# Guide for Establishing an Archaeological Database with Open Software

Jonathan S. Lim, Carla Klehm, Kristin Landau, and Malcolm Williamson

*Center for Advanced Spatial Technologies, University of Arkansas*


Foundations of the Urban Cholula Salvage Archaeology Geodatabase (UCSAG) National Endowment for the Humanities Award PW-285206-22

**Version 2.0, March 2024**

## Acknowledgments

The authors would like to thank Josue Gomez, Adriana Saenz, Carlos Cedillo Ortega, and their team at Instituto Nacional de Antropología e Historia, for inviting us to Puebla, Mexico to present this guide. Thanks also to John Wilson and Hayley Hames (CAST) for their technical assistance.

The version of this document is V2.0, updated 3/21/2024.

# Table of Contents

# 1 Introduction

## 1.1 Scope

This guide is a step-by-step workflow for establishing an archaeological database using only open software. It also contains instructions for converting existing datasets, both within and without ArcGIS Desktop, into a form that is readily uploaded to the database. The reasons behind creating this guide are twofold. First, we find that many archaeologists begin with data collection, record observations in various fields, further organized into tables, rather than beginning data collection and management within a geodatabase. Unless the archaeologist has training and access to GIS, programming, and/or access to relatively easy-to-use geospatial software (e.g., ESRI products), data often remains in more simple digital, digitized forms. These forms are often isolated, i.e., they don't link or interact with one another, nor are they integrated with other forms of digital data (e.g., georeferenced maps, imagery, point clouds, GNSS data). It can be harder for that data to be indexed, located, reused, compared, and/or incorporated by other researchers, creating a gap between data use and data potential. Second, opportunities for training in spatial data methods in archaeology, let alone spatial data management, are limited (see Klehm 2023; https://doi.org/10.1017/aap.2022.38). As with spatial methods in archaeology and the digital humanities more broadly, there are a series of technical and logistical "bottlenecks" that humanists studying the past find themselves facing: entry-level geospatial courses exist in university coursework, but finding content tailored to archaeological and heritage problems is more difficult to come by; training is found either at the entry or expert level, with little resources or mentorship to bridge that gap; few opportunities exist to explore the potential of new equipment and software; and support systems for geospatial software or instruments are often developed for other industries (e.g. agriculture, forestry), making it difficult to find help. There exists a real need for guidance, mentorship, and training materials.

This guide was developed based on conversations with archaeological colleagues in Mexico and their experiences of data curation, digitization, and organization. They are struggling with the backlog of processing decades of rescue (or salvage) archaeology projects that are in various states of digitization. The data have fairly standardized forms (e.g., artifacts by type, human remains, faunal remains) but are not associated with one another, and their spatial relationships are yet to be explored. Further, our colleagues expressed a strong interest in being able to share the CRM data collectively with other researchers, as—like in the United States—there is less frequently published and harder to find, but no less valuable.

Our workflow intentionally uses FOSS (free and open-source software) platform solutions, including PostgreSQL, PostGIS, and QGIS, as part of our commitment to making not just data more widely available and accessible, but research capabilities and training more equitable despite location and limited resources.

This workflow presented in this guide assumes that the PostgreSQL database will initially be installed and operated locally on a Windows-based laptop. If moving it to a server for remote access, work closely with the server administrator to set up account security and permissions. Before reading this manual, it is strongly recommended that the user sketch out a potential database structure, as in the example shown in Figure 1, on the following page.

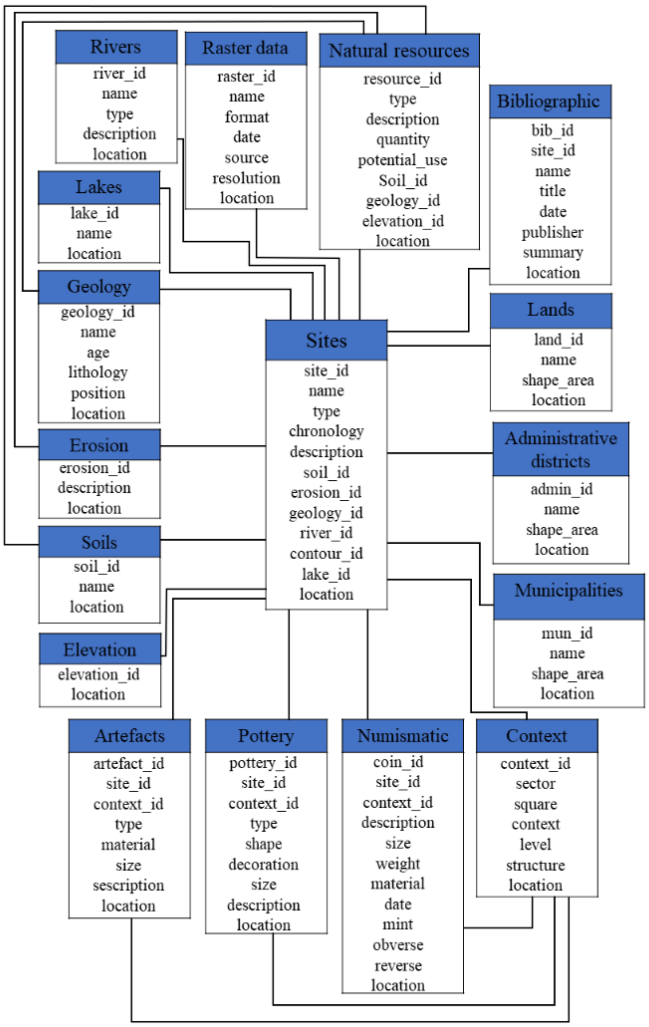


**Figure 1. Before starting, it may be helpful to visually sketch how each existing table relates to one another. From doi:10.57573/be-ja.13.99-114**

## 1.2 Section summaries and software used

Section 2 provides instructions on how to export any existing tables and vector data still residing on **ArcGIS Desktop 10.3**. Vector data needs to be in .shp (shapefile) format, and tables should be in .dbf or .csv format.

Section 3 shows users how to install and configure **PostgresSQL**, an open and powerful database management software package. **PostGIS** will also be used. This is an extension of PostgresSQL that allows the handling of spatial data of all types. We will also install and configure **DBeaver**, which provides a powerful user interface for administering the database— note that it is very similar in function to **PgAdmin**, which comes bundled with PostgresSQL, but is notably more stable and faster.

Section 4 will cover how to move all existing datasets into the database with PostGIS. It will also briefly explain how to create schemas in DBeaver, which serve as a means for organising data. The establishment of different schema may be necessary for a variety of purposes: To limit access to the entire database, to organise data by function, etc.

Section 5 will demonstrate how **QGIS**, a powerful and free geographic information system (GIS) package, can easily connect to the database, in order to display and analyse data the data stored within.

Finally, section 6 will propose two methods for making entries in the database. **LibreOffice Base** is a free alternative to Microsoft Office's **Access**. Forms may be generated to create a well-laid out and customised user interface

for easy data entry. Furthermore, new spatial data products created in QGIS can be saved directly to the database, assuming they are connected.

## 1.3 Addendum: Further considerations when plannning database structure

This section was added following the February collaboration meeting in Cholula, Mexico. In the workshop, we established a basic strategy for organising the Cholula dataset moving forwards. A significant amount of planning and data cleaning is required to allow the tables to be uploaded to the database. Most notably, where a field is anticipated to have multiple, repetitive values, we have decided to create a new table containing all possible values. Schema essentially serve as 'folders' in a database to organize data. A new schema will be established for every project ("Project Schema"). We would also suggest that an additional schema (a "Universal Schema") be created for common tables that are shared between projects. In summary for every field, the following decisions need to be made (Figure 2):

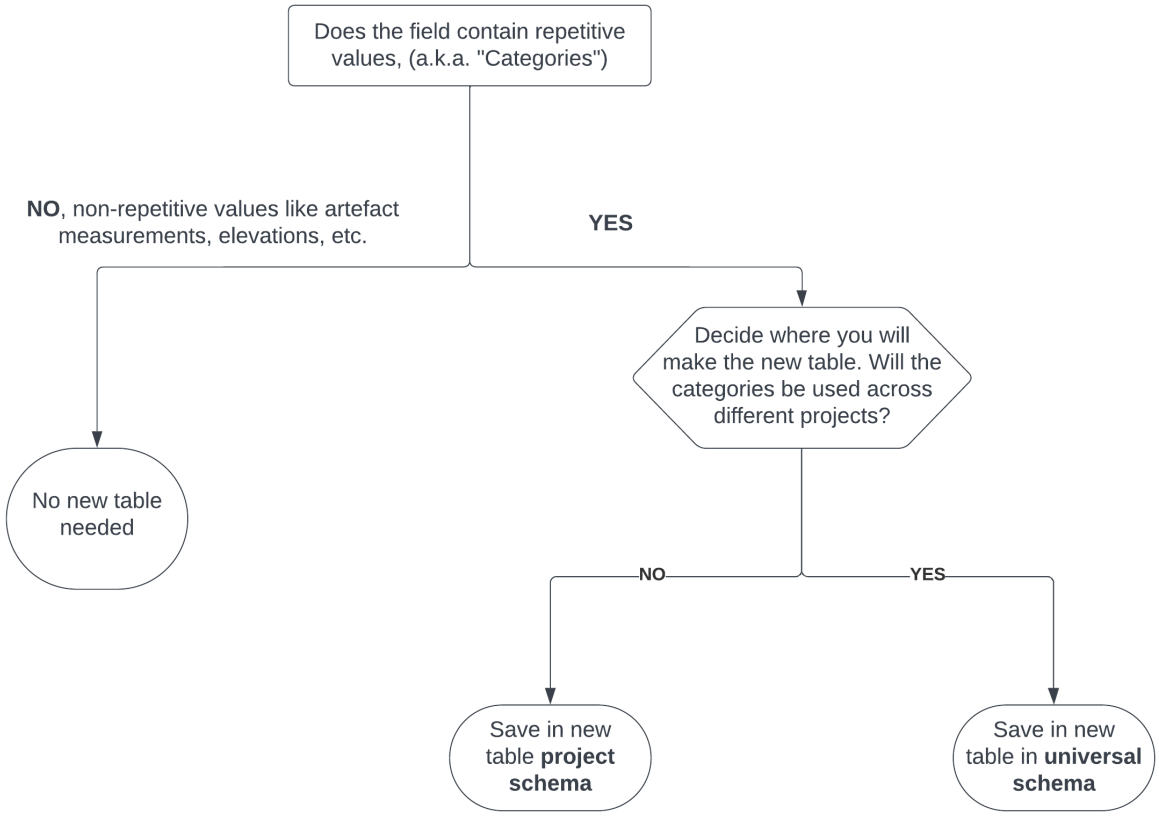


**Figure 2. General considerations when deciding on how to organise a database**

# 2 Preparing existing data on ArcGIS Desktop 10.3 for transfer

## 2.1 Exporting vector data

1. In order to transfer existing spatial data in ArcGIS Desktop 10.3 to the new database, vector data files must be exported as .shp files, and non-spatial tables should be exported as .dbf files.

2. First, lets export the vector data. Open your project in ArcMap. Click on the button indicated in Figure 3 to open the catalog pane, if it is not already open.

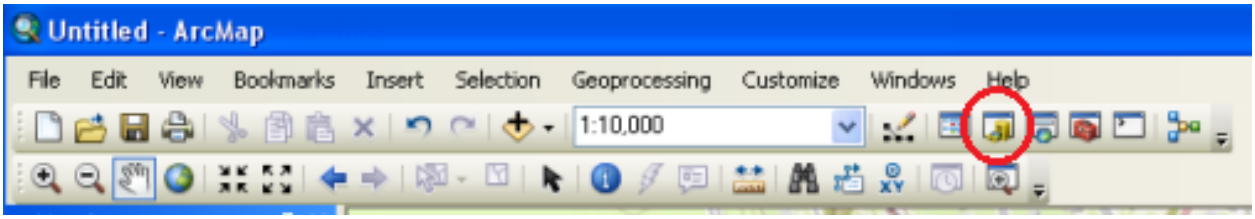


**Figure 3**

3. With the catalog pane open, expand the directories to identify the vector data you intend to transfer (Figure 4). They are likely stored in a geodatabase associated with your project.

