer>Java
Usage: java [-options] class [args...]
            (to execute a class)
        or  java [-options] -jar jarfile [args...]
            (to execute a jar file)
where options include:
-d32      use a 32-bit data model if available
-d64      use a 64-bit data model if available
-server    to select the "server" VM
           The default VM is server.

-cp <class search path of directories and zip/jar files>
-classpath <class search path of directories and zip/jar files>
           A ; separated list of directories, JAR archives,
           and ZIP archives to search for class files.

-D<name>=<value>
           set a system property
-verbose:[class|gc|jni]
           enable verbose output
-version   print product version and exit
-version:<value>
           Warning: this feature is deprecated and will be removed
           in a future release.
           require the specified version to run
-showversion print product version and continue
-jre-restrict-search | -no-jre-restrict-search
           Warning: this feature is deprecated and will be removed
           in a future release.
           include/exclude user private JREs in the version search
-? -help   print this help message
```

This shows java is completely installed.

Pre-processing of OSM data in OSM2PO:



```
Microsoft Windows [Version 10.0.22631.3593]
(c) Microsoft Corporation. All rights reserved.

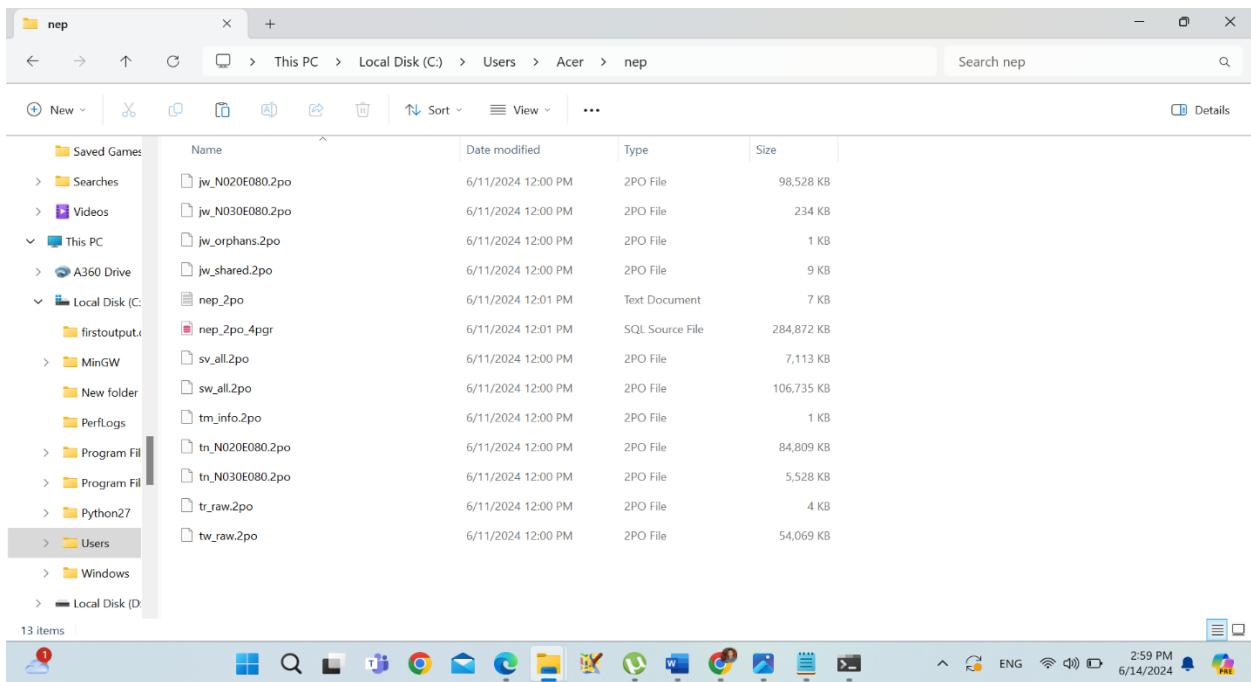
C:\Users\Acer>java -jar C:\Users\Acer\Desktop\Lab_8\osm2po\osm2po-core-5.5.11-signed.jar cmd=c prefix=ne
p C:\Users\Acer\Desktop\Lab_8
\nepal-latest.osm.pbf
v.5.5.11

OpenStreetMap-Data to Topology Converter with integrated RoutingEngine.
(c) 2024 - Carsten Moeller, info@osm2po.de, Pinneberg, Germany

Licence: This software is FreeWare without nuisance. You are free to
make copies, give exact copies of the original to anyone, distribute
it in its unmodified form via electronic means. You may not reverse
engineer, de-compile or disassemble it, rent, lease, lend or sell it.
This software is provided 'AS IS', without warranty of any kind,
so use it at your own risk.

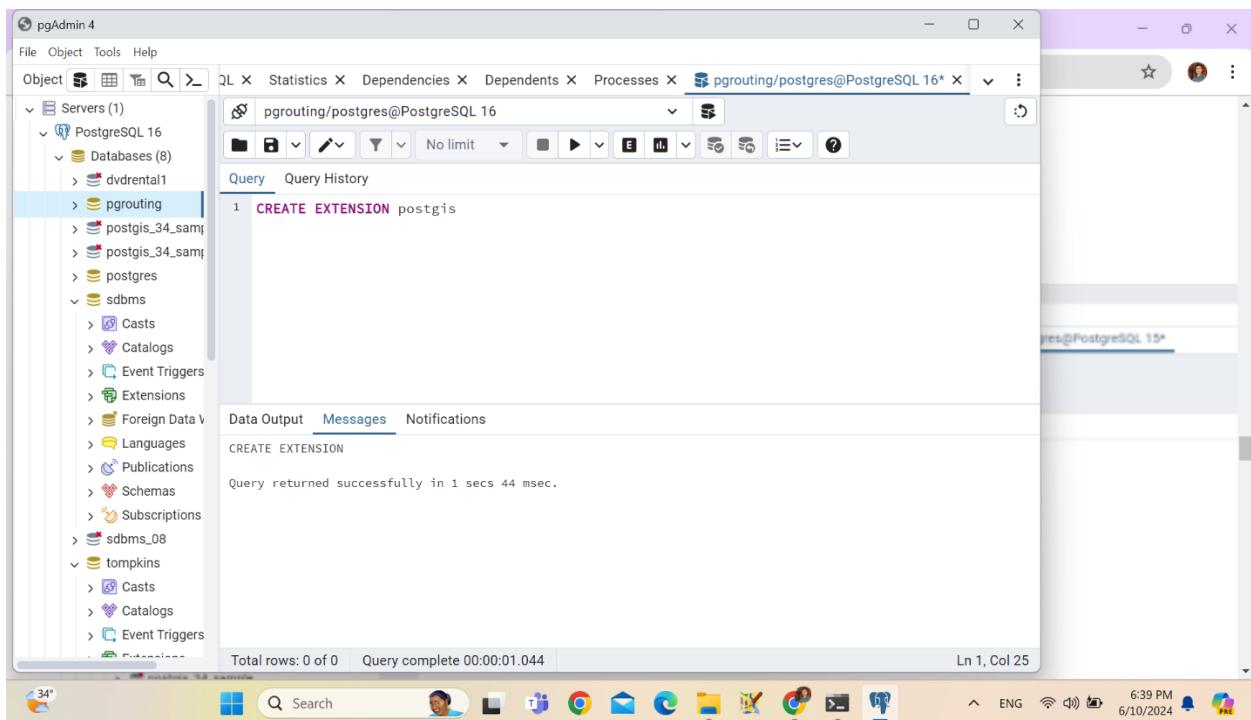
INFO  Reading Configuration from jar:file:/C:/Users/Acer/Desktop/Lab_8/osm2po/osm2po-core-5.5.11-signed.
