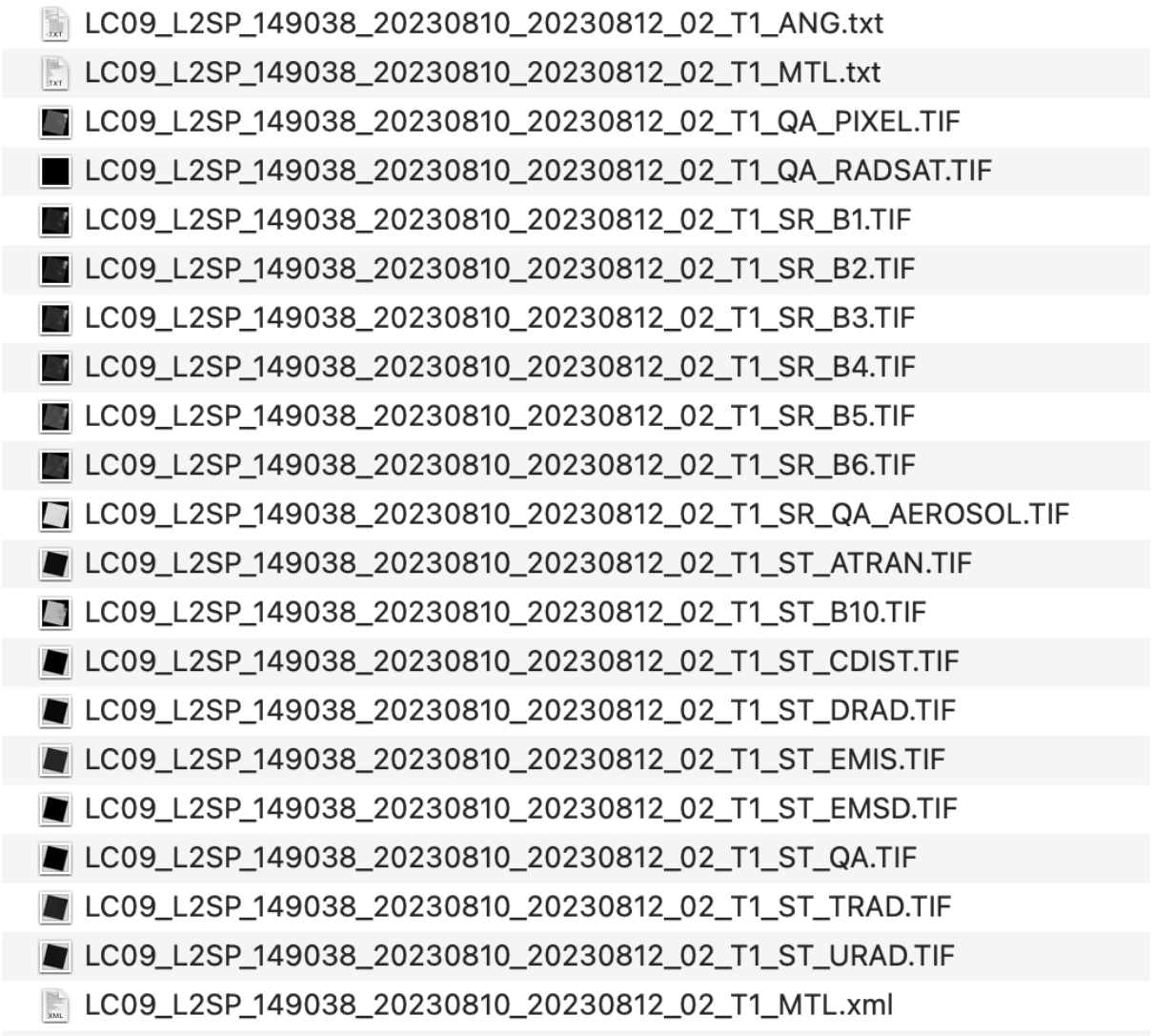
Exercise # 7

In this exercise we will be using additional raster processing functions. These functions although available in other software, can come handy is you can use it at one place. We plan to run these on Landsat 9 data. To download Landsat 8 data for this exercise, you may follow this link https://earthexplorer.usgs.gov (remember you will be needing and earth data login) or you can check your shared course directory.