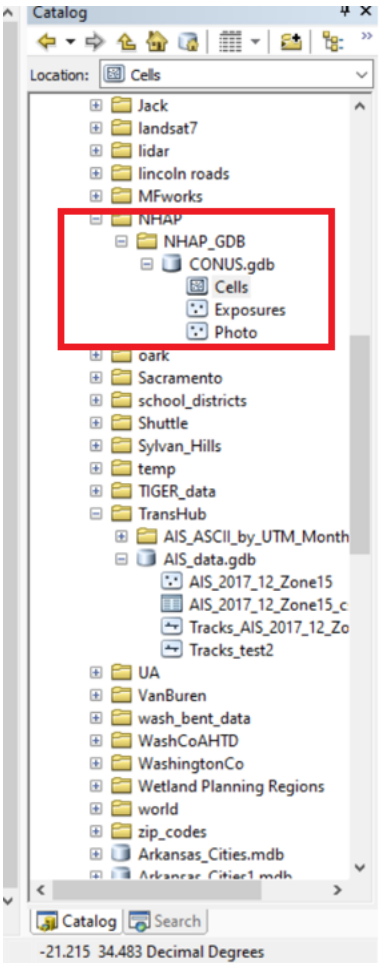


**Figure 4**

4. Right click on a feature class, and click Export to Shapefile (Single) (Figure 5). If transferring multiple feature classes at a time, shift left-click or control left-click to select all the relevant files, then right click and Export to Shapefile (Multiple).

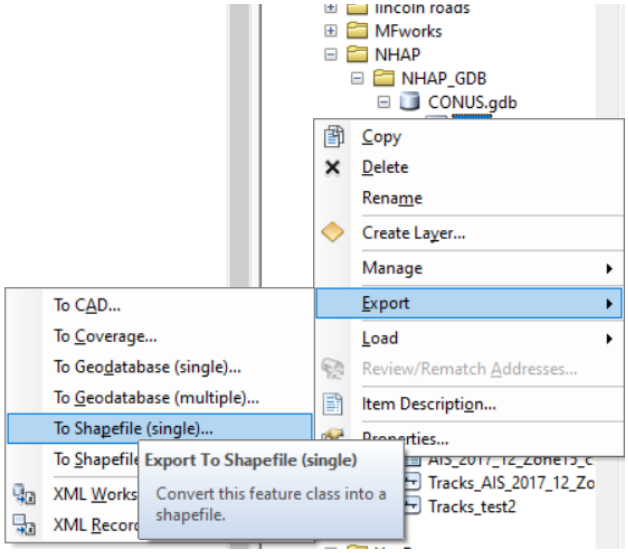


**Figure 5. Note: If dealing with multiple files, Click "To Shapefile (Multiple)" instead.**

5. The Feature Class to Feature Class window will appear. Choose an output location folder on your computer that is not a geodatabase. It is very important to ensure that the file or files are exported as .shp files (Figure 6). Click OK.

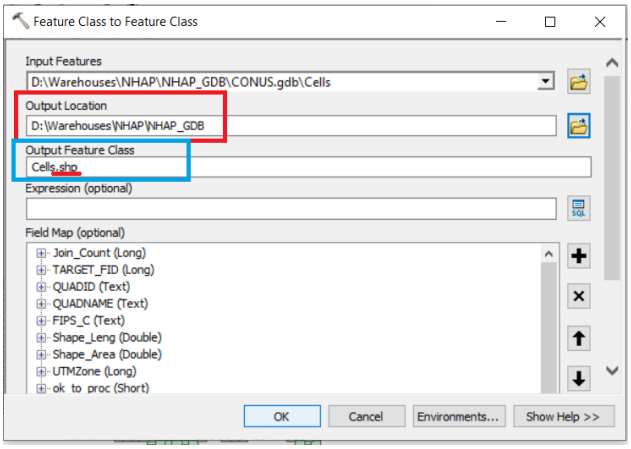


**Figure 6**

## 2.2 Exporting tables

1. Your vector data should now be exported in the correct format (.shp) to the location you specified. We will now do the same with the data in table format stored in ArcGIS Desktop, to export them as .dbf files.

2. Access and open your table using your preferred method. A window showing the table, as shown in Figure 7, should be visible.

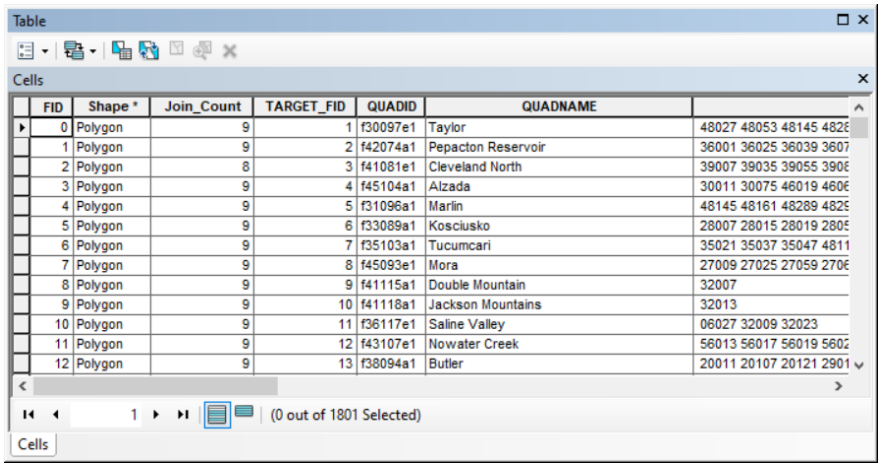


**Figure 7**

3. Click on the first icon in the top left corner, then click Export (Figure 8)

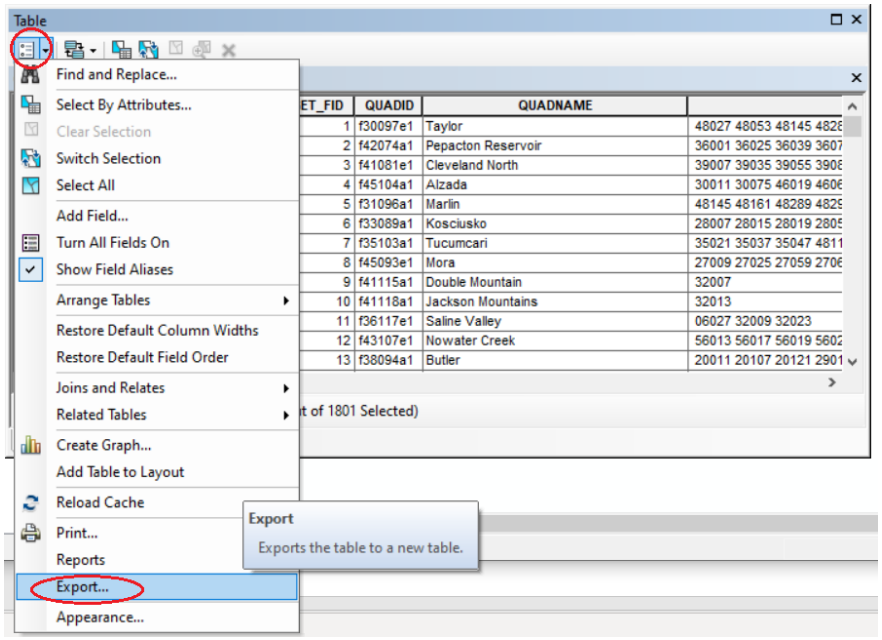


**Figure 8**

4. The Export Data window should appear. Click the folder button next to the Output Table field (Figure 9).

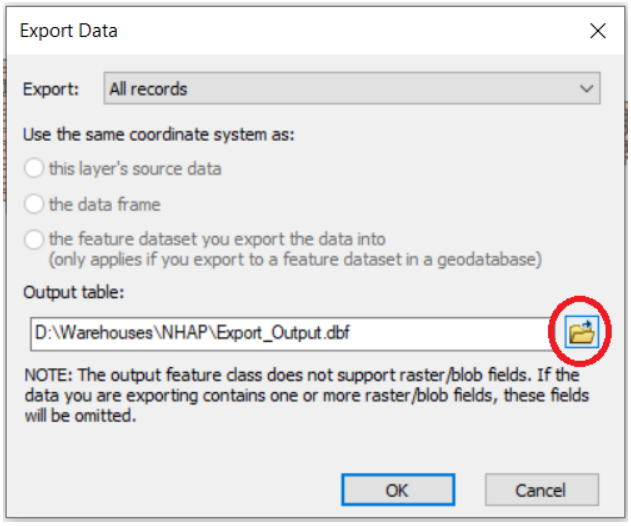


**Figure 9**

5. Press Save as Type drop down list, and select the format "dBase Table". This will save the file as a .dbf file. Select an output location that is not a geodatabase, and click Save to close this window (Figure 10).

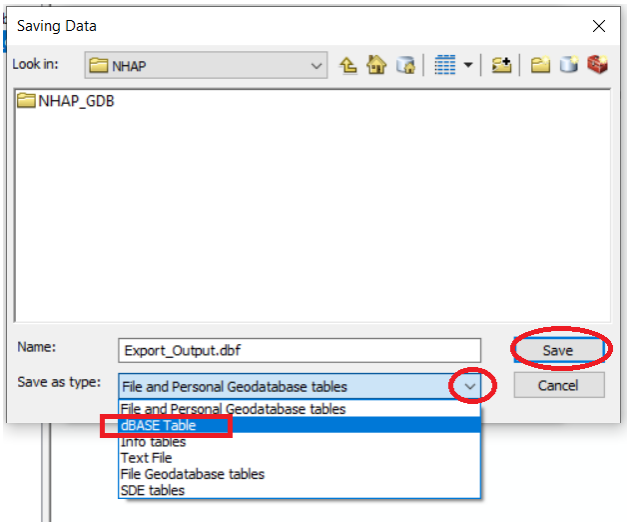


**Figure 10**

6. With the output location set successfully, and the filetype set to .dbf, click OK to export the table (Figure 11)

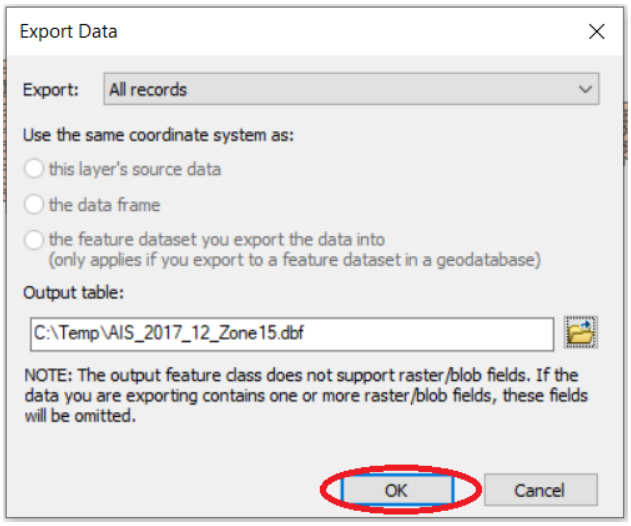


**Figure 11**

# 3 Installing Database and Management Software

## 3.1 Installing PostgreSQL and PostGIS

1. PostgreSQL is available for download from https://www.postgresql.org/download/
2. Follow the links, and download the appropriate version for the device
3. Click on the executable file to run the installer
4. Specify the installation directory (Figure 12). If installing for local use (ie. not on a server for remote use), use the default settings for ease of use. Click Next.

