



Point Cloud

Concepts, tools and technologies

Vincent Picavet, 2016

Plan

- 1 - Introduction
- 2 – Working Environment
- 3 - LIDAR : Light Detection And Ranging
- 4 - LAS, LAZ and LAStools/libLAS (WS step 1)
- 5 - PostgreSQL, PostGIS and PDAL (WS step 2)
- 6 - pgpointcloud and pgAdmin (WS step 3)
- 7 - QGIS (WS step 4)
- 8 - Conclusion

Introduction

Oslandia

Open Source GIS Expertise

Training

Development

Consulting

Support

Technologies

QGIS, PostGIS (core committers)

PDAL, pgpointcloud

Business applications C++/Python

Working Environment

```
git clone https://github.com/Oslandia/workshop-pointcloud
```

Environnement

Slides : *supports.ods*

WS : *README.md*

Ubuntu virtual machine (Virtualbox)

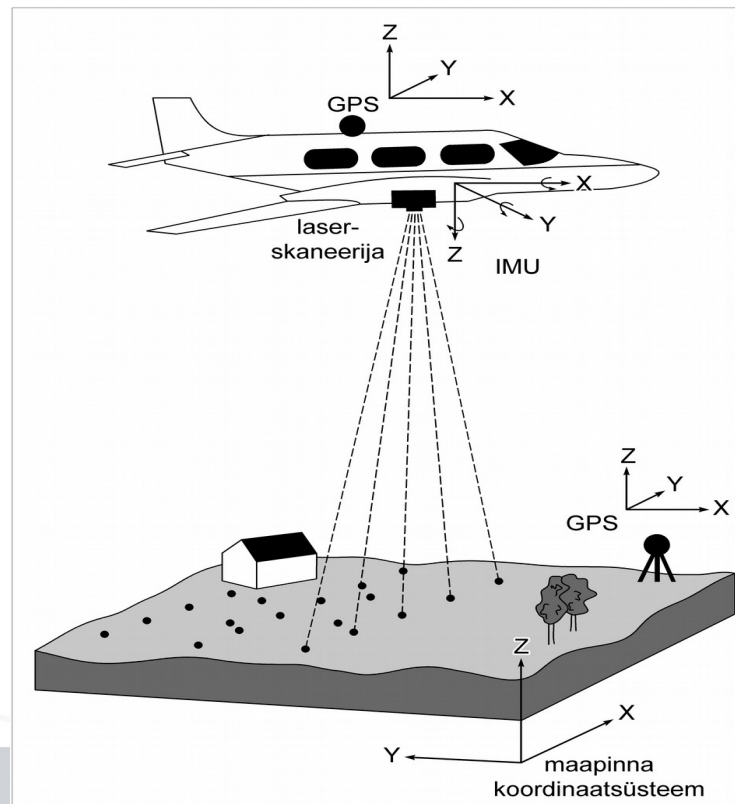


LIDAR : Light Detection And Ranging (1)

Ground or embedded sensor (plane, car, ...)