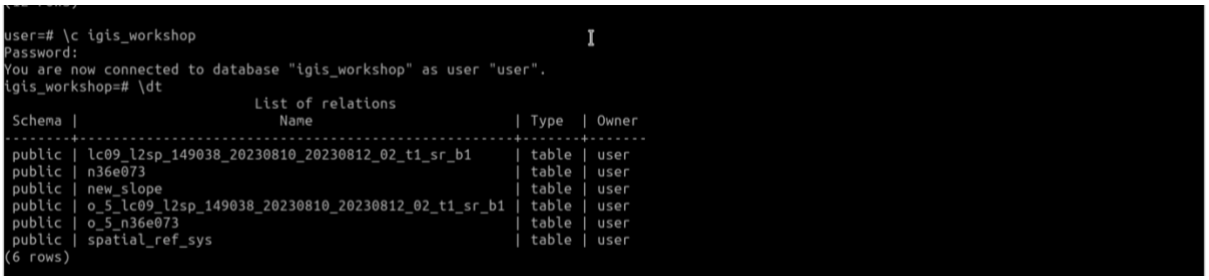
The data contains the following, contents, the contents of these files, are varying. The text files contain file information and metadata. These files are generally well known to many GIS users. We will be importing the rasters files here into our database, in bulk.



To do this you can use the following commands. This will import all raster files into the database you are working on. In case you feel like you need to create separate raster database, please feel free to do so. This will ease out the process of data handling for you. You already have learnt how to make a new database.

raster2pgsql -d -I -C -M -F -t 100x100 -s 4326
LC09_L2SP_149038_20230810_20230812_02_T1_SR_B*.TIF | psql -U user -d exercise_db

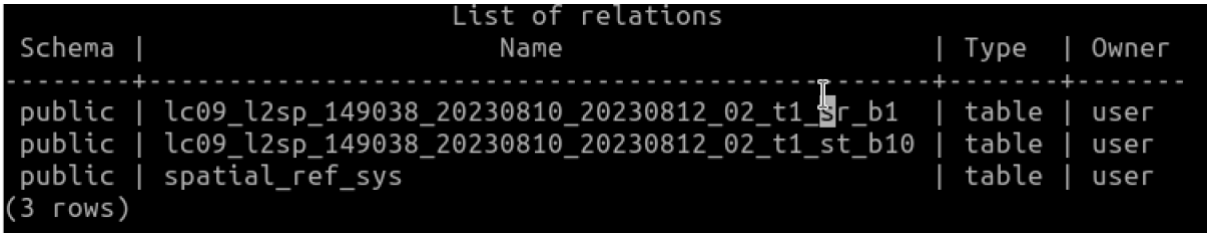
You must have noticed that the above commands take a while, since the data is heavier than the usual single, file we have been importing.



If you check your table however you will find that not all files have been correctly imported. In fact, all that you can find is one single file and its overview file. To fix this you can use two approaches. One is of creating a loop, the other is of creating a multiband virtual raster using GDAL and then importing the files in your database.

For the first instance, you can type the following command,

for i in LC09*B*.TIF; do raster2pgsql -d -I -C -M -F -t 100x100 -s 4326 $i `echo $i | cut -d . -f1` | psql -U user -d exercise_db; done



We just imported the band 1 and band 10 from our data. This was done, to reduce time taken. However, if you want you can do it alternatively with more data files included.

To understand what happened above, type the following command in your terminal while in the Landsat directory.

ls LC09*B*.TIF

Alternatively, you can switch to GDAL,

gdalbuildvrt -separate landsat9bands.vrt LC09*B*.TIF

gdalinfo landsat9bands.vrt

You will get the following output on your screen.

```
prodig@prodig-standardpcq35ich92009:~/Documents/Landsat9Image$ gdalbuildvrt -separate landsat9bands.vrt LC09*B*.TIF
0...10...20...30...40...50...60...70...80...90...100 - done.
prodig@prodig-standardpcq35ich92009:~/Documents/Landsat9Image$ gdalinfo landsat9bands.vrt
Driver: VRT/Virtual Raster
Files: landsat9bands.vrt
       LC09_L2SP_149038_20230810_20230812_02_T1_SR_B1.TIF
       LC09_L2SP_149038_20230810_20230812_02_T1_SR_B2.TIF
       LC09_L2SP_149038_20230810_20230812_02_T1_SR_B3.TIF
       LC09_L2SP_149038_20230810_20230812_02_T1_SR_B4.TIF
       LC09_L2SP_149038_20230810_20230812_02_T1_SR_B5.TIF
       LC09_L2SP_149038_20230810_20230812_02_T1_SR_B6.TIF
       LC09_L2SP_149038_20230810_20230812_02_T1_ST_B10.TIF
Size is 7711, 7841
Coordinate System is:
```

raster2pgsql -d -I -C -M -F -t 100x100 -s 4326 landsat9bands.vrt > LC09_bands.sql
psql -d exercise_db -U user -f LC09_bands.sql