jar!/de/cm/osm2po/osm2po.config
INFO  Plugin /C:/Users/Acer/Desktop/Lab_8/osm2po/osm2po-plugins/osm2po-plugins-5.5.11.jar loaded
```

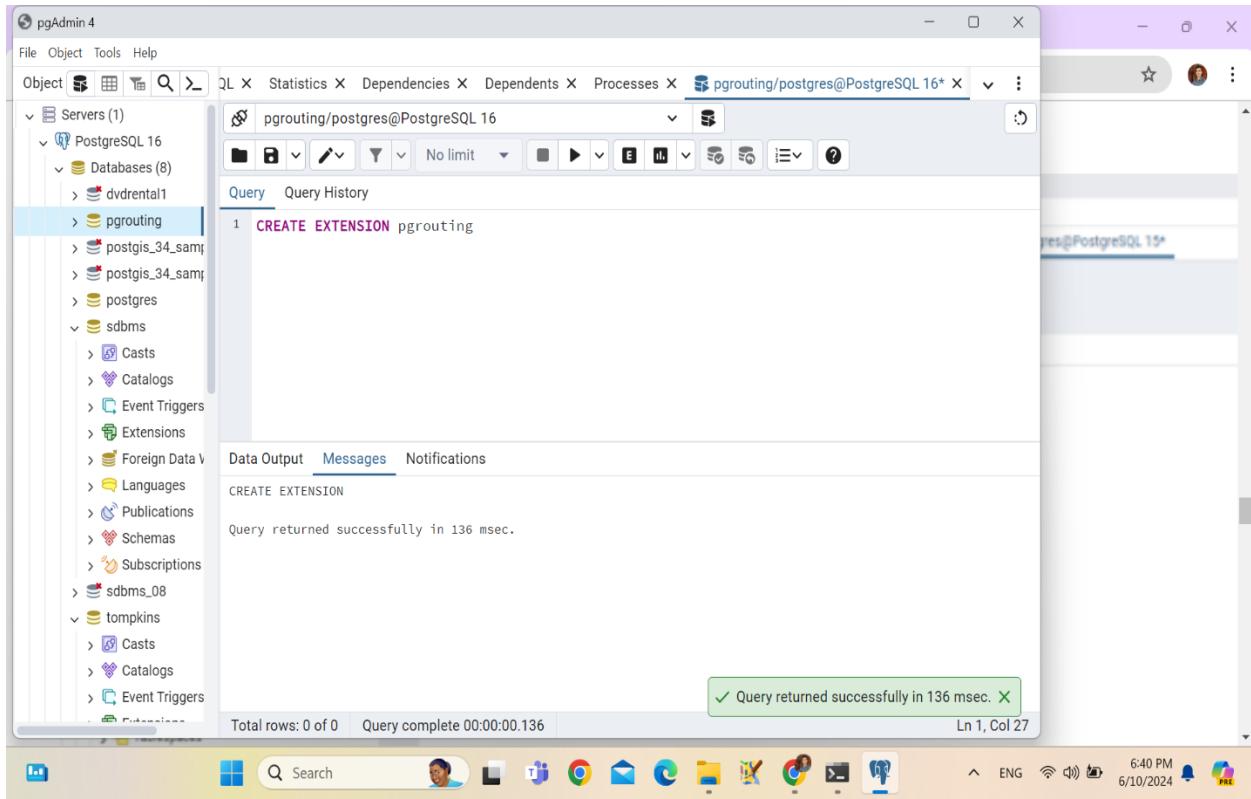
Provides with the location of converted SQL file.



Creating Database pgrouting in pgAdmin and Extension Postgis and pgrouting:

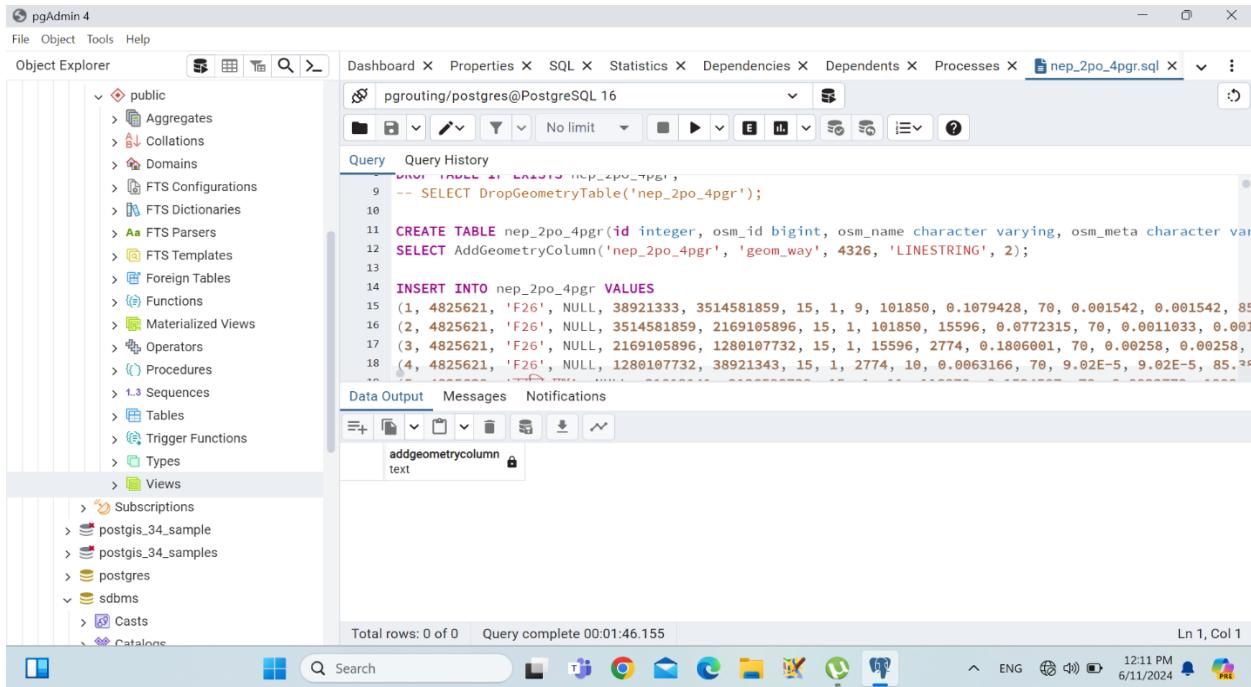
Go to pgAdmin> Click on database> create database pgrouting> create extensions postgis and pgrouting.