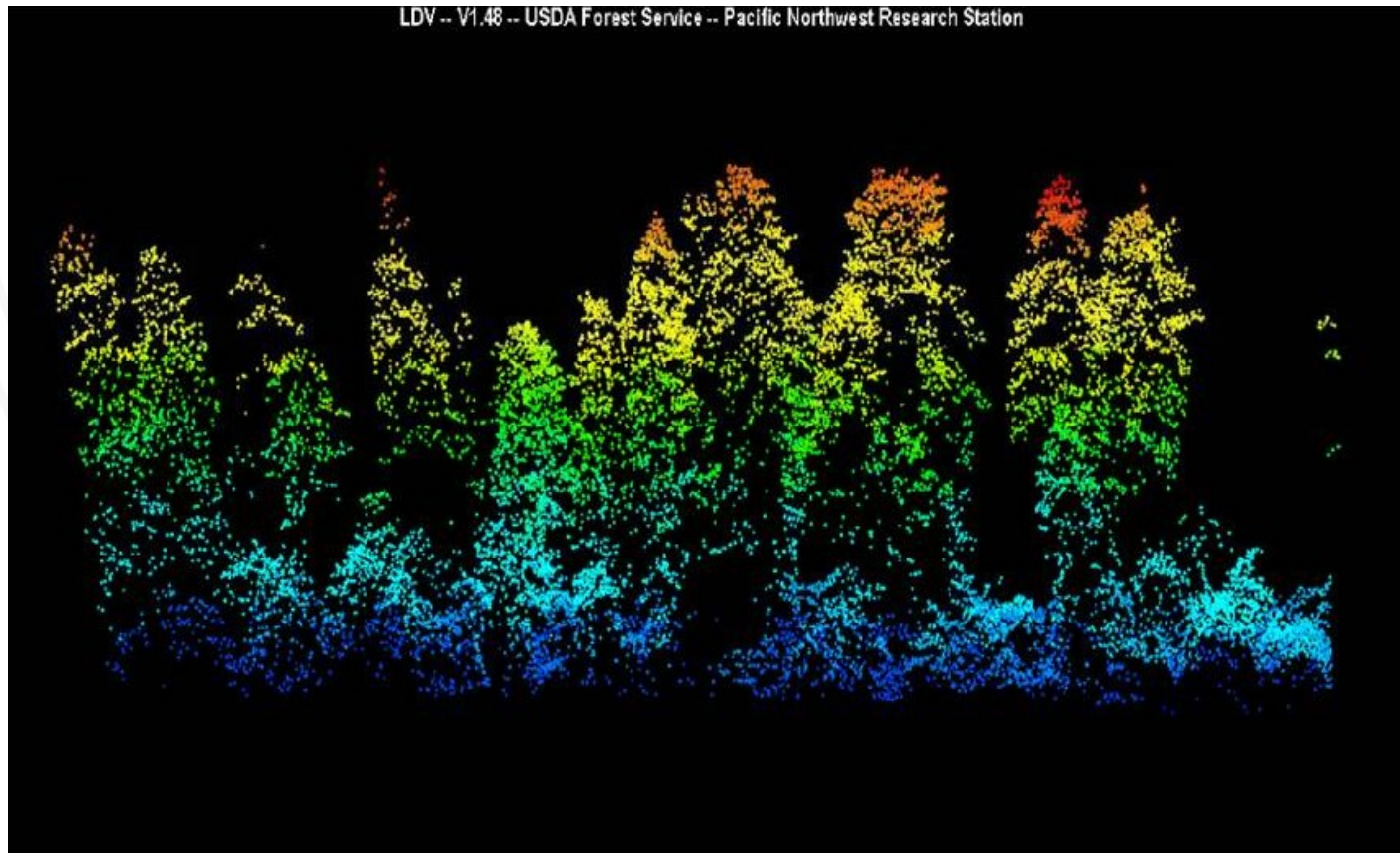
More than 100 000 pulses per second

Reflected pulses (vegetation, buildings, ground, ...) are caught by the scanner



LIDAR : Light Detection And Ranging (2)

Waveform analysis: we keep a “return” for each main intensity peak (threshold)