Since the sql file, is massive, about 1.2G hence this might take a while to finish.

```
ANALYZE
psql:LC09_bands.sql:4228: NOTICE:  Adding SRID constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding scale-X constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding scale-Y constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding blocksize-X constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding blocksize-Y constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding alignment constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding number of bands constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding pixel type constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding nodata value constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding out-of-database constraint
psql:LC09_bands.sql:4228: NOTICE:  Adding maximum extent constraint
 addrasterconstraints
----------------------
```

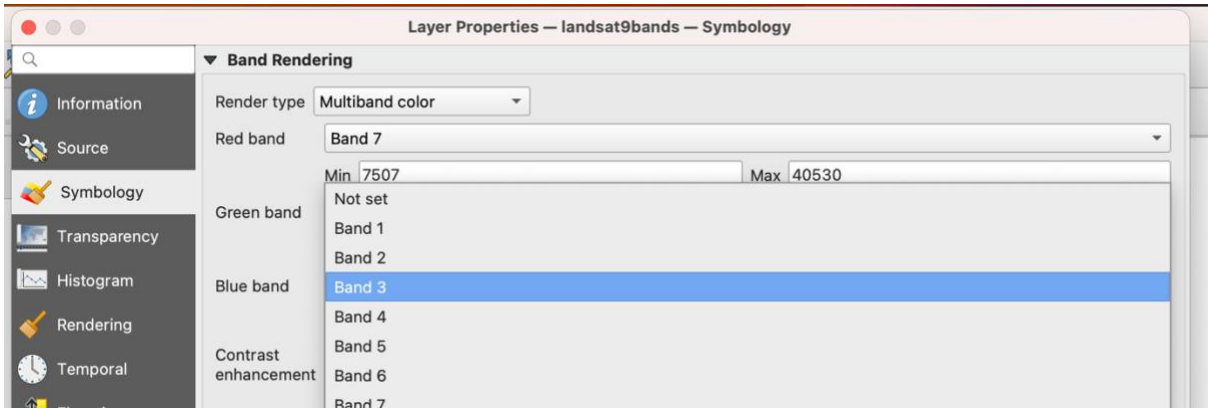
```
VACUUM
prodig@prodig-standardpcq35ich92009:~/Documents/Landsat9Image$ psql -U user -W
Password:
psql (14.4 (Ubuntu 14.4-0ubuntu0.22.04.1))
Type "help" for help.

user=# \c igis_workshop
Password:
You are now connected to database "igis_workshop" as user "user".
igis_workshop=# \dt
            List of relations
 Schema |      Name       | Type  | Owner
--------+-----------------+-------+-------
 public | landsat9bands   | table | user
 public | spatial_ref_sys | table | user
(2 rows)

igis_workshop=#
```

You may want to open this file in QGIS and see how the bands are organized. You can click on the properties and change which one of the bands are displayed in your display window

as RGB combination.



Now we are going to run some raster algebra expressions on our files and create NDVI images from our data.

For that we need to combine band 5 and 4 using the following equation

Landsat 8-9, NDVI = (Band 5 – Band 4) / (Band 5 + Band 4)

Also, you could import separate bands as two tables, and then create a NDVI,

raster2pgsql -C -I -M -F -Y -l 5 -s 4326 -t 100x100 LC09_L2SP_149038_20230810_20230812_02_T1_SR_B4.TIF B4 | psql -U user -W -d exercise_db

raster2pgsql -C -I -M -F -Y -l 5 -s 4326 -t 100x100 LC09_L2SP_149038_20230810_20230812_02_T1_SR_B5.TIF B5 | psql -U user -W -d exercise_db

CREATE TABLE public.ndvi AS SELECT ST_MapAlgebraExpr(arast, 1, brast, 1, '([rast1] - [rast2]) / ([rast1] + [rast2])::float', '32BF') AS rast FROM (SELECT a.rast as arast, b.rast as brast FROM B4 a INNER JOIN B5 b ON a.rid = b.rid) as joined;

Remember that this is a very time taking process, and you might better off with clipped images.