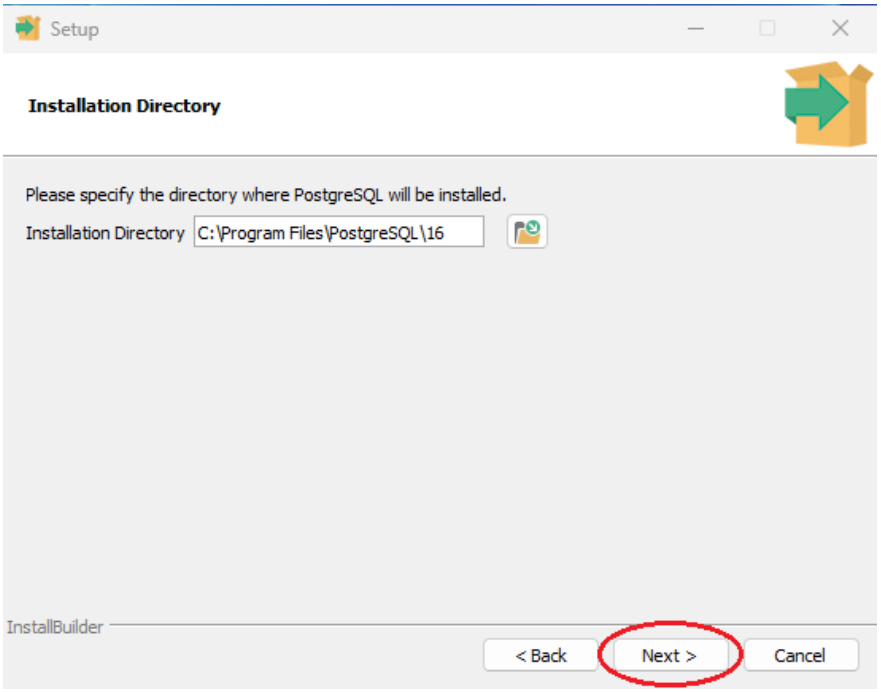


**Figure 12**

5. On the next page— Select Components— ensure all four components are ticked (Figure 13). Click next.

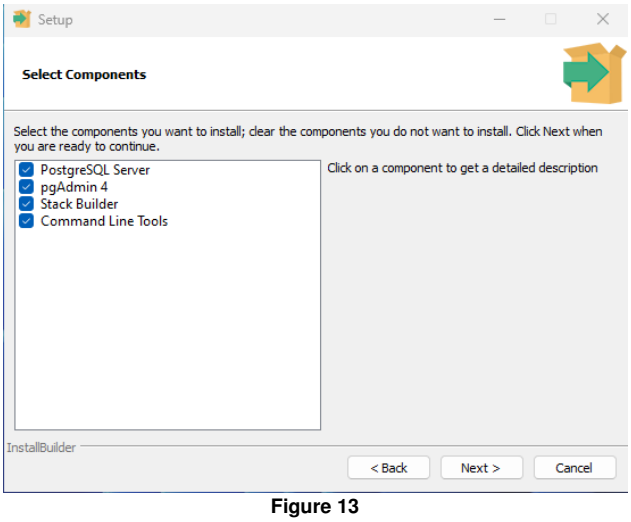


**Figure 13**

6. On the next page— Data Directory— specify location of data folder to be created for the database. If installing on a laptop, leave it as default . Click next.
7. Specify a password for accessing the server. Write this down somewhere safe. Click next.
8. The next page will ask you to select a port number. If installing for local use, leave this as default: 5432. Click Next.
9. On the next page, leave the database cluster specification as Default. Click next.
10. You will be presented with a summary of your selections so far (Figure 14). Double check the details, then click next to install PostgreSQL.

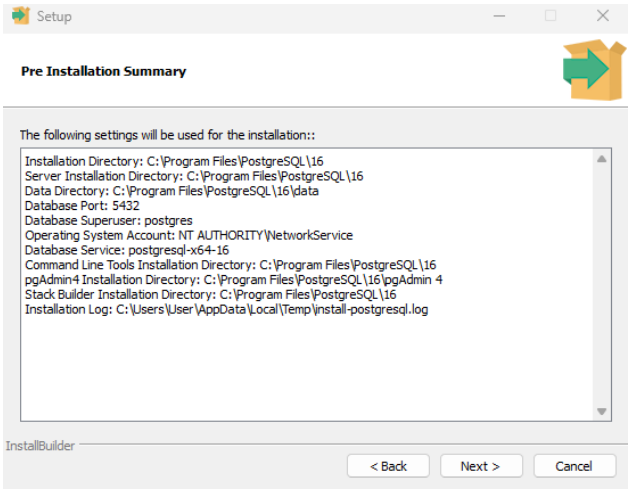


**Figure 14**

11. After the installation is complete, you will be presented with another window. The tick box to run Stack Builder, then click Finish (Figure 15). This is very important, as it will allow you to install extensions for your database as needed; most notably PostGIS, which is used to handle spatial data. Press next.

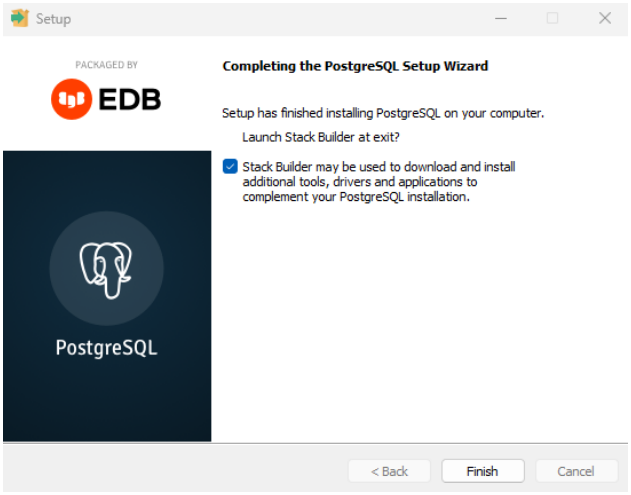


**Figure 15**

12. If using locally, ensure PostgreSQL 16 (x64) on port 5432 is selected (Figure 16). Click next.
13. Expand the Spatial Extensions group and tick PostGIS (Figure 17)

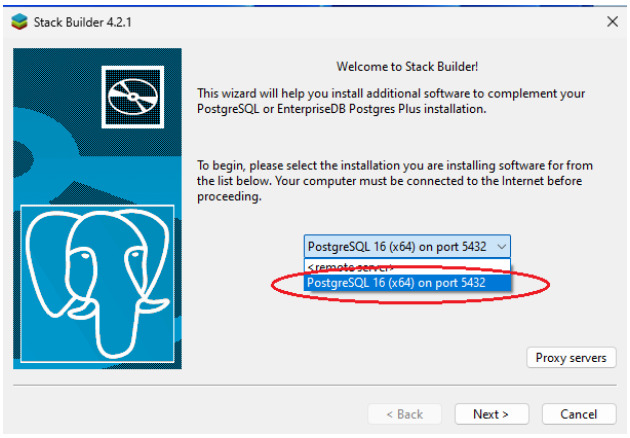


**Figure 16**

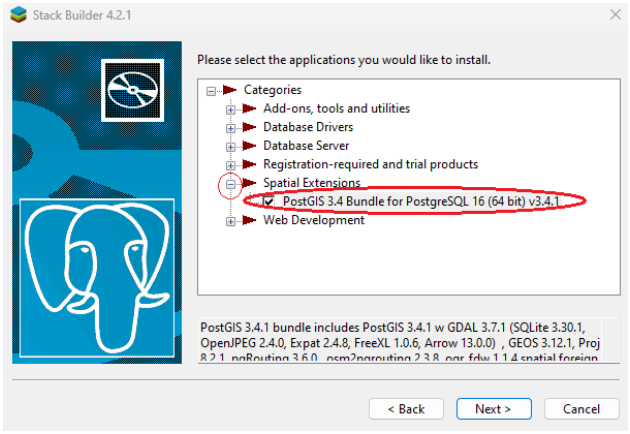


**Figure 17**

14. Expand the Database Drivers group, and tick Npgsql and PsqlODBC 64 bit (Figure 18). This will allow you to use Python commands, if so desired. Click Next.

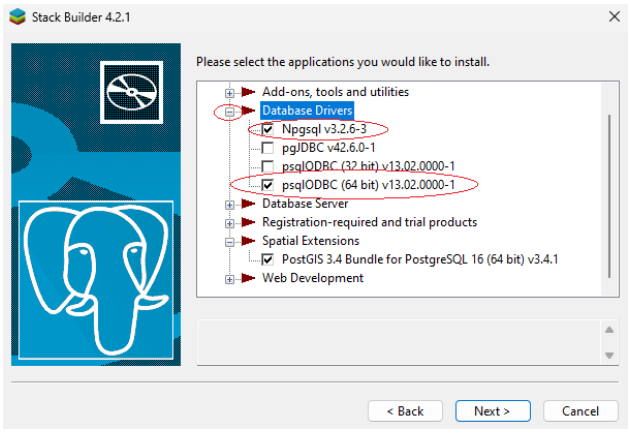


**Figure 18**

15. Review your selections, then click Next.

16. Installers for the three components you selected will now open. Click through them with default settings if you are installing for local use. PostgreSQL and PostGIS are now installed.

## 3.2 Installing DBeaver

1. We will now install DBeaver, which is a graphical interface to manage and query your database. It is very similar to PGAdmin, which is already installed as part of your PostgreSQL package. However, DBeaver comes highly recommended, as it is reputedly faster and more stable. It can be downloaded from: https://dbeaver.io/download/

2. Click the installer. If installing for local use, click through the installer to install in a location of your choice.

3. Upon completion, optionally create a desktop shortcut and click Finish.

4. Start DBeaver.

5. In the toolbar, click Database, and then New Database Connection (Figure 19)

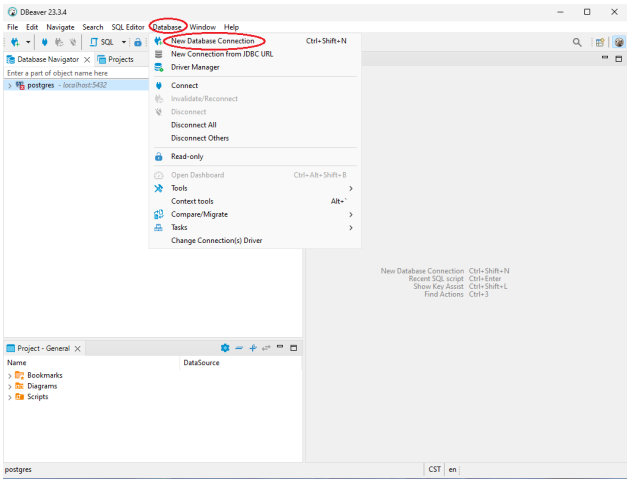


**Figure 19**

6. The 'Connect to a database' window will appear (Figure 20). Ensure PostgreSQL is highlighted like so, and click next.

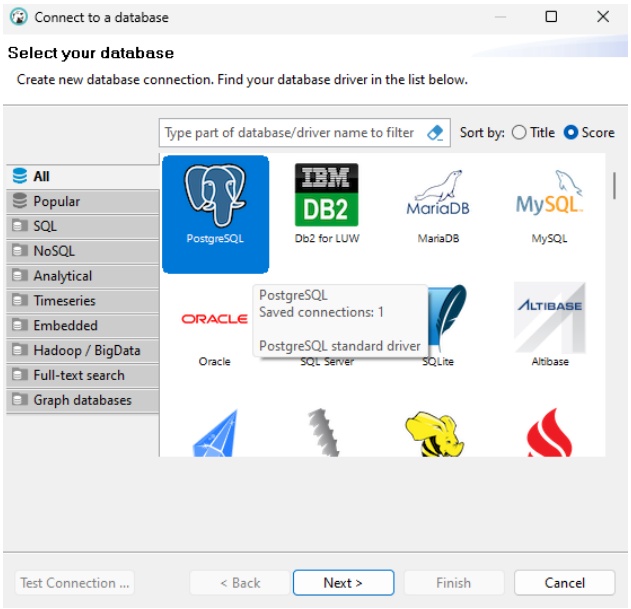


**Figure 20**

7. Enter the password specified above in Section 3.1.7 (Figure 21). If using database locally, leave all other options as default and click Next. Ensure the Show All Databases box is ticked.

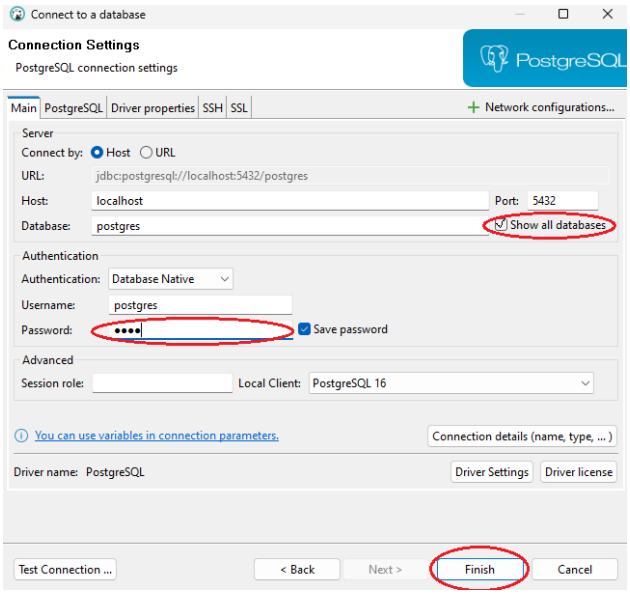


**Figure 21**

8. You should now be connected to the database.

9. Expand Postgres->Databases->postgres->Extensions (Figure 22). You will notice that PostGIS and the other two extensions are not yet present. Right click on Extensions, and press Create New Extension.