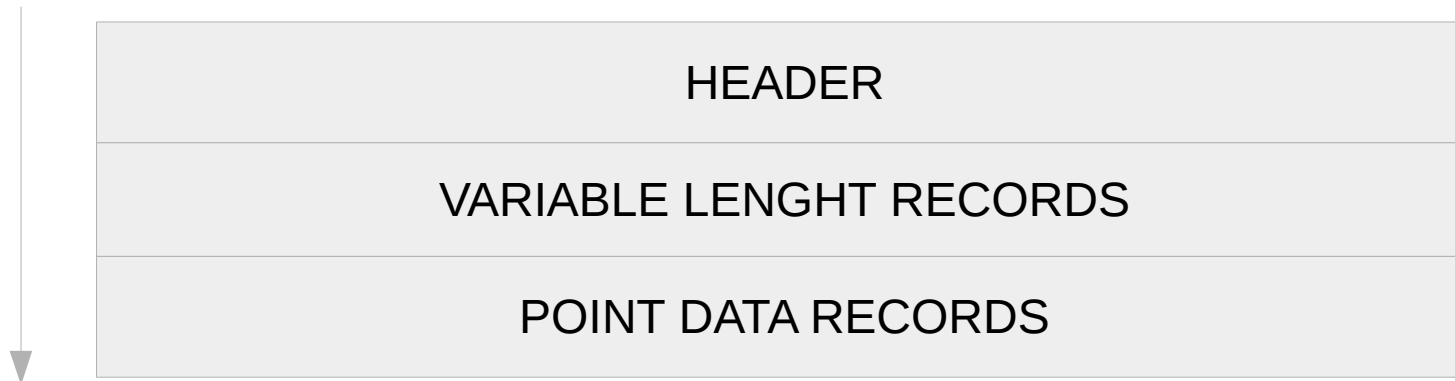
$$z = \frac{t * c}{2}$$



Public file format for the interchange of point cloud data

Specifications

<http://www.asprs.org/committee-general/laser-las-file-format-exchange-activities.html>



LAZ: compressed version of LAS (public format)

<https://www.cs.unc.edu/~isenburg/lastools/download/laszip.pdf>

Tools to work with LAS files

<https://github.com/LAStools>

<https://github.com/libLAS/libLAS>



Which one ?

<http://www.liblas.org/lastools.html>



Each point of the cloud is georeferenced

natural will to store points in a spatial database!

Database

PostgreSQL : ORDBMS, libre, possibility of adding new data type, operators, functions, ...

PostGIS : add support for geographic objects to PostgreSQL



A point cloud

may contains several billions of points

... where each point can be represented by more than 10 dimensions



Store each point one by one in a database is unthinkable!

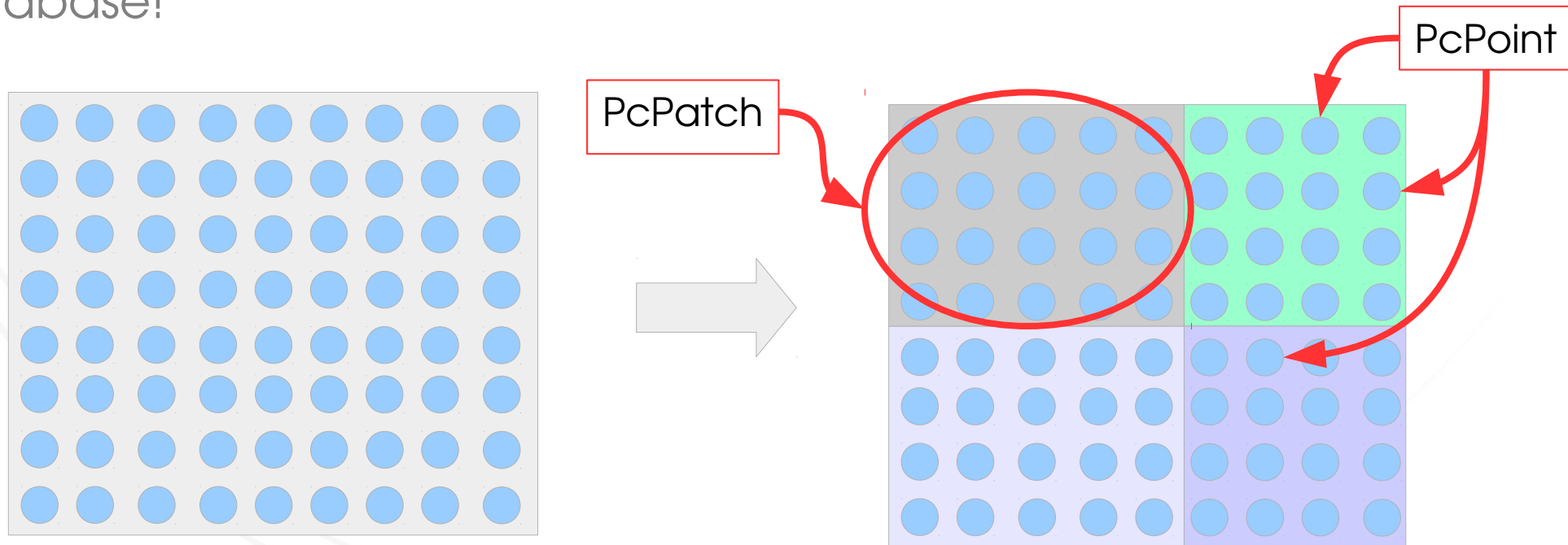
pgpointcloud (2)