Importing SQL file:

Click Alt + O in query tool> Add the sql file.



pgAdmin 4

File Object Tools Help

Dependencies X Dependents X Processes X nep_2po_4pgr.sql X public.nep_2po_4pgr/pgRouting/postgres@PostgreSQL 16 X

Object Explorer

- public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (2)
 - nep_2po_4pgr
 - spatial_ref_sys
 - Trigger Functions
 - Types
 - Views
 - Subscriptions
 - postgis_34_sample
 - postgis_34_samples
 - postgres
- edbms

Query Query History

```
1 SELECT * FROM public.nep_2po_4pgr
2 ORDER BY id ASC
```

Data Output Messages Notifications

	id [PK] integer	osm_id bigint	osm_name character varying	osm_meta character varying	osm_source_id bigint	osm_target_id bigint	clazz integer	fl integer
1	1	4825621	F26	[null]	38921333	3514581859	15	1
2	2	4825621	F26	[null]	3514581859	2169105896	15	1
3	3	4825621	F26	[null]	2169105896	1280107732	15	1
4	4	4825621	F26	[null]	1280107732	38921343	15	1
5	5	4825630	कान्ति पथ	[null]	31019141	2126598729	15	1

Total rows: 1000 of 397414 Query complete 00:00:03.053 Ln 2, Col 17

Search 12:20 PM 6/11/2024 ENG

pgAdmin 4

File Object Tools Help

Dependencies X Dependents X Processes X public.nep_2po_4pgr/pgRouting/postgres@PostgreSQL 16 X

Object Explorer

- public
 - Aggregates
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions
 - Materialized Views
 - Operators
 - Procedures
 - Sequences
 - Tables (2)
 - nep_2po_4pgr
 - spatial_ref_sys
 - Trigger Functions
 - Types
 - Views
 - Subscriptions
 - postgis_34_sample
 - postgis_34_samples
 - postgres
 - Casts

Query Query History

```
1 SELECT * FROM public.nep_2po_4pgr
2 ORDER BY id ASC
```

Scratch Pad X

Data Output Messages Geometry Viewer X Notifications

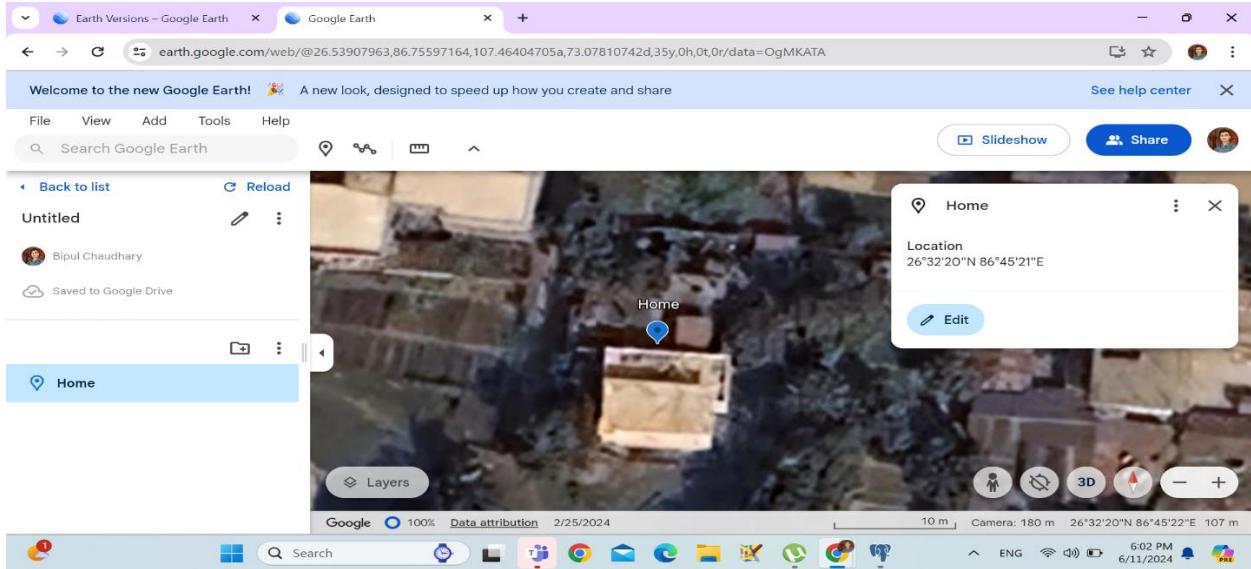
Total rows: 1000 of 397414 Query complete 00:00:02.317 Ln 1, Col 1

Search 5:41 PM 6/11/2024 ENG

Table of the SQL file created and through the column,with the geometry viewer tab above geographic data is obtained.

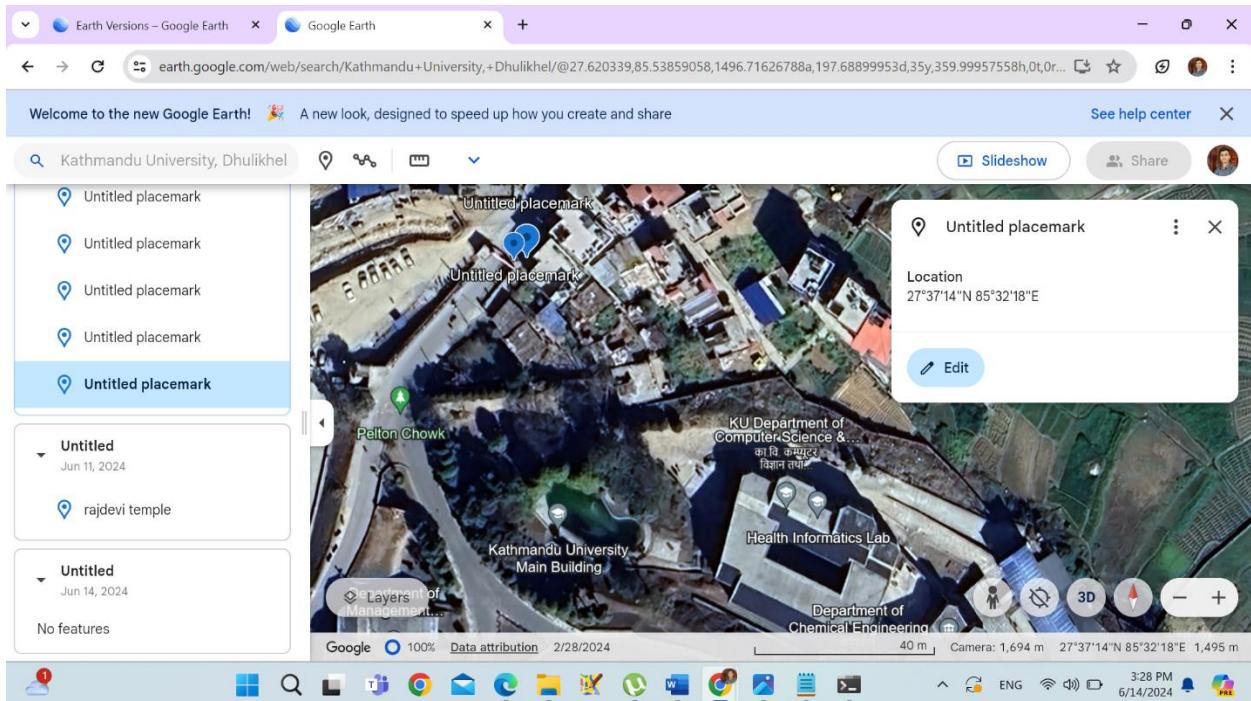
Location Of Origin(start):

This is the geographic location of my home which is located in Rajbiraj, saptari ,Nepal. This location is obtained by navigating with the help of Google Earth tool.