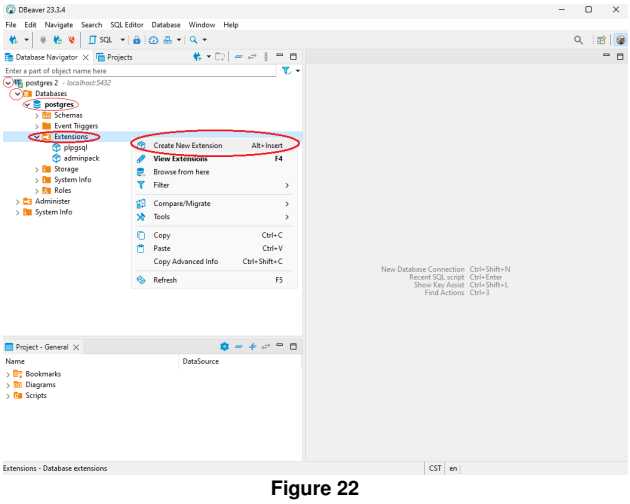


**Figure 22**

10. The Install Extensions window will appear. Set the Schema as "public" and scroll down to select postgis (Figure 23). Consider also selecting the raster and point cloud options if you will be storing these types of data in your server in the future. Click OK.

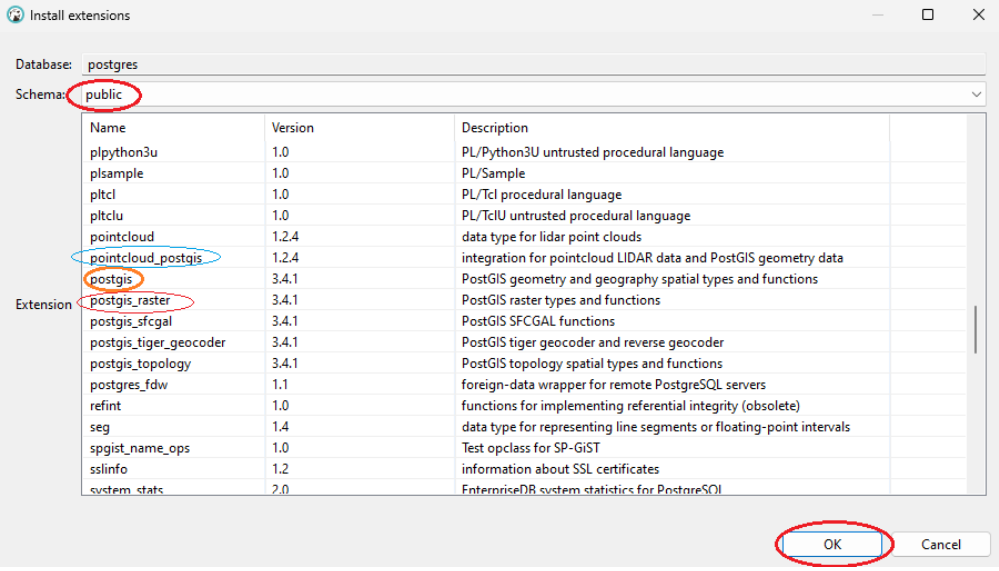


**Figure 23**

11. If done correctly, you should now see postgis as an extension in the expanded list (Figure 24)

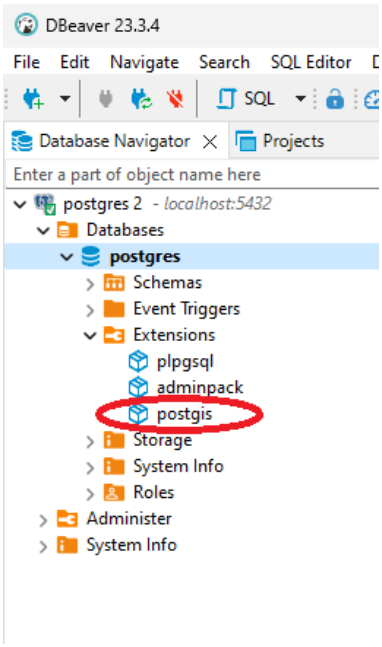


**Figure 24**

# 4 Moving existing data into the database

## 4.1 Loading exported ArcGIS vector data and tables into the database

1. In Section, we exported the required files from ArcGIS Desktop to be loaded into the database. The vector data were exported as .shp files, and non-spatial tables were exported as .dbf files.

2. We will now load these files into the database.

3. Press the start menu or the Windows key, and start typing "PostGIS". You will be presented with the results. The app we are looking for is called "PostgIS PostGIS Bundle 3 for PostgreSQL x64 16 Shapefile and DBF Loader Exporter" (Figure 25).

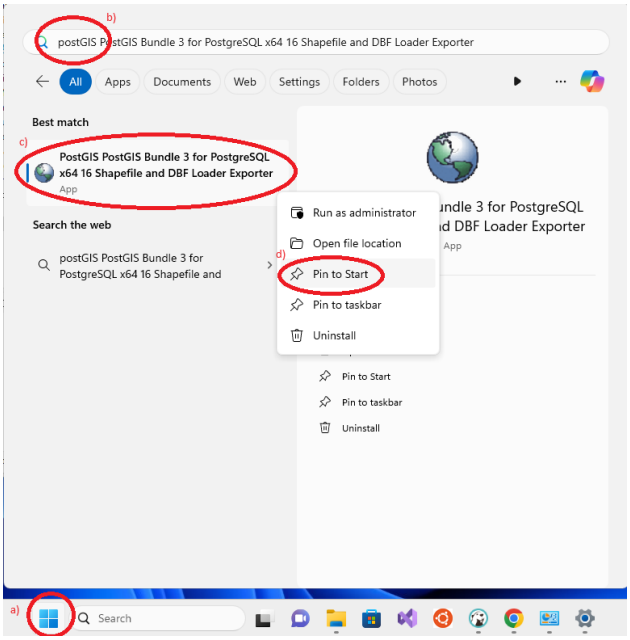


**Figure 25**

4. Optionally, right click and pin it to start for ease of access. Click on the app to run it.

5. The PostGIS Shapefile Import/Export manager will now appear. Press view connection details (Figure 26).

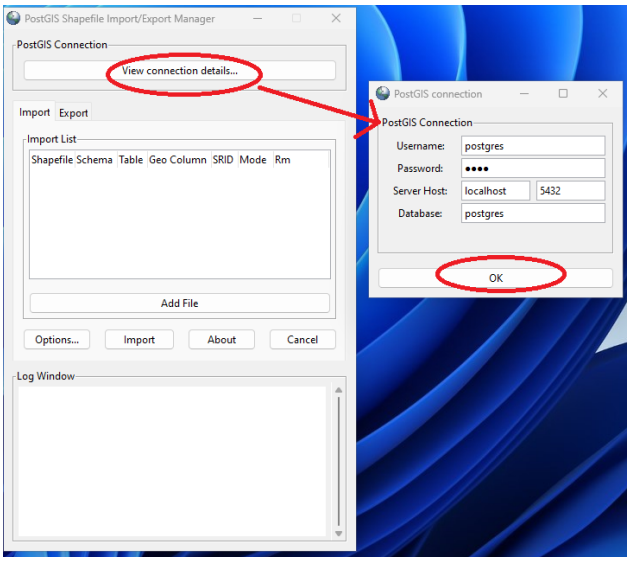


**Figure 26**

6. In the resulting window, fill in the details to connect to the database you created in the previous section. By default, your username is postgres. Keep server host details as default (localhost, port 5432) if working locally. Press OK. (Figure 26)

7. If successful you will see a confirmation message in the Log Window (Figure 27).

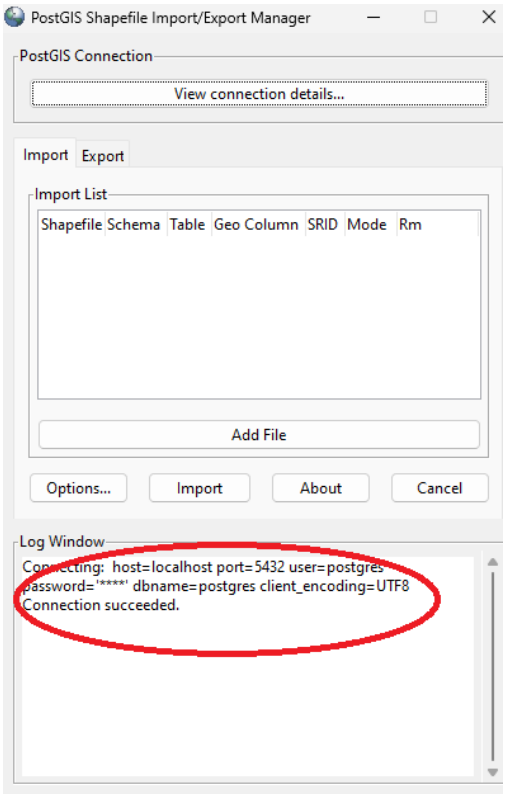


**Figure 27**

8. First, lets upload the vector data, which are stored as .shp files. Press Add File (Figure 28)

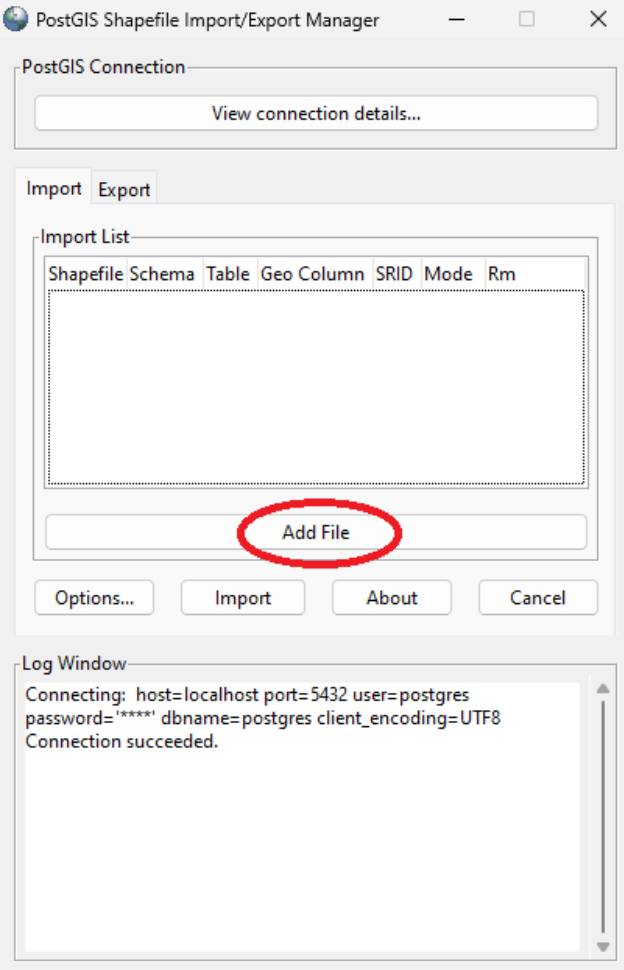


**Figure 28**

9. This will open up the Select a Shape File window. Ensure the file type is set to Shape Files .shp, then use the navigation panel on the left hand side to navigate to where the vector data is stored. Shift click to select all desired files, then press Open (Figure 29).

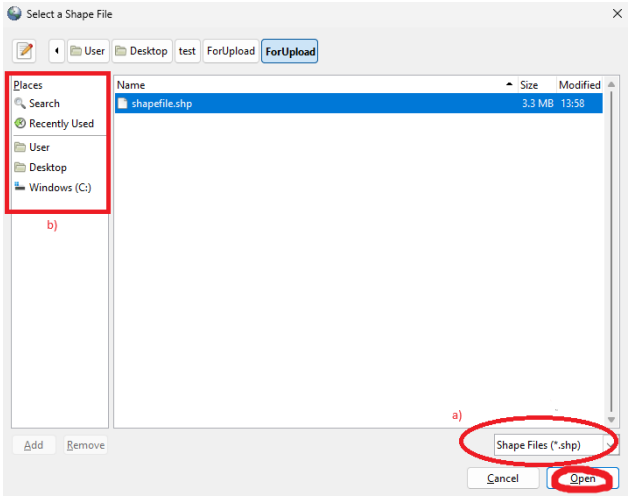


**Figure 29. Note that the file type is specified (a). Use the panel to navigate to the shapefiles (b)**

10. The files will now be added to a list. However, by default, their spatial reference SRID's are set to 0. Double click on **each individual file** under the SRID column and enter the correct number— if necessary, the relevant coordinate system SRID can be looked up on https://epsg.io/ (Figure 30).

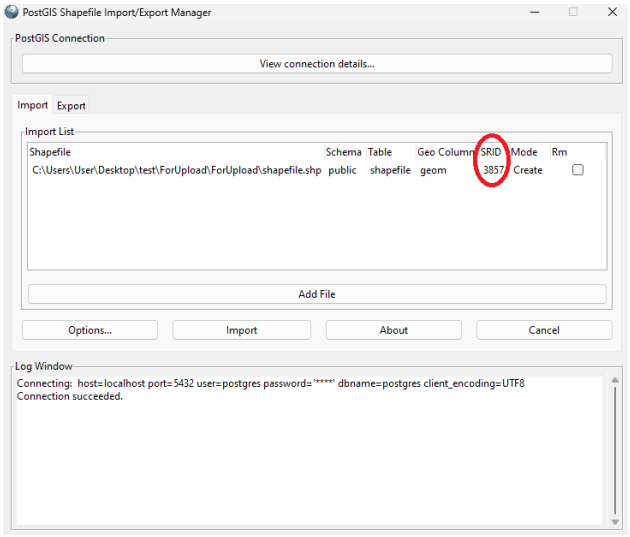


**Figure 30. Note that the SRID for each shapefile must be manually changed by double clicking in the column indicated. If local coordinate grid, leave as 0**

11. Now to add the non-spatial tables, which are stored as .dbf files. Press Add File again (Figure 31)

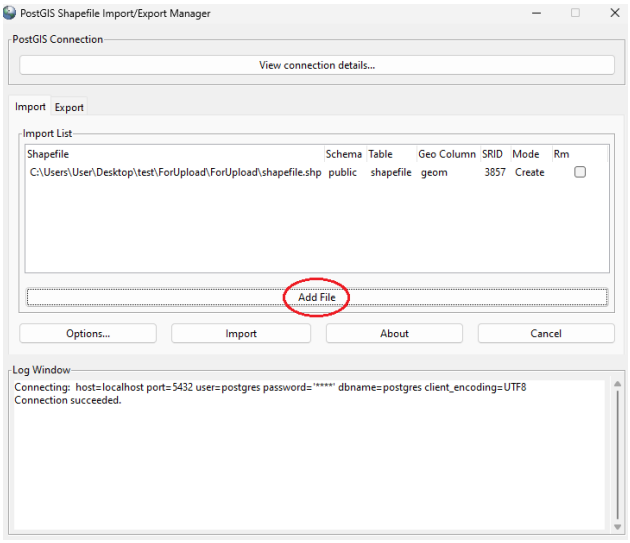


**Figure 31**

12. In the Select a Shape File window, ensure that the file type in the bottom right is set to .dbf, then navigate once again to where the tables are stored. Shift click to select all desired tables, then press open (Figure 32)

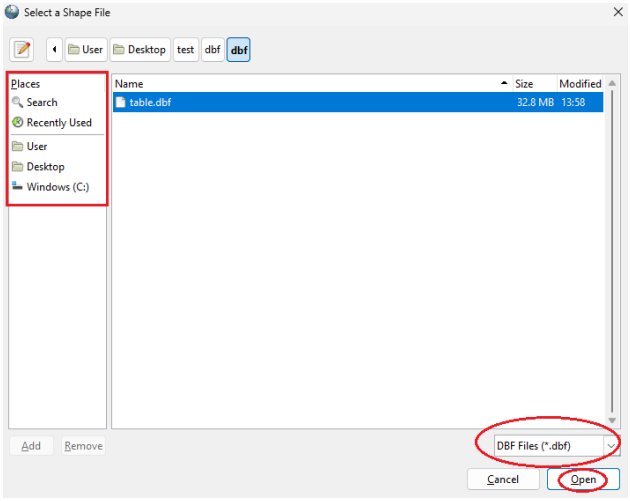


**Figure 32**

13. It is not necessary to change the SRID of the tables if they contain no coordinate data. Click on Import when ready. After doing so, for every file you will see a confirmation that the import was successful. Note that for the table data (i.e. non-shapefile data) it will tell you they cannot be opened, and only attribute data was imported. This is to be expected (Figure 33).

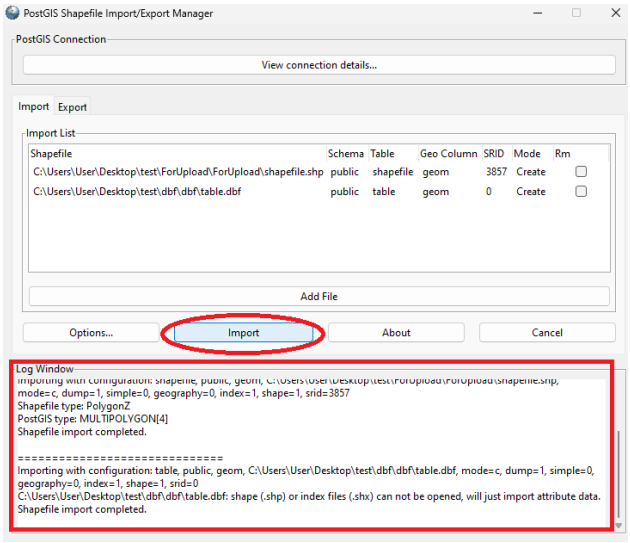


**Figure 33**

## 4.2 Loading other tables into the database

1. It is possible to load non-spatial tables to the database via DBeaver, as long as they are in .csv format.
2. Ensure that all relevant tables are in .csv form. If they are on Microsoft Excel or other spreadsheet software, the Save As function should have the option to do this.
3. Open DBeaver. Navigate to the Schema–> Tables where you want to store the table. Right click on Table, then click on Import Data (Figure 34).

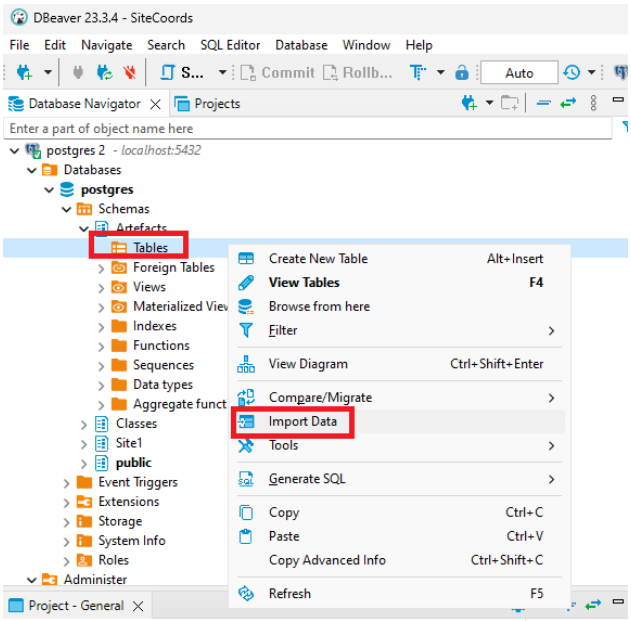


**Figure 34**

4. Follow the instructions on the import wizard that will appear, navigating to where the tables are stored.

## 4.3 Organizing the data with schemas

1. Now that your data has been stored in the server, they can be organized in however way you want.
2. In a database, this is done though the creation of Schema, which essentially serve as "Folders" to organize your data.
3. Open DBeaver. Expand Databases->[Database Name]-> Schemas-> Public-> Tables. You will note that all the data you have uploaded are stored in here (Figure 35).

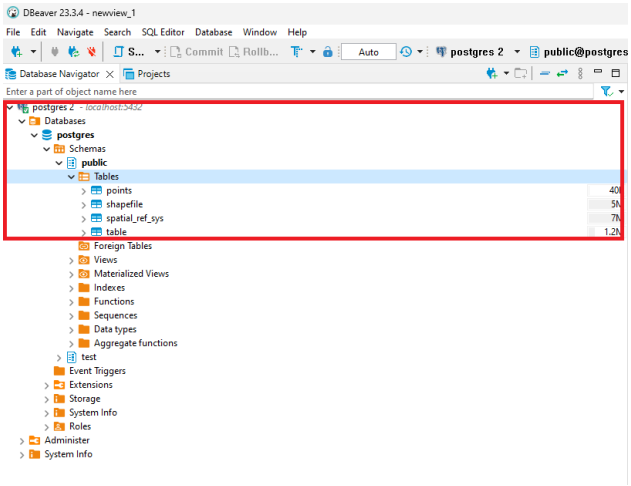


**Figure 35**

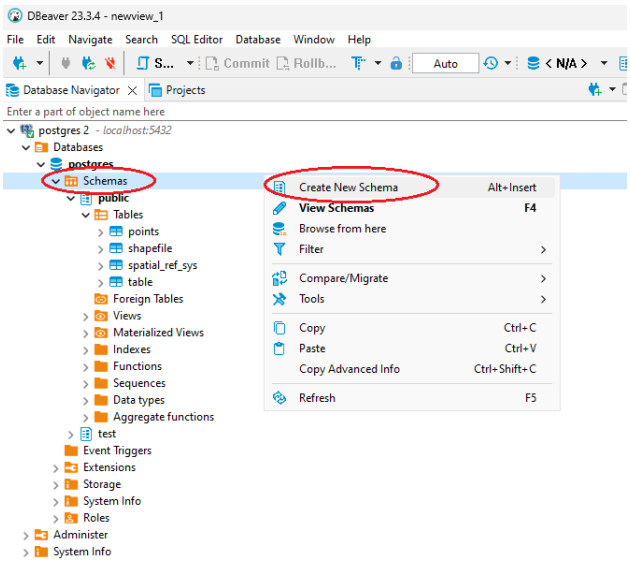


**Figure 36**

4. We will now create a new schema to store data that you want to keep seperate from one another. Right click on Schemas, and then click on Create New Schema (Figure 36)

5. Assign the new schema a name, and then click OK.

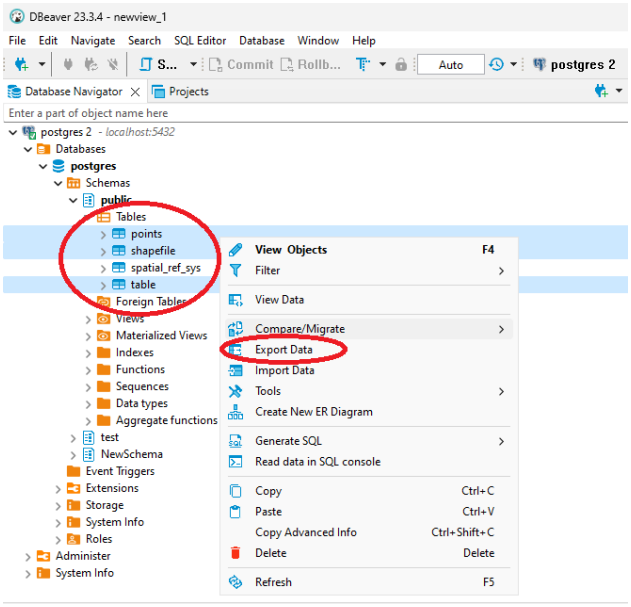


**Figure 37**

6. The new schema should now be present in the database. To transfer the desired files to the new schema, shift left-click or control left-click on the files in the Public schema, then click Export Data (Figure 37).

7. In the resulting window, click next 3 times then click Proceed to complete the transfer (Figure 38).

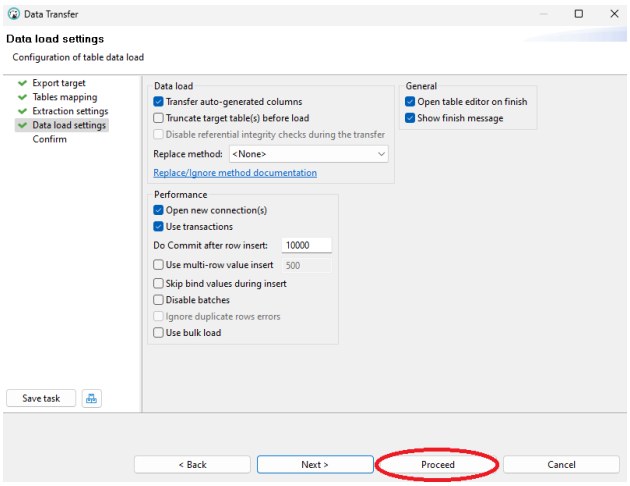


**Figure 38**

8. By expanding the new schema, verify that the desired files have been transferred (Figure 39).

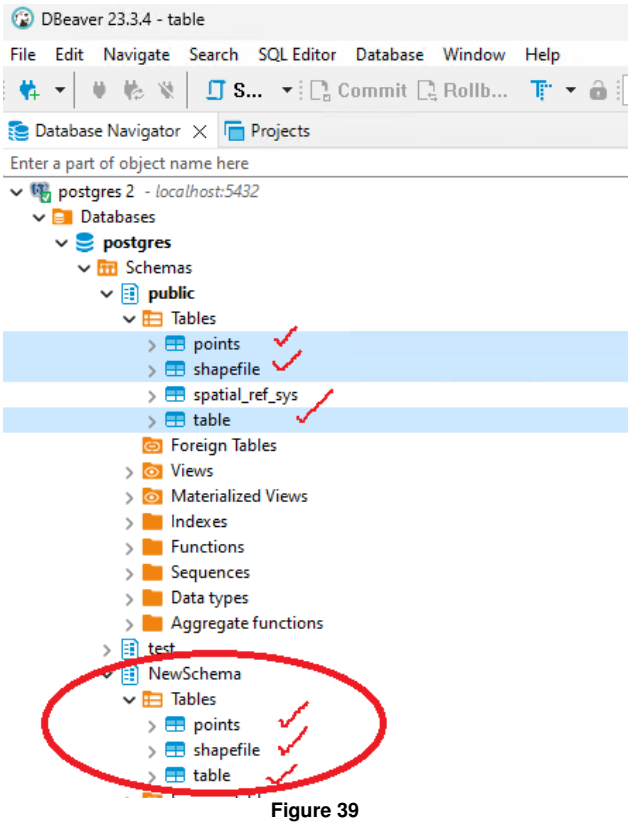
DBeaver 23.3.4 - table
File Edit Navigate Search SQL Editor Database Window Help
Commit
Database Navigator
Projects
Enter a part of object name here
postgres 2 - localhost:5432
Databases
postgres
Schemas
public
Tables
points
shapefile
spatial_ref_sys
table
Foreign Tables
Views
Materialized Views
Indexes
Functions
Sequences
Data types
Aggregate functions
test
NewSchema
Tables
points
shapefile
table

**Figure 39**

# 5 Interacting with the database using QGIS

## 5.1 Connecting to the database

1. Download and install QGIS, an open and powerful GIS software package: https://qgis.org/en/site/
2. Open QGIS. On the left hand browser panel, right click on PostgreSQL, then click on New Connection (Figure 40)

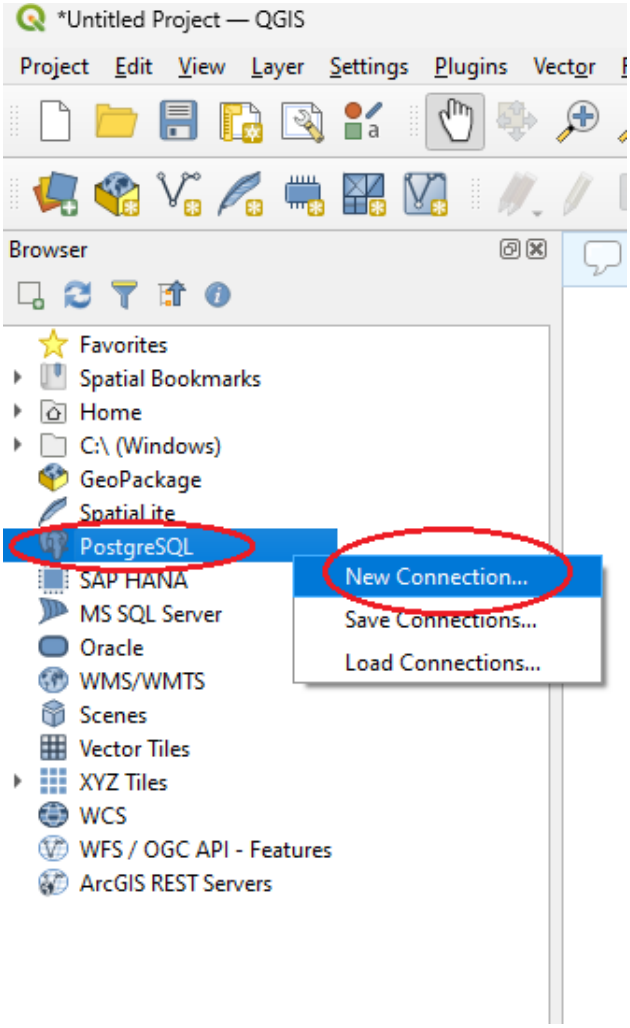


**Figure 40**

3. This will display the Create a New PostGIS Connection window. We will now carefully fill in the details to ensure the connection is successful.
4. This step assumes the user is working locally, vs remotely connecting to a server. If not, please fill in the details with the help of your server administrator to ensure proper security. As shown in Figure 41: Under "Name" type an arbitrary value, like the name of the project— this will be the name of the connection and is not too important. Under host, type "localhost". Leave the Port as 5432. Enter the name of the database that you have been working on in the previous section— be sure to check the name in DBeaver if unsure. Under the Authentication group, click on Basic. Fill in the username ("postgres" by default) and password set above, then tick both boxes next to those fields labelled "Store". This will let you access the database without logging in every time.

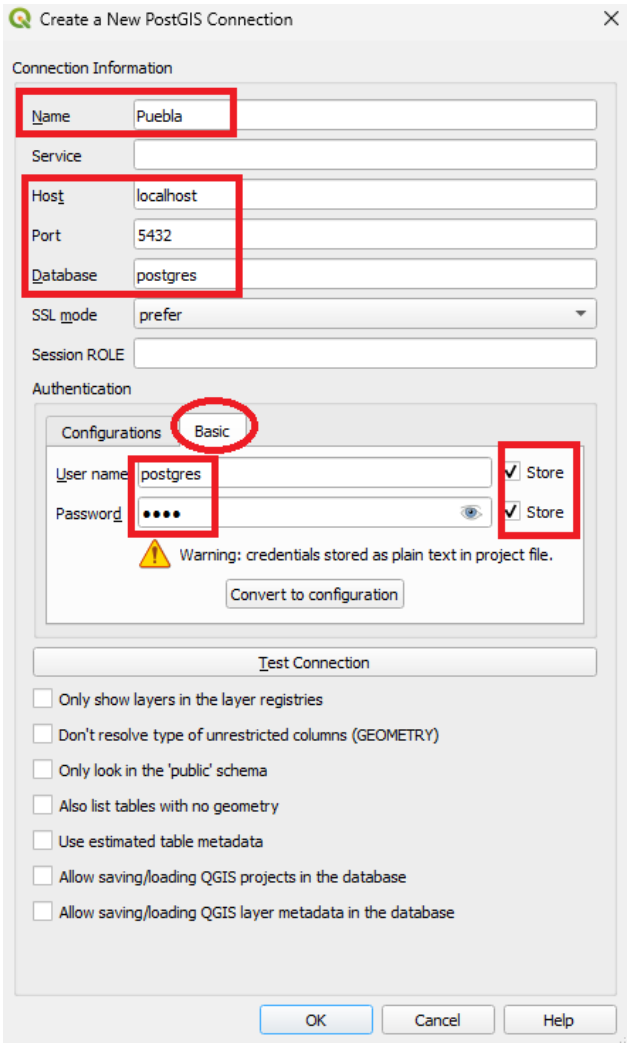


**Figure 41. Note that the Name field can be arbitrary: Write anything!**

5. Press Test Connection. If the details are correct, you will receive a confirmation message on top of the window (Figure 42). Press OK to continue.

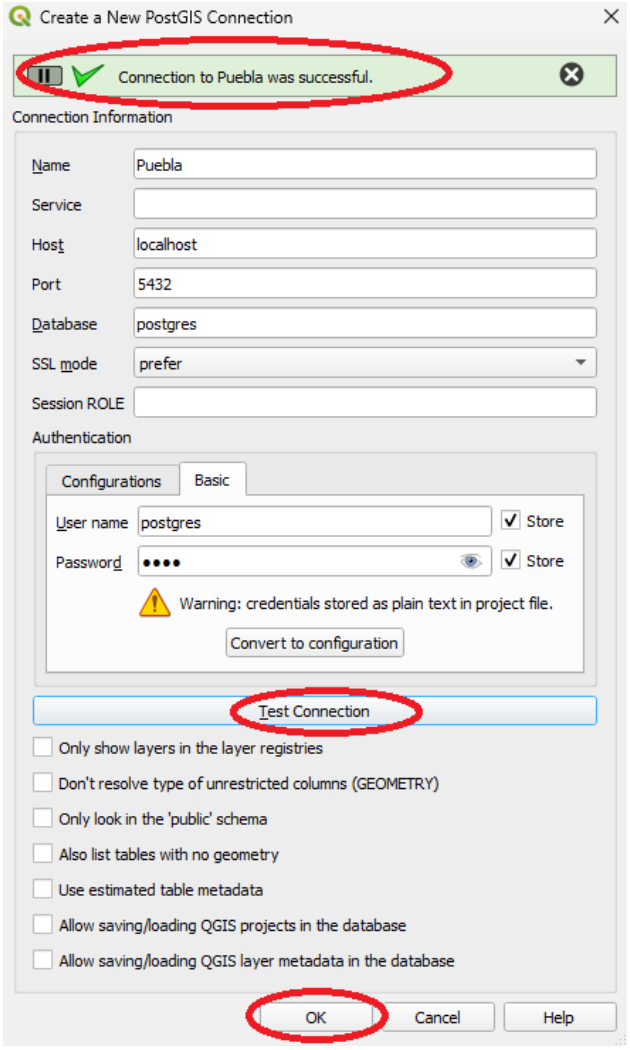


**Figure 42**

6. When the next window appears, press OK to confirm you are saving your password

7. You may now access your data within QGIS in a variety of ways. For example, you may now expand the PostgreSQL group in the left-hand browser panel to see your schema, and any vector data stored within. These may be dragged into the main window to be added as a layer to be viewed and modified in QGIS (Figure 43). Alternatively, you may right click on the shapefile and click Add Layer to Project.

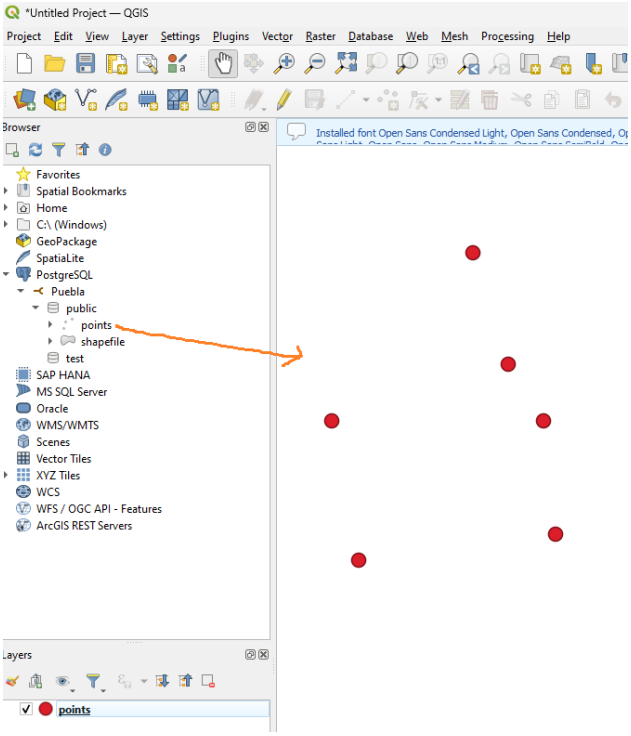


**Figure 43. This data can be dragged into the main screen, as indicated by the arrow**

8. However, you will notice that this method only works for vector data. To add and view your tables in QGIS, in the toolbar click on Database–> DB Manager (Figure 44)

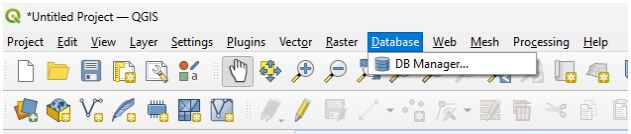


**Figure 44**

9. In the DB Manager window, the PostGIS group can be expanded to show all your Schema and vector/table data. This can likewise be added to your QGIS project by dragging into the main window, or right clicking on the data and clicking Add to Canvas. This will also work for tables (Figure 45)

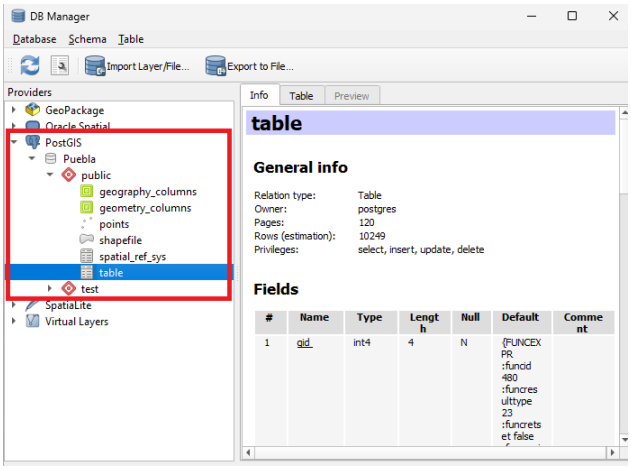


**Figure 45**

10. Save your QGIS project if you plan to work on it further.

## 5.2 Manipulating the spatial data

1. We may now take advantage of the relational nature of this database to carry out spatial analyses within QGIS, like table joins. In this section we will bring both spatial and non-spatial table data from a connected database into QGIS and then analyse them in tandem.

2. We shall explore this through a simple example. This scenario takes place over a small cultural landscape in Alaska, named Site 1. Throughout this landscape, nine artifacts were found through foot surveys. When these items were collected and recorded in the field, there was no time to examine these artifacts in detail. **Only** their locations and unique artifact ID were recorded with a GNSS device, and they were stored in finds bags labelled with their artifact ID, from "1" to "9". Once back in the lab, this spatial data was plotted in a GIS (Figure 46).

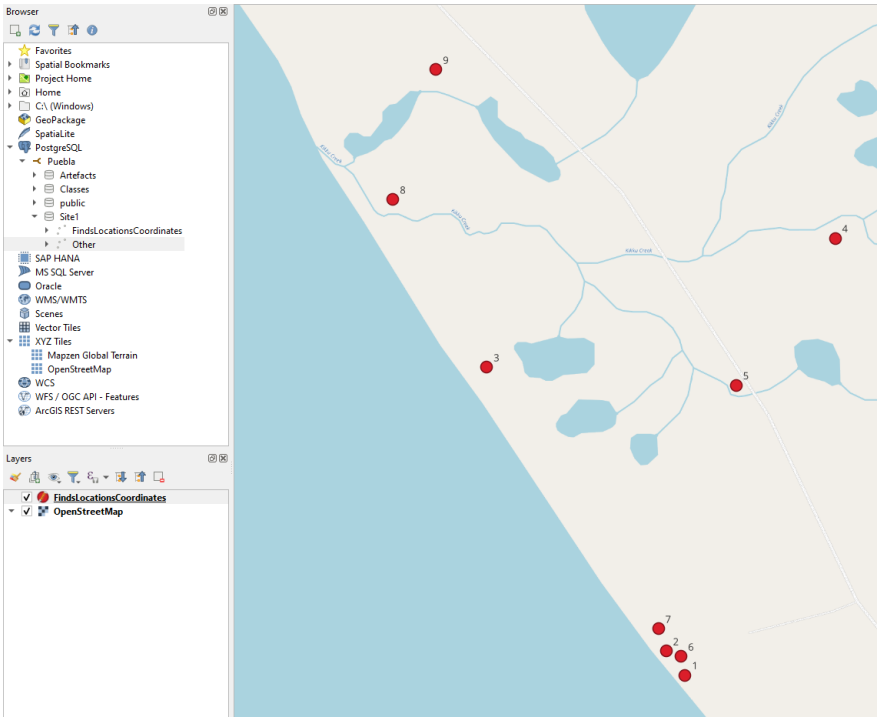

Figure 46

3. Meanwhile, an expert cleaned and catalogued the finds, making note of their ID, material, and type. This was recorded in a table, and then saved to the database (Figure 47).

| ID | Material | Type |
|---|---|---|
| 1 | Wood | Mask |
| 2 | Lithic | Bowl |
| 6 | Wood | Doll |
| 8 | Lithic | Arrowhead |
| 3 | Lithic | Arrowhead |
| 5 | Lithic | Arrowhead |
| 7 | Ceramic | Sherd |
| 9 | Wood | Shaft |
| 4 | Wood | Shaft |

Figure 47

4. At this point of the research it is not possible to make any useful spatial observations about the finds, as the finds table is not associated with real-world coordinates. However, by doing a Join in QGIS between these two tables in the database, it is now possible to examine the data in its entirety.

5. In this example, the coordinate data in the database is already loaded into the QGIS project. To download the table of artifacts, or any other non-spatial table for that matter, on the QGIS toolbar click Database-> DB Manager (Figure 48).

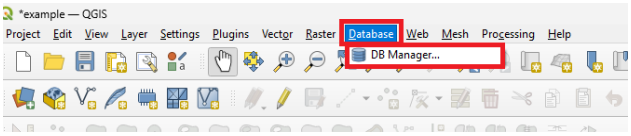


**Figure 48**

6. I have navigated to where the artifact data is stored in the database, which is then left-clicked and dragged into the project (Figure 49).

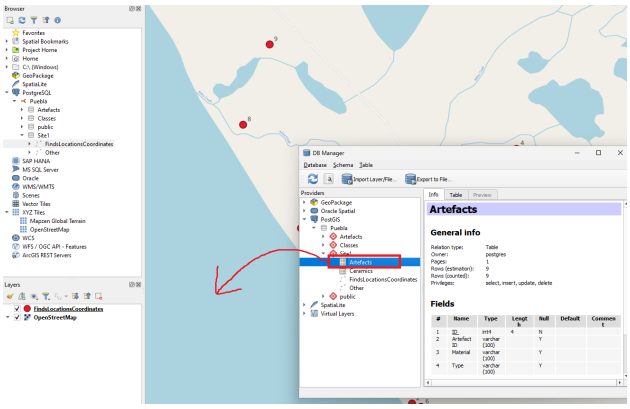


**Figure 49**

7. It should now appear in the layers panel (Figure 50).

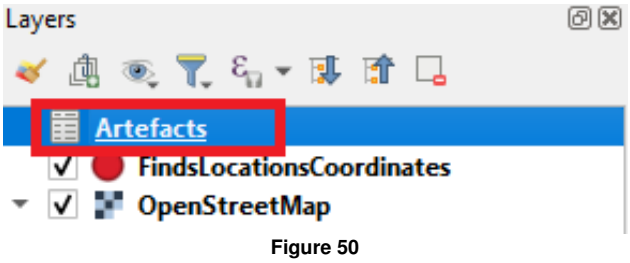


**Figure 50**

8. We can now initiate a join. Right click on the coordinates layer, and then click properties (Figure 51).

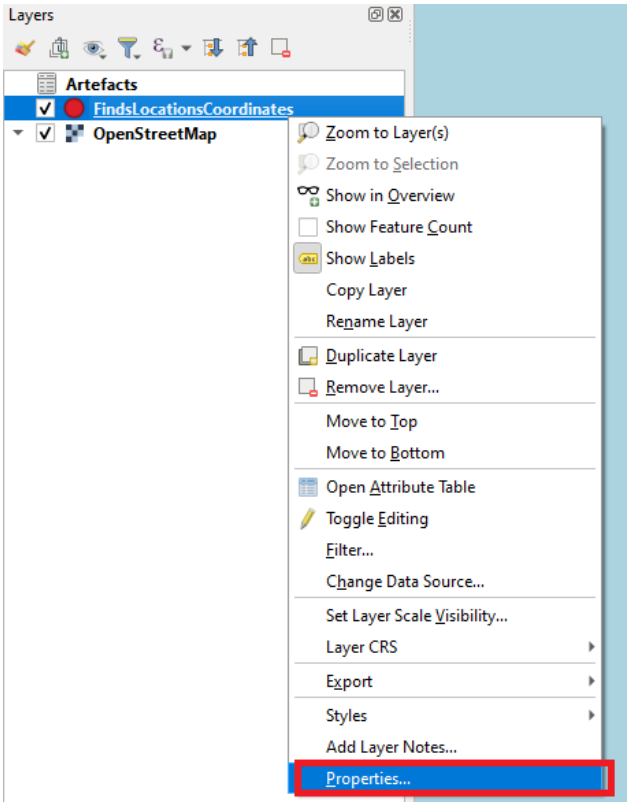


**Figure 51**

9. In the properties menu, click on Joins and press the green plus sign (Add). In the resulting screen, specify the table you have just downloaded, and the join/target fields. Press OK on this window, which should close it. Then click Apply, then OK again (Figure 52).

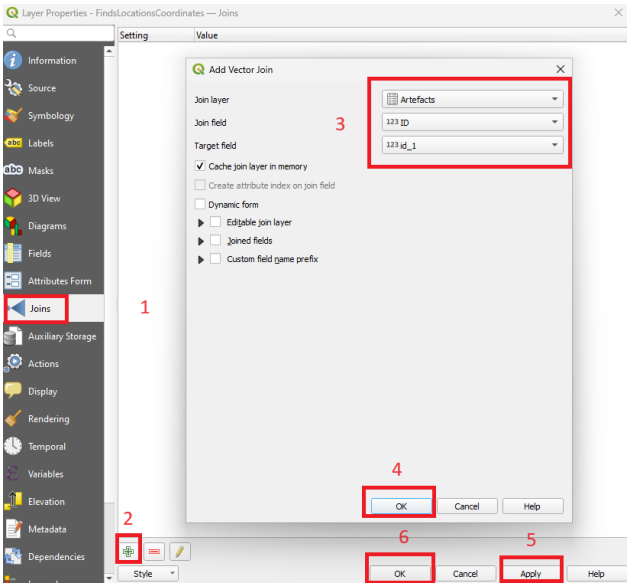


**Figure 52**

10. The tables, both stored in an external database, should now be joined, showing both coordinate and artifact data (Figure 53). This allows spatial observations to be made. In this example, the cluster of points circled in blue are artifacts more associated with domestic activity (Pottery sherd, stone bowl, doll, mask) vs. the subsistence artifacts (Arrowheads and shafts) scattered throughout the landscape. This implies that a village site is close by.

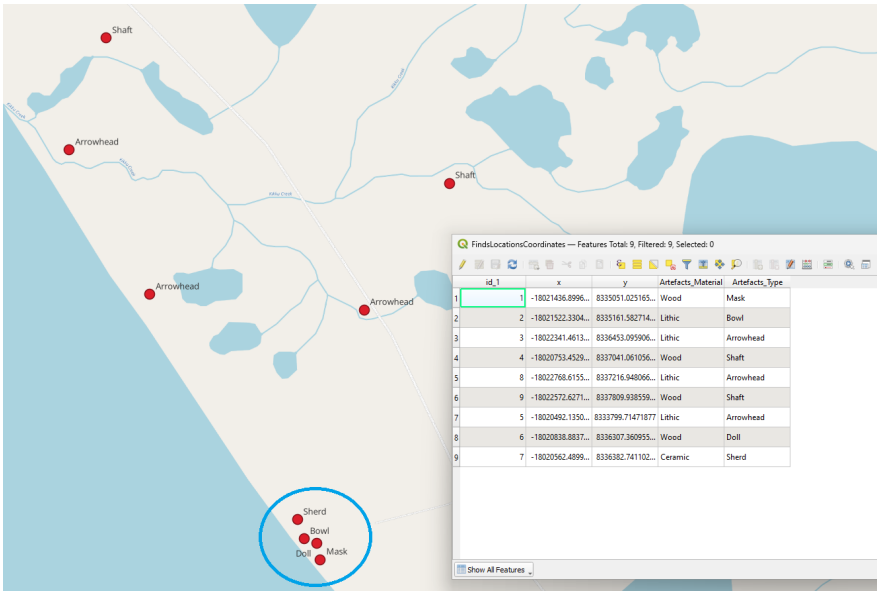


**Figure 53**

# 6 Looking to the Future: Data entry for future projects

## 6.1 Data entry with LibreOffice

1. To explain the principles of data entry, a simple scenario is presented here. Imagine you are tasked with cataloguing all the ceramics on a site. You will need to design 1) **a table** that contains all the relevant information, and also 2) What is known as a "**Form**" — this is an easily understood and intuitive interface for entering new data for your table. The form should also have the ability take advantage of the relational nature of the database, by drawing information from other tables.

2. There are many ways to add data to your new database. One way is through LibreOffice, formerly known as OpenOffice, a free database management suite similar to Microsoft Office. We will be using LibreOffice Base, which is very similar in function to Microsoft Access.

3. First download and install Temurin, an open alternative to the 64 bit Java runtime environment https://adoptium.net/en-GB/temurin/releases/

4. Next, download and install LibreOffice from https://www.libreoffice.org/

5. Start LibreOffice Base (Figure 54).

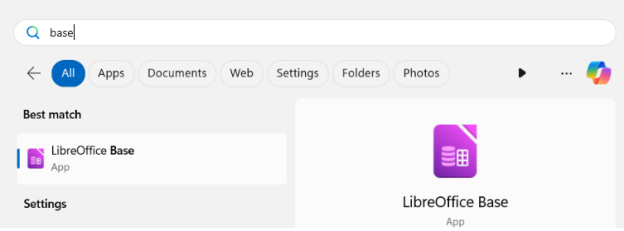


**Figure 54**

6. You will be presented will the Database Wizard window. Tick Connect to an existing database, and select PostgreSQL (Figure 55). Click Next.

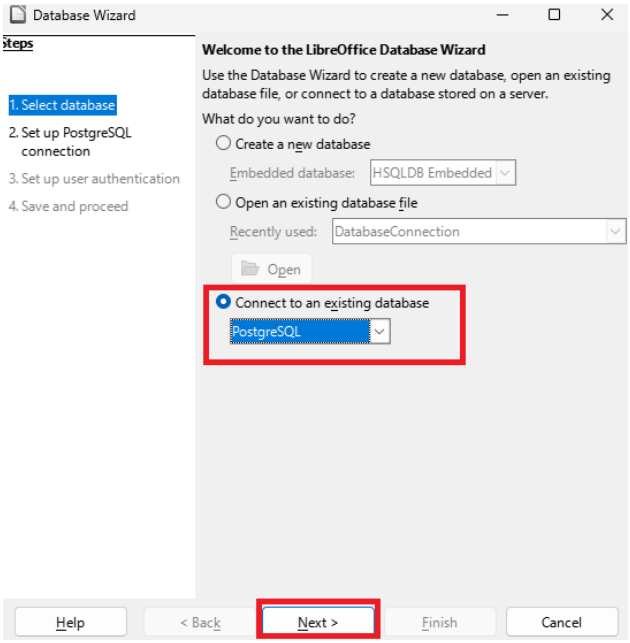


**Figure 55**

7. As per usual, enter the Database name, and also Server and Port Number (localhost and 5432 respectively, if working locally), then click Next (Figure 56).

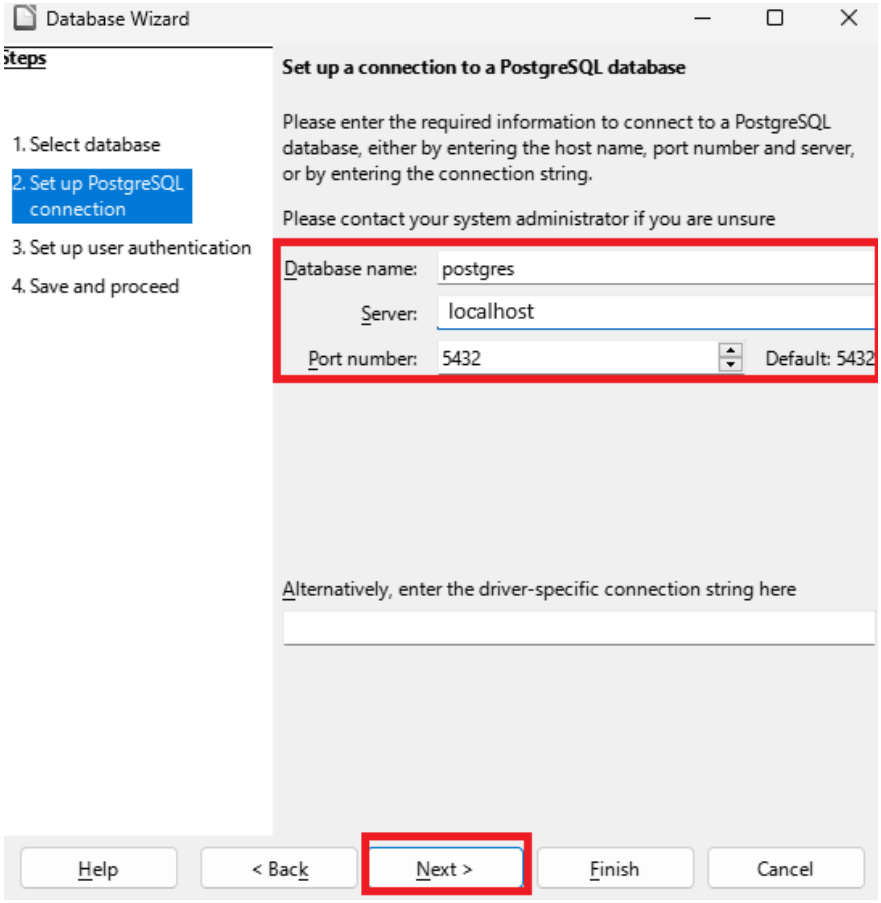


**Figure 56**

8. In the next screen, type your username, and then tick Password required. Click Test Connection, and enter the password when prompted. Press Ok when prompted, and you should be presented with a confirmation of connection if all the details are correct. Click on Finish. Save the Base project file somewhere memorable, if you will be using it regularly to enter data (Figure 57).

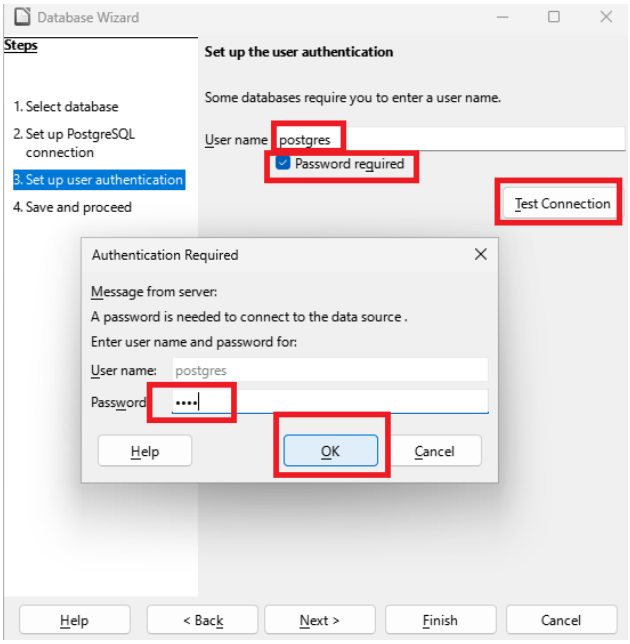


**Figure 57**

9. We shall now create the table to record the ceramics finds. Press Create Table in Design View (Figure 58).

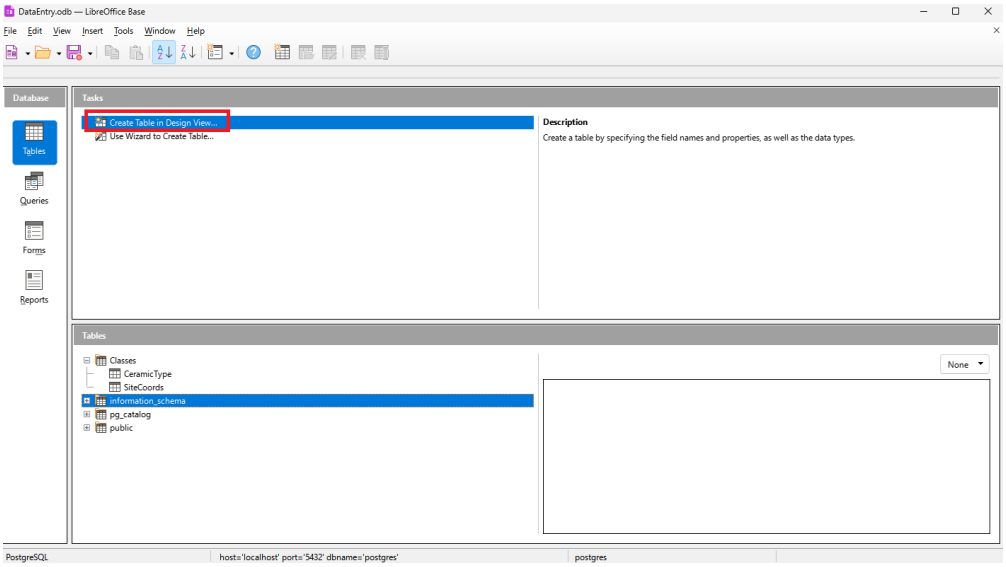


**Figure 58**

10. A blank Table Design window will now appear. This is where you specify the Fields (i.e. Column names) of your table for entering the ceramics data, as well as the data types of each field. In the example I have made in Figure 59, I have created a simple table with a variety of fields of different data types. Two fields, Ceramic Type and Square, are intended to draw from other tables— in this case, two tables named CeramicType and SiteCoords in a Schema named Classes in the geodatabase. They each consist of two columns, an ID field and

a values field (Figure 60). Note that in the table you are creating, the data types of the field must match the values field of the existing tables you are calling the data from.

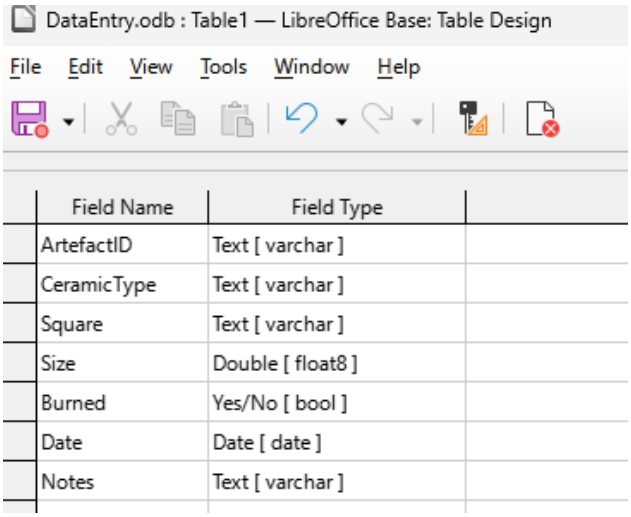


**Figure 59**

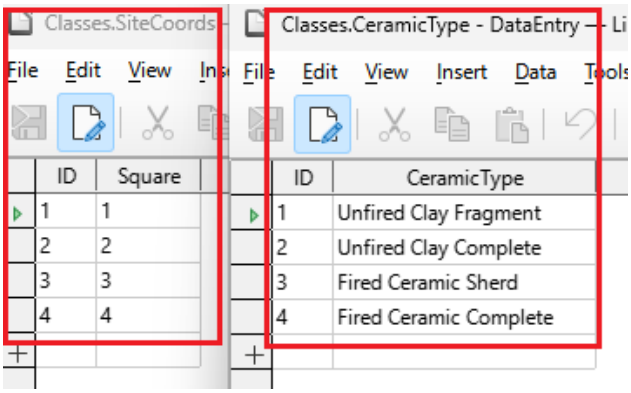


**Figure 60**

11. When finished, press the Save button (Figure 61). When the Save As window appears, specify the Schema you are saving the table to, as well as the table name. In this example I have saved it to a Schema specifically for the site in question, Site 1. Click OK.

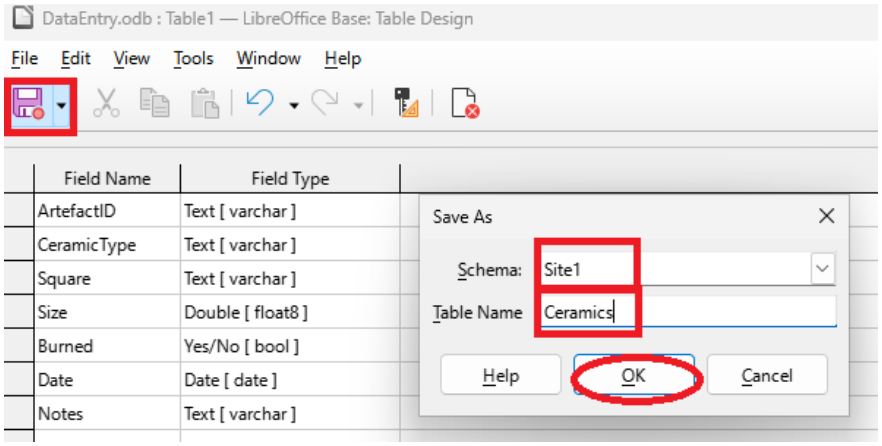


**Figure 61**

12. When prompted to make a primary key, click Yes.
13. Close the Table Design window to return to the main Base screen.