pgpointcloud

<https://github.com/pgpointcloud/pointcloud>

PotgreSQL extension for storing point cloud data

Organizes points by patch to reduce the size of the table stored in the database!



pgpointcloud (3)

Schema

Takes care of the variability of points' format

XML Document

Stored within the *pointcloud_formats* table

```
INSERT INTO pointcloud_formats (pcid, srid, schema) VALUES (1, 4326,
'<?xml version="1.0" encoding="UTF-8"?>
<pc:PointCloudSchema xmlns:pc="http://pointcloud.org/schemas/PC/1.1"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <pc:dimension>
    <pc:position>1</pc:position>
    <pc:size>4</pc:size>
    <pc:description>X coordinate as a long integer. You must use the
                        scale and offset information of the header to
                        determine the double value.</pc:description>
    <pc:name>X</pc:name>
    <pc:interpretation>int32_t</pc:interpretation>
    <pc:scale>0.01</pc:scale>
  </pc:dimension>
  <pc:dimension>
    <pc:position>2</pc:position>
    <pc:size>4</pc:size>
    <pc:description>Y coordinate as a long integer. You must use the
                        scale and offset information of the header to
                        determine the double value.</pc:description>
    <pc:name>Y</pc:name>
    <pc:interpretation>int32_t</pc:interpretation>
    <pc:scale>0.01</pc:scale>
```

Patch compression

None, dimensional, GHT or LAZ

Defined in the XML schema

```
<pc:metadata>  
  <Metadata name="compression">dimensional</Metadata>  
</pc:metadata>
```

Dimensional compression

Well-suited for small patches (low variability)

Each dimension of a PcPatch uses it's own dimensional compression algorithm (RLE, ZLIB, SIGBITS)

```
int  
pc_dimstats_update(PCDIMSTATS *pds, const PCPATCH_DIMENSIONAL *pdl)
```



Point Data Abstraction Library

Command line tools

Allows to work with point cloud (reading, filtering, writing,...)

Pipeline

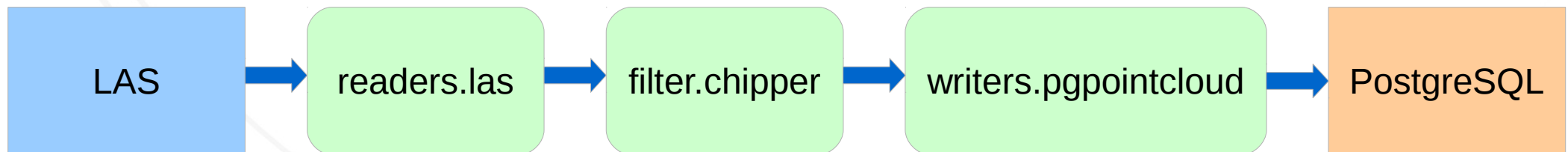
Sequence of operations for building a processing chain

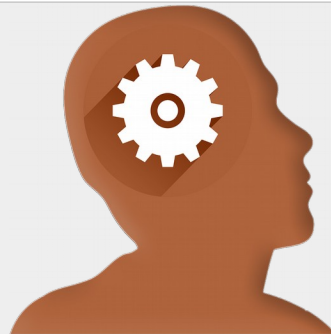
JSON format since v1.2 (XML for earlier version)



Example :

```
{
  "pipeline": [
    "/home/vpi/data/auvergne/lidarverne/opendata.craig.fr/opendata/lidar/agglos/2013_clermont-ferrand/clermont.las",
    {
      "type": "filters.chipper",
      "capacity": "1000"
    },
    {
      "type": "writers.pgpointcloud",
      "connection": "dbname='foss4g' user='postgres' port='5433'",
      "table": "lidar",
      "compression": "dimensional",
      "srid": "2154"
    }
  ]
}
```

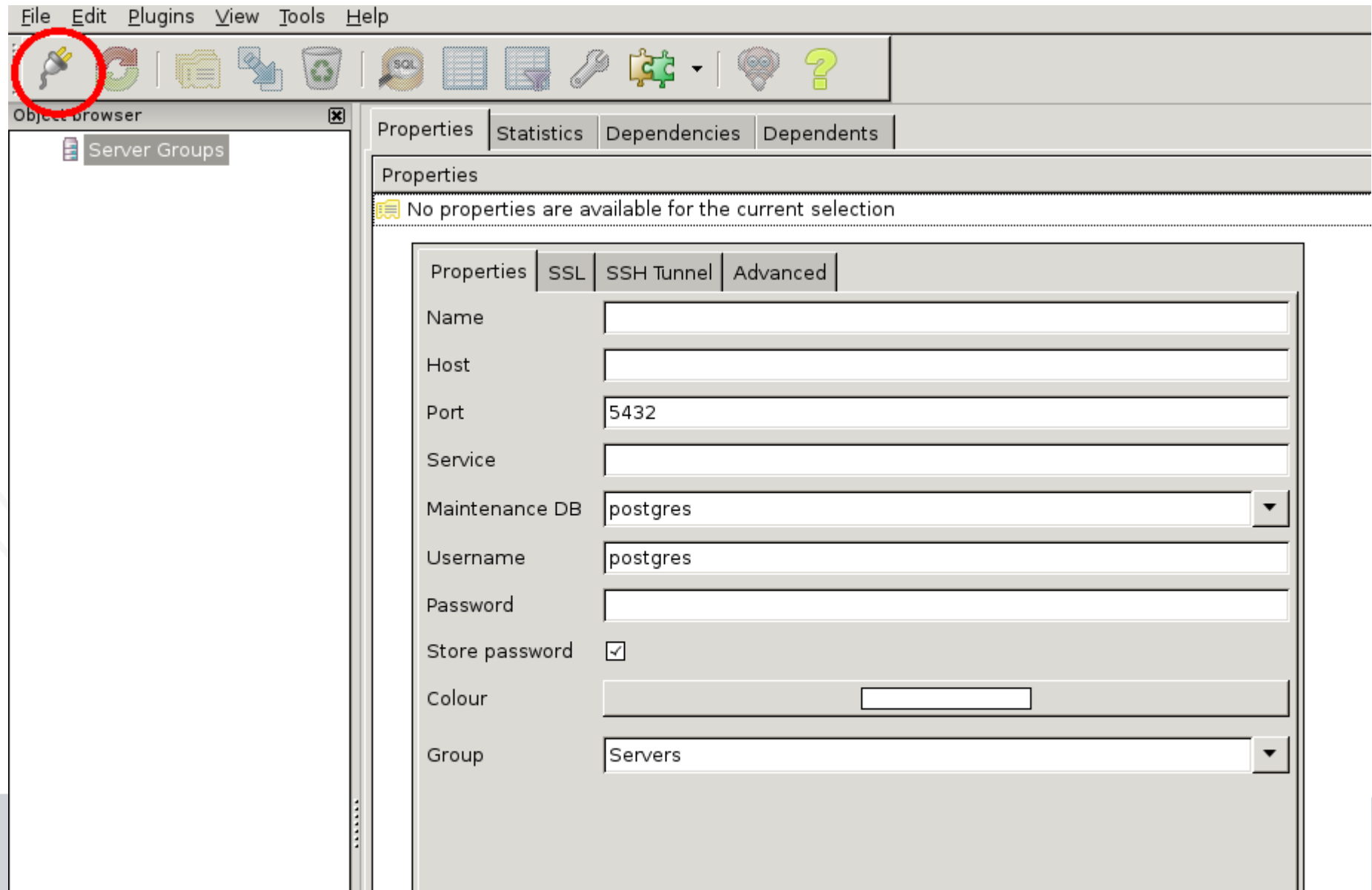




STEP 2

pgAdmin (1)

GUI to work with PostgreSQL



pgAdmin (2)



GIS Software :

Libre

Cross-platform

C++, Python plugins



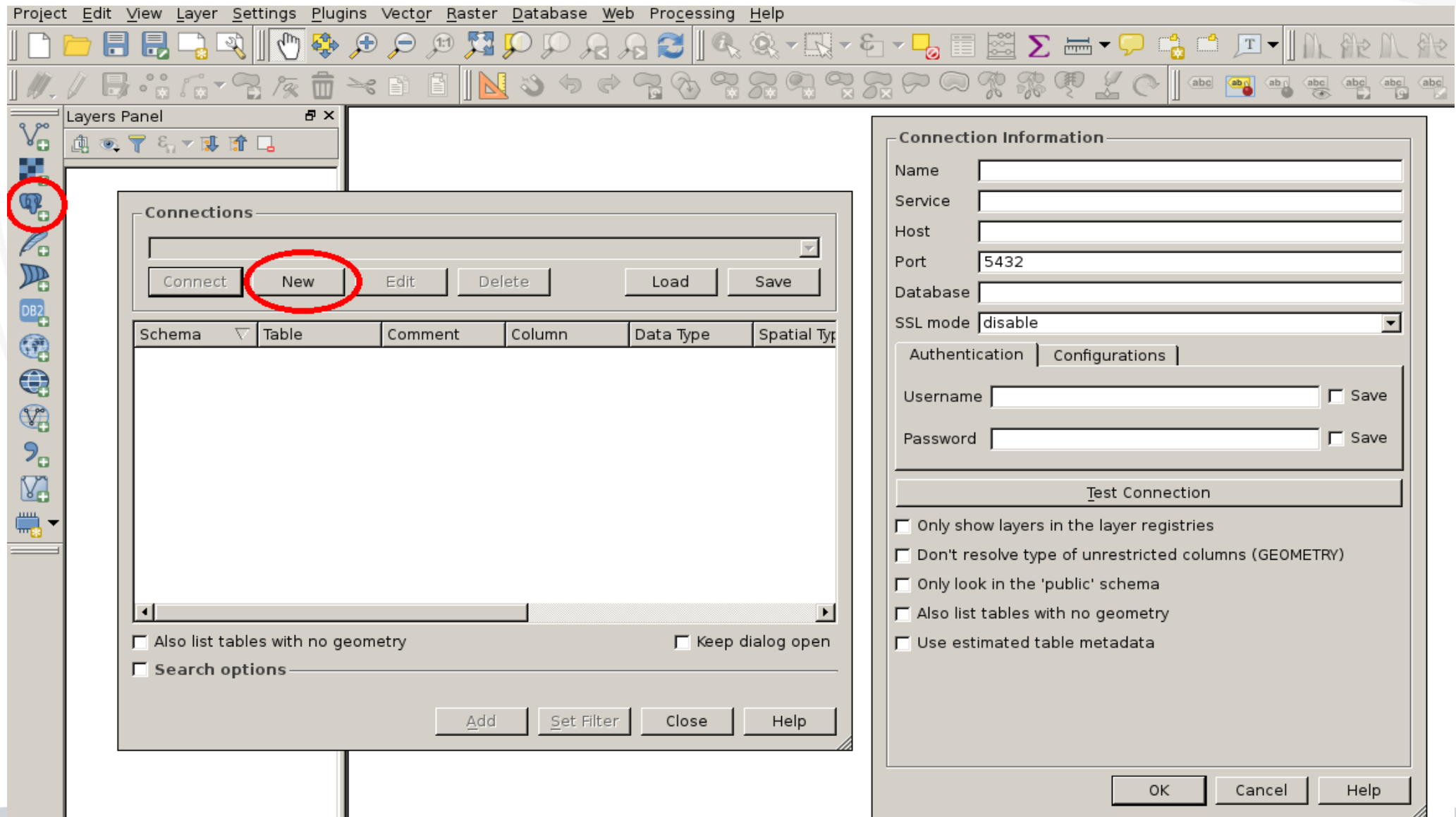
QGIS with point cloud??

Connexion with spatial databases like PostGIS

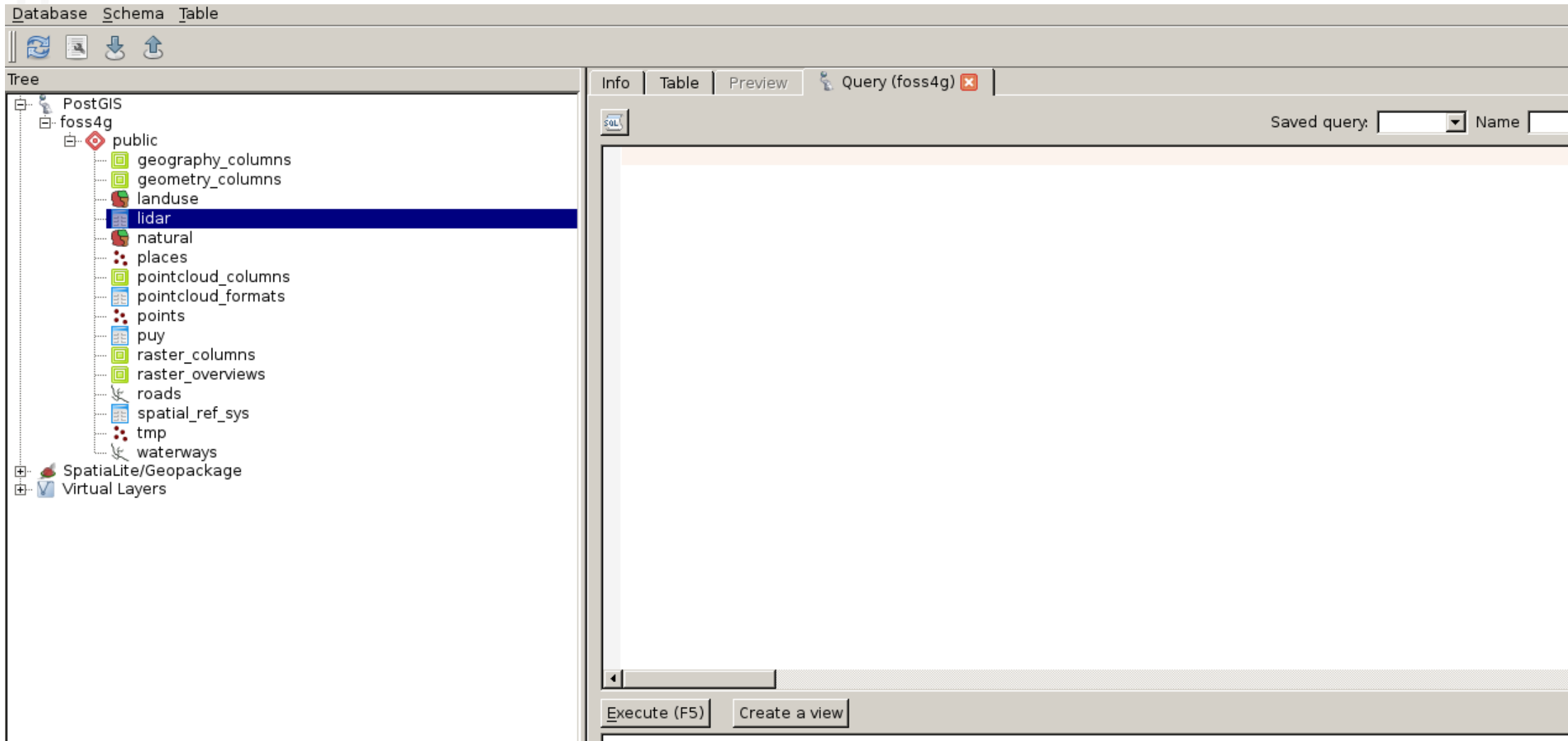
Knows what a PcPatch is!

QGIS (2)

Add PostGIS layer

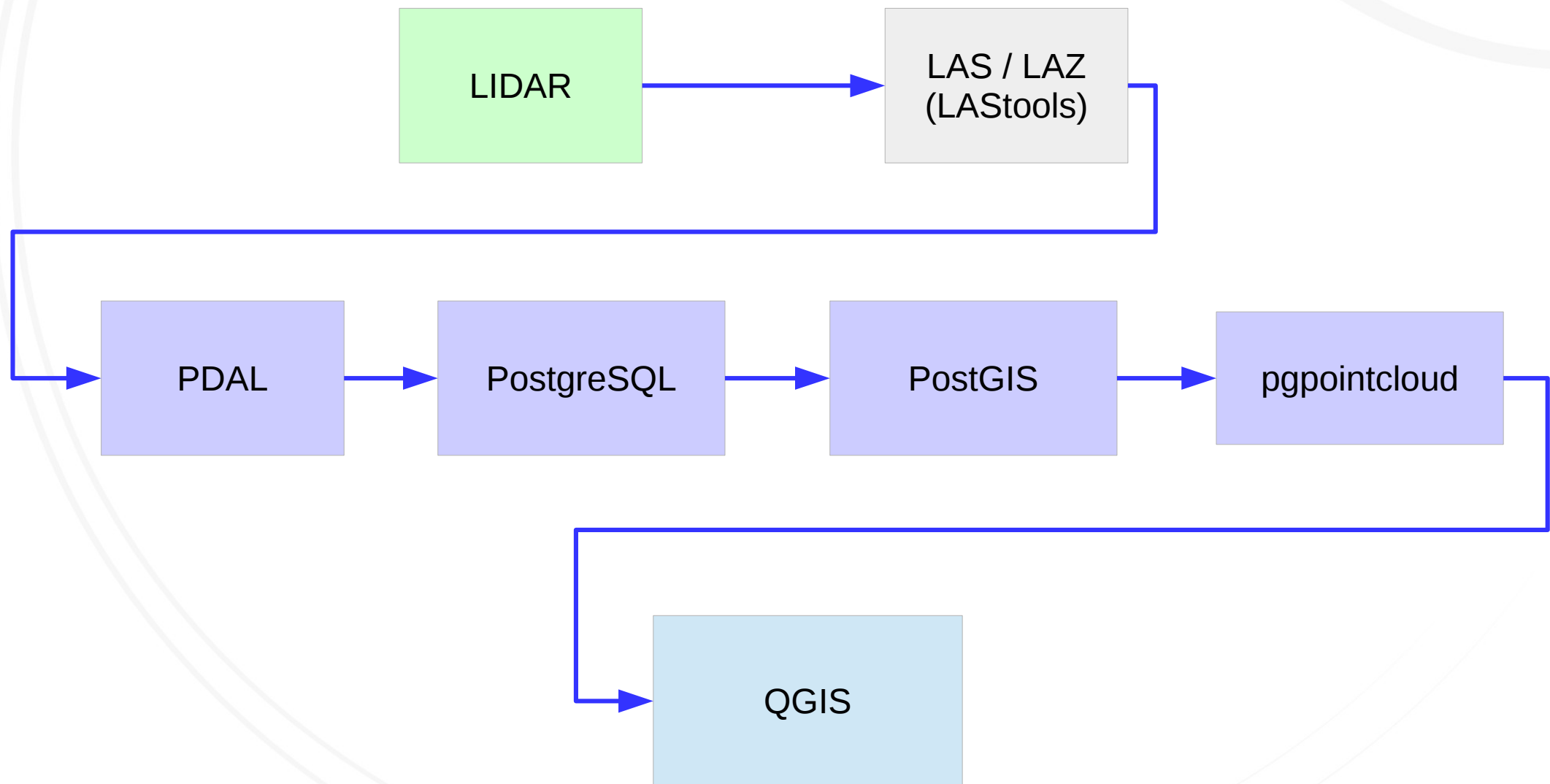


Database Manager :





Conclusion



Questions