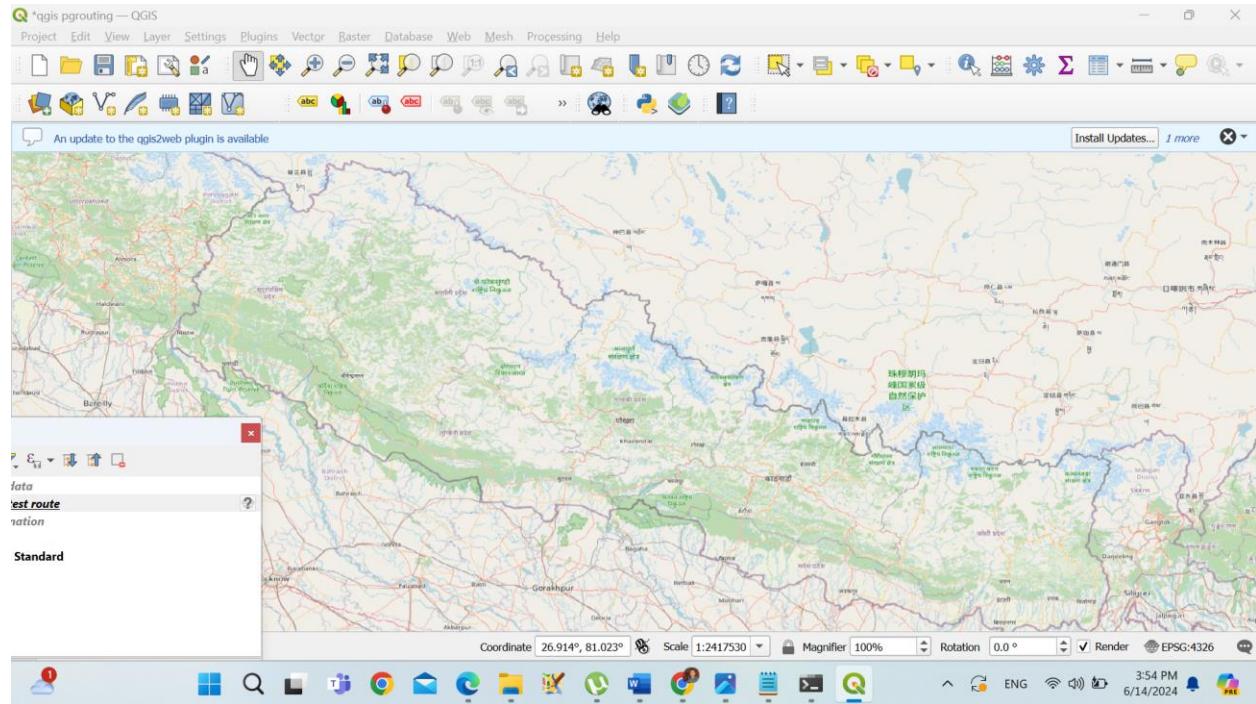
Location Of Destination:

This is the geographic location of KATHMANDU UNIVERSITY, Dhulikhel which is located in Dhulikhel, Kavre ,Nepal. This location is obtained by navigating with the help of Google Earth tool.



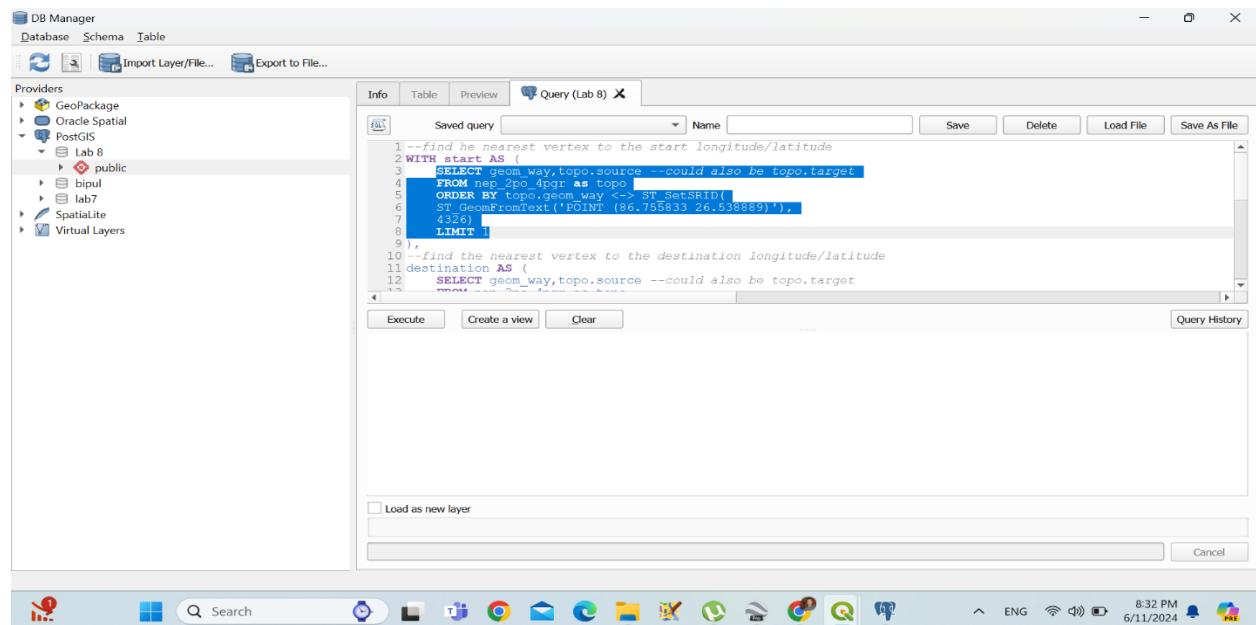
OSM DATA plugin:

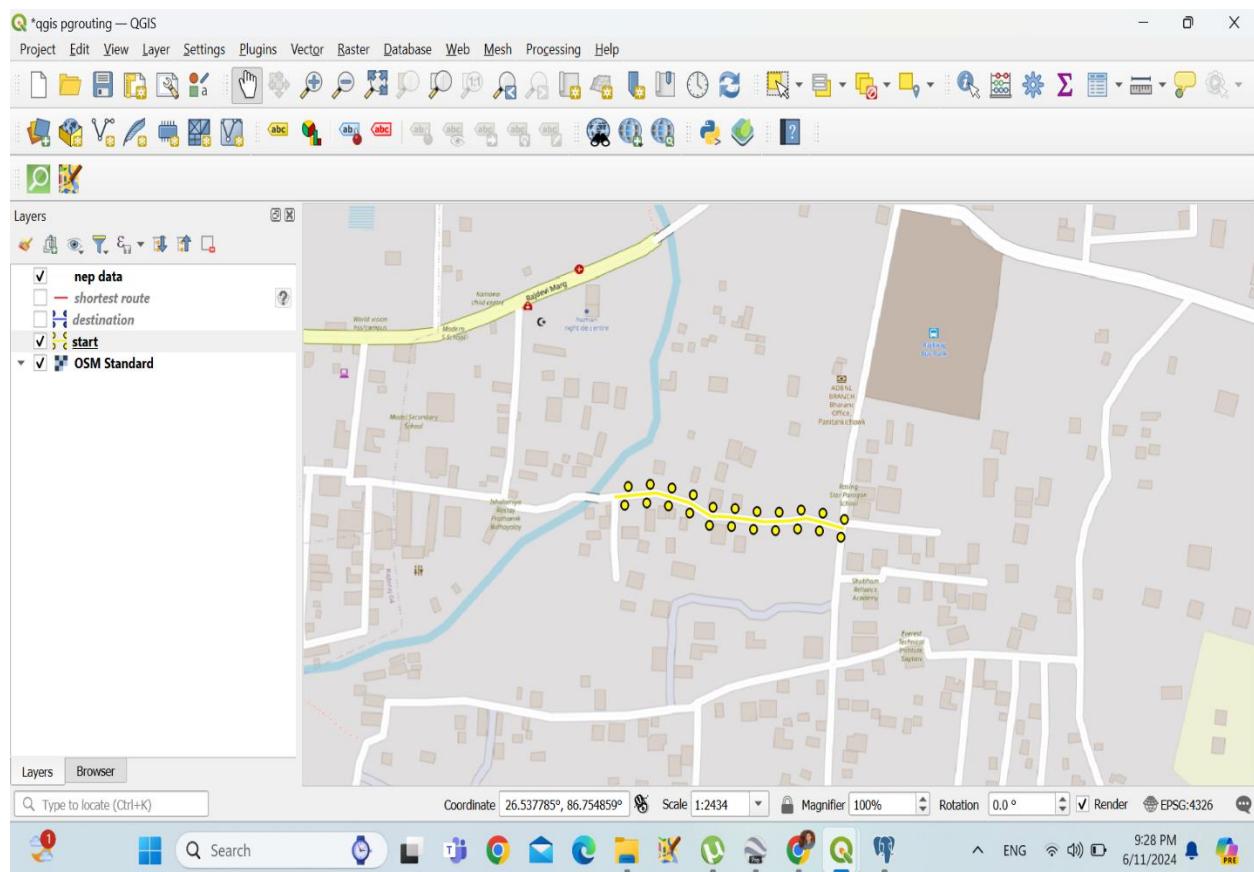
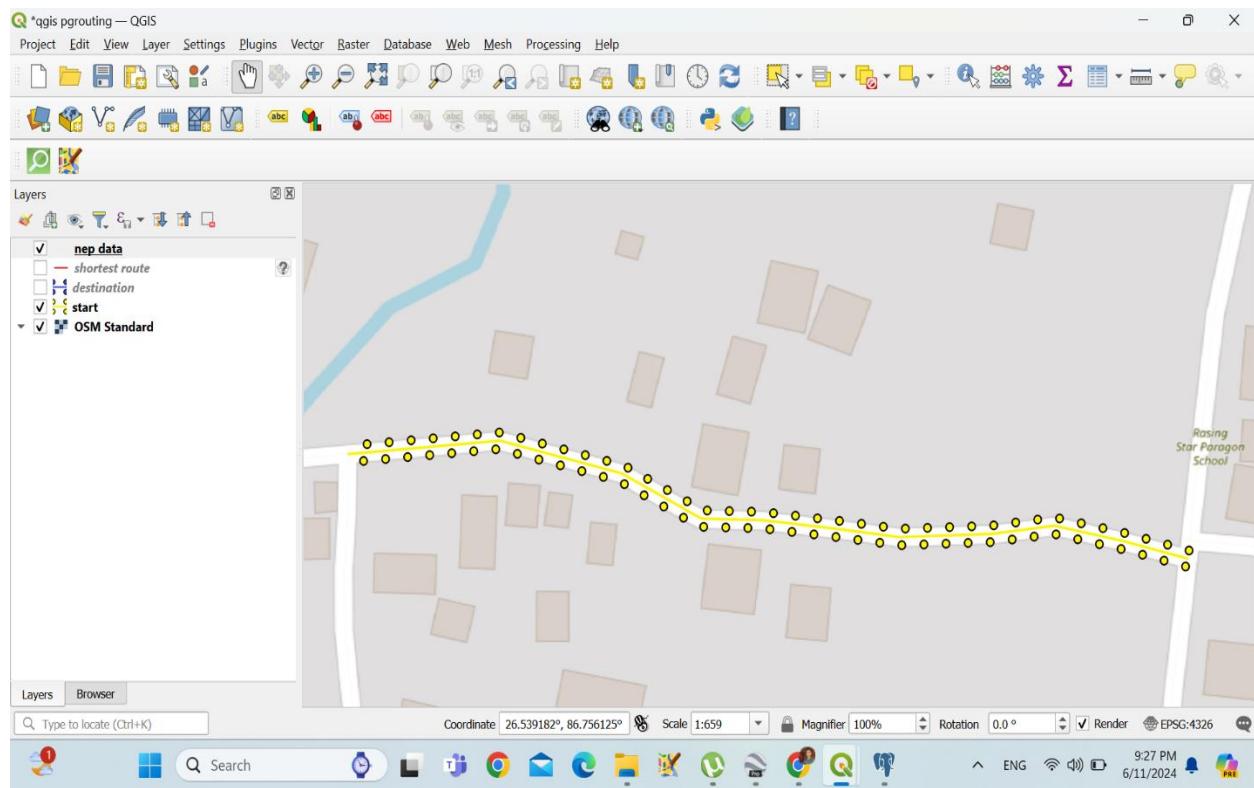
Open QGIS > in Browser> Add new connection to the Database crated in pgAdmin> Go To DB Manager> Write spatial query mentioned below:



For start:

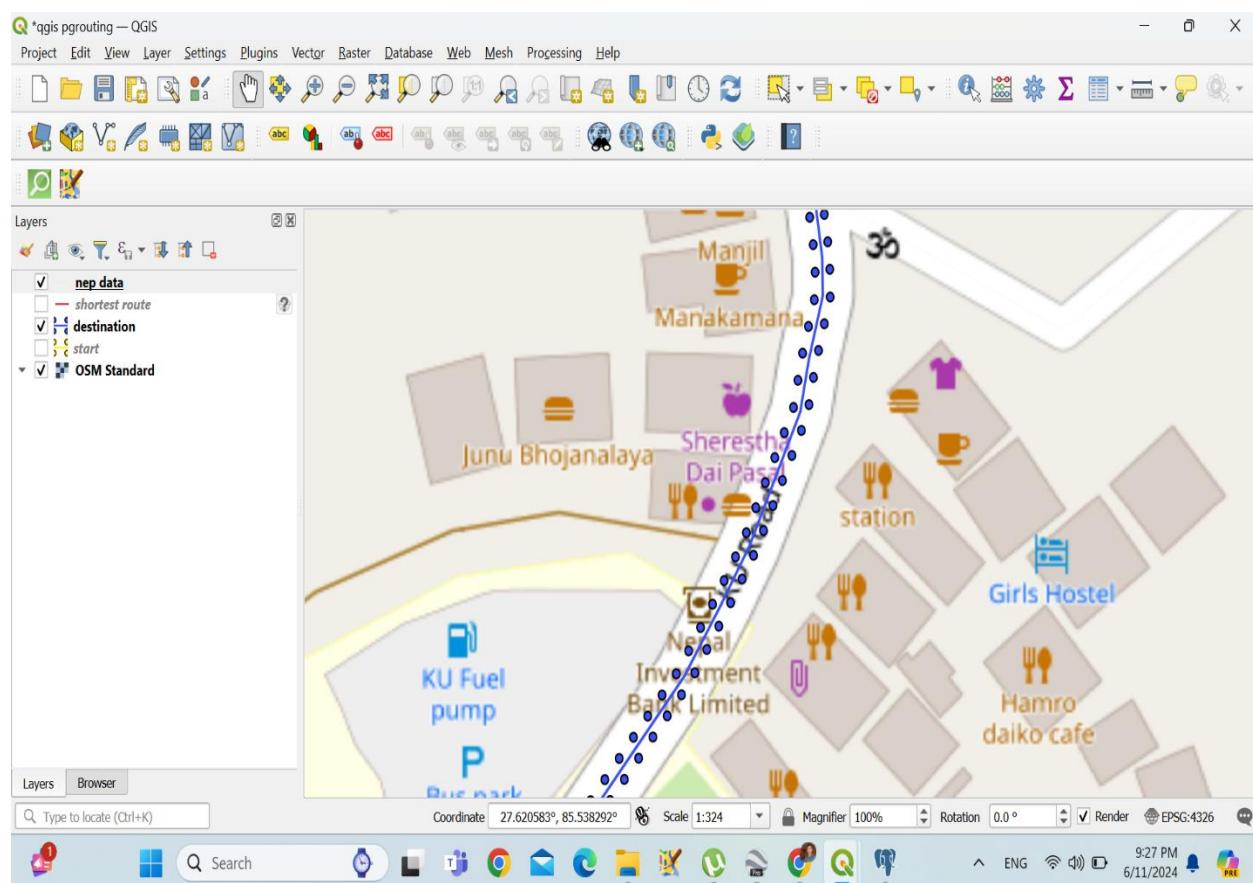
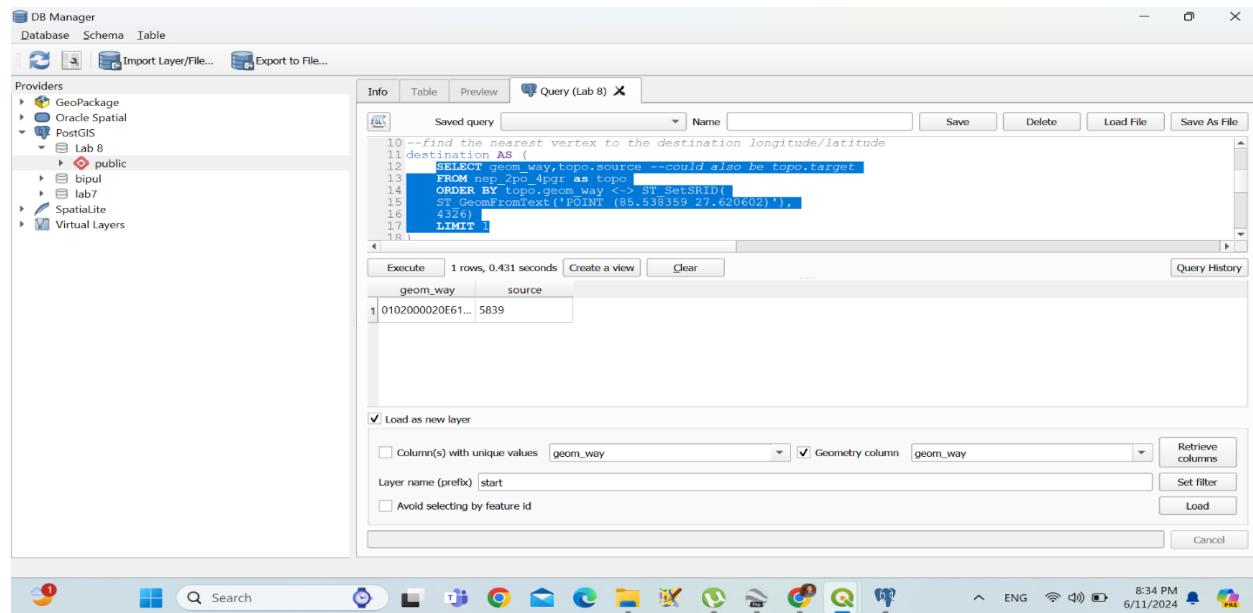
Write the required query> execute only for the origin> load as layer.





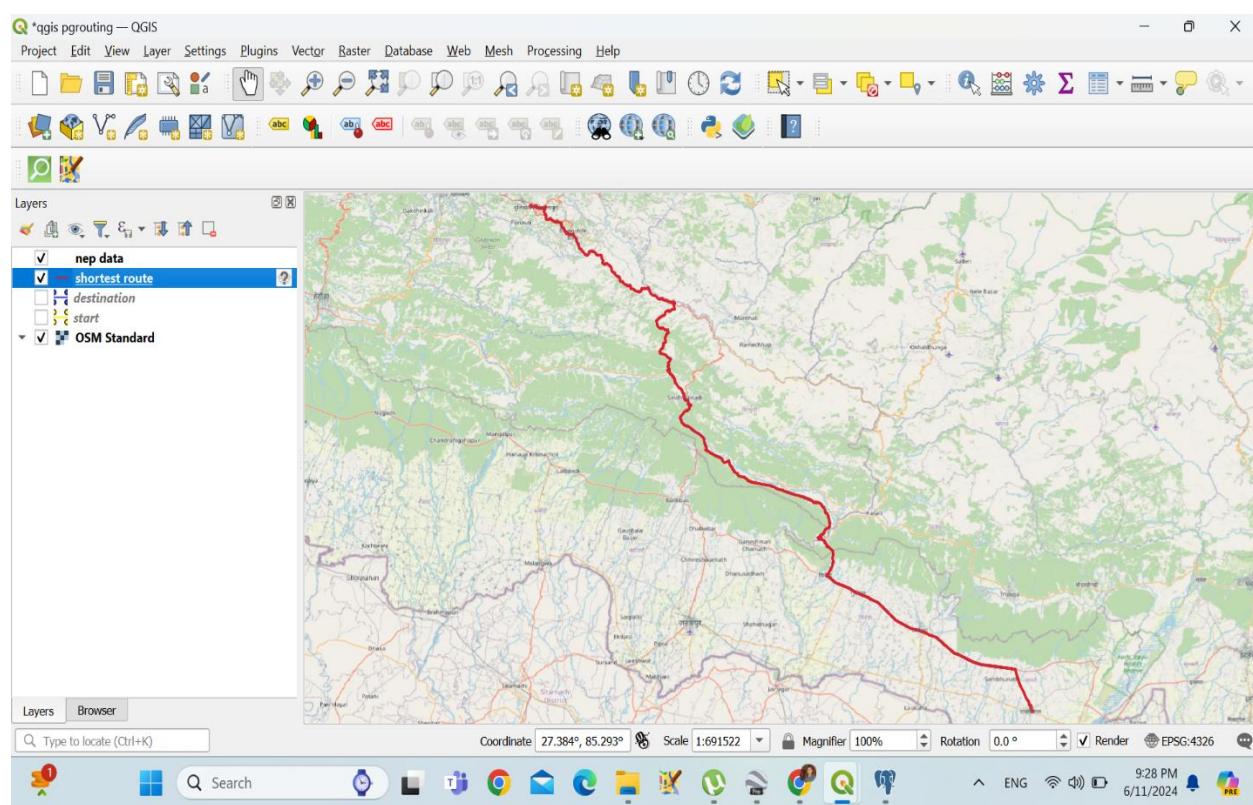
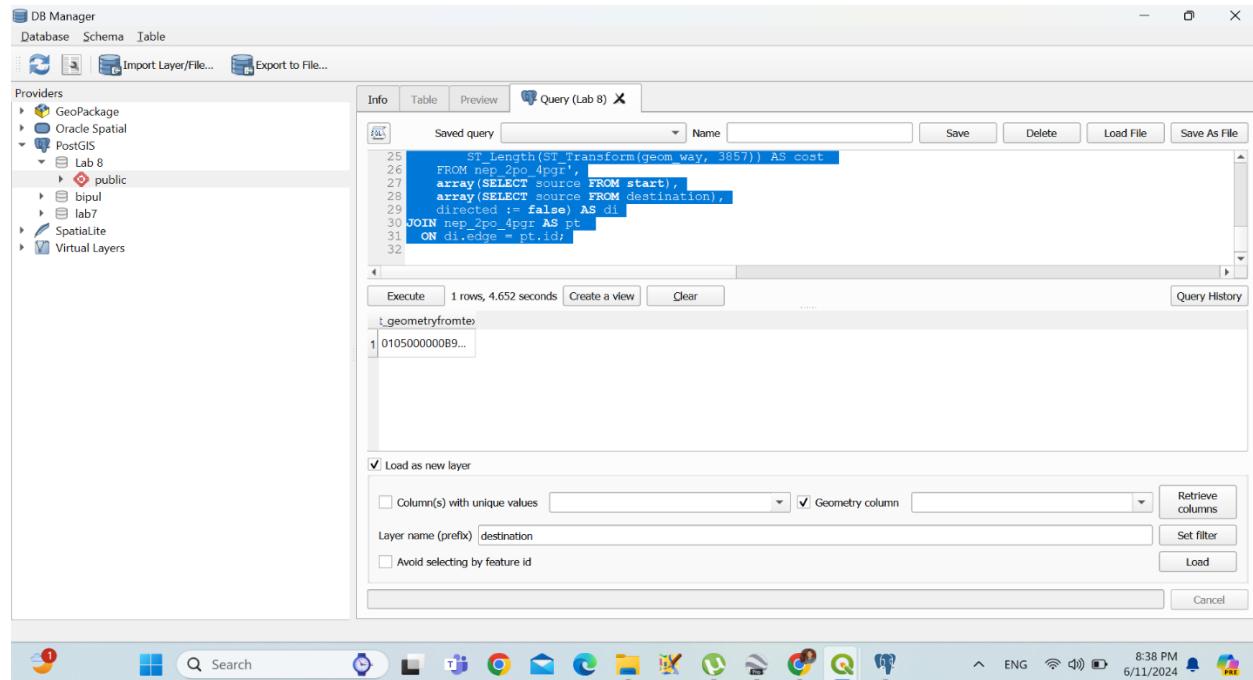
For Destination:

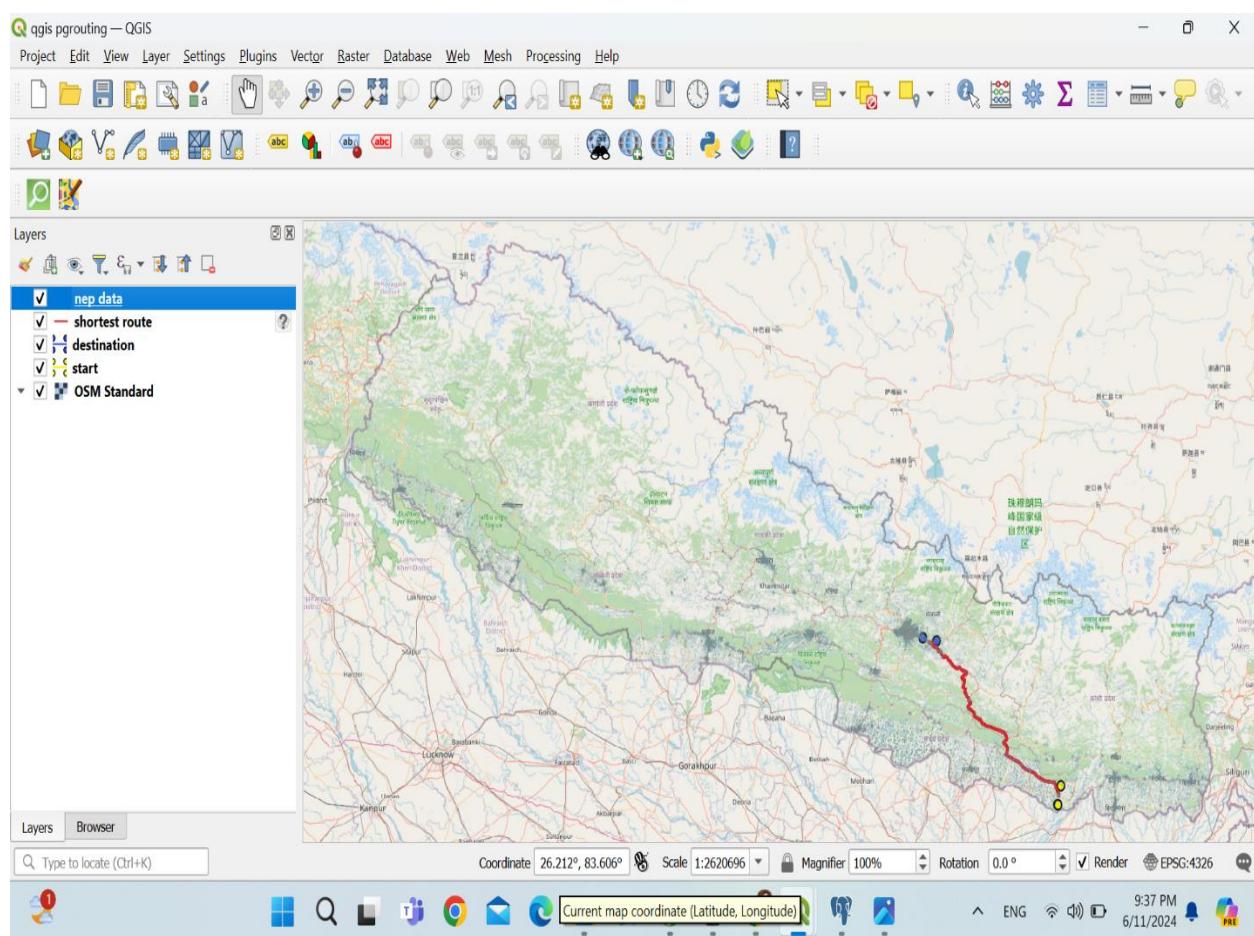
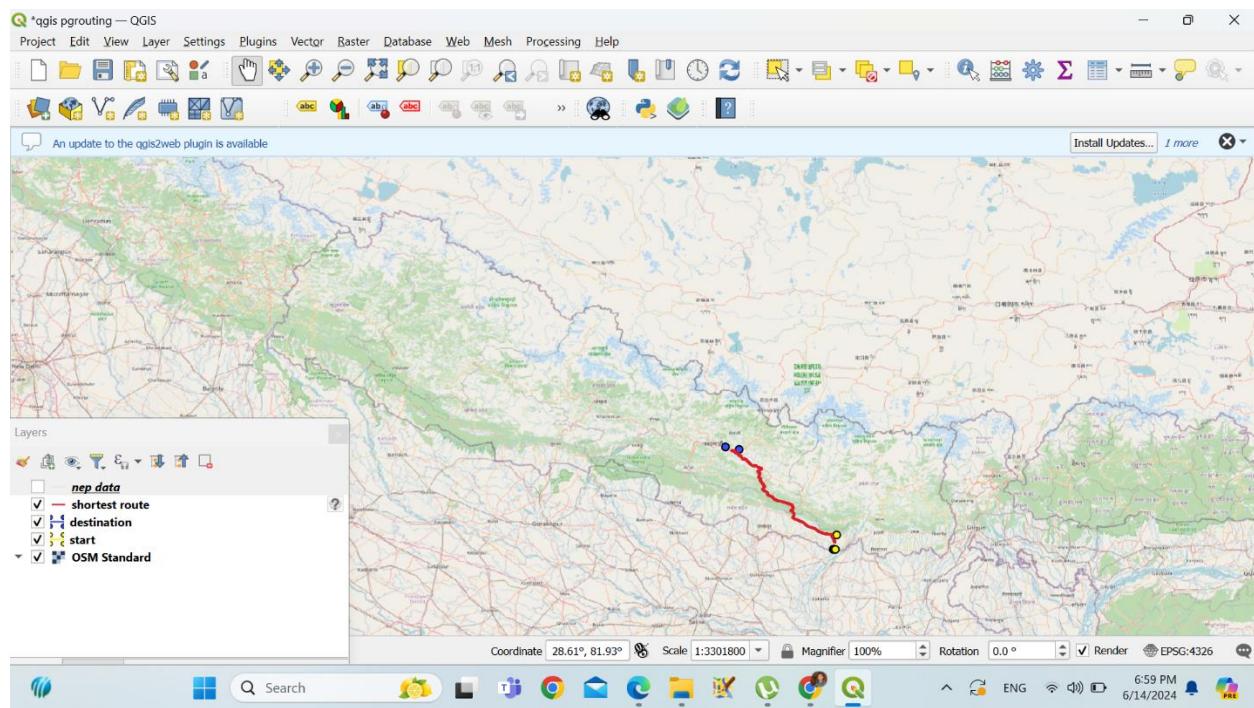
Execute only for the destination > load as layer.



For Shortest route:

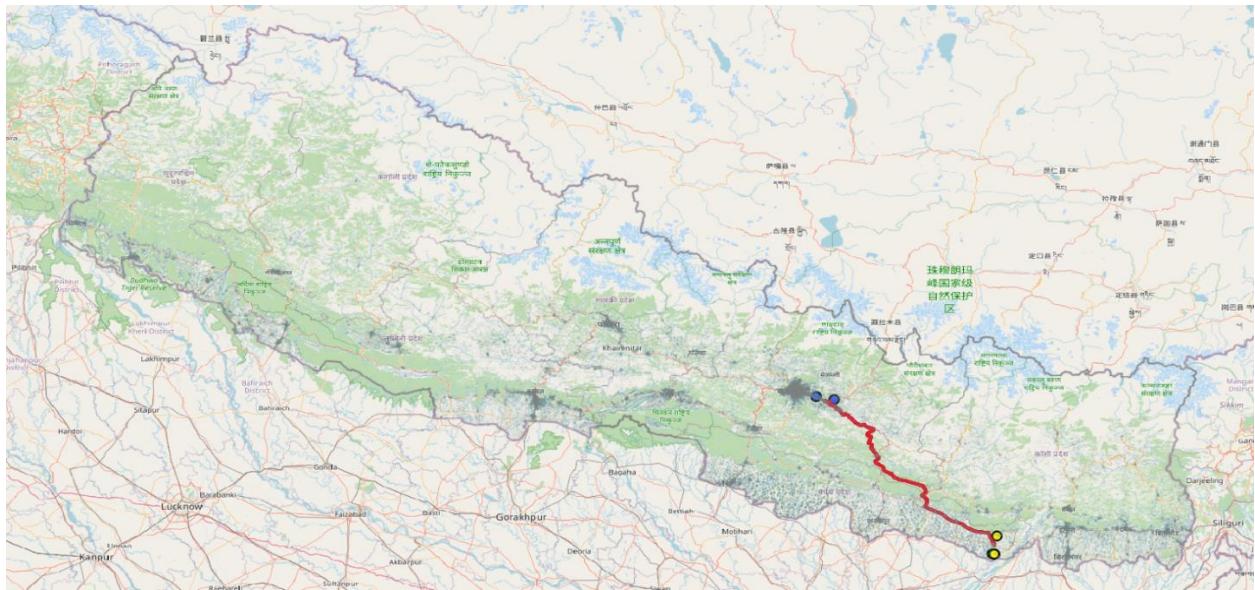
Execute whole query > load as layer.





Shortest route determination using pgRouting:

Export the map as image.



4 CONCLUSION:

In this way the shortest route between the two points was established using pgRouting. The spatial queries were a very crucial part of the whole process. The accuracy of which must be the perfect to achieve the path.

5 References

- Muñoz, B. (2020). Introducing Direct SQL Connection for QGIS, Databricks & more.
- Puyam S. Singh, R. B. (2015). *Applied Geomatics*. Springer.
- Ramesh Janipella, R. V. (n.d.). Spatial Data. 2019.
- Regina O. Obe, L. S. (2017). *Pgrouting A practical guide*.
- Smiraglia, R. P. (2001). *Network Analysis*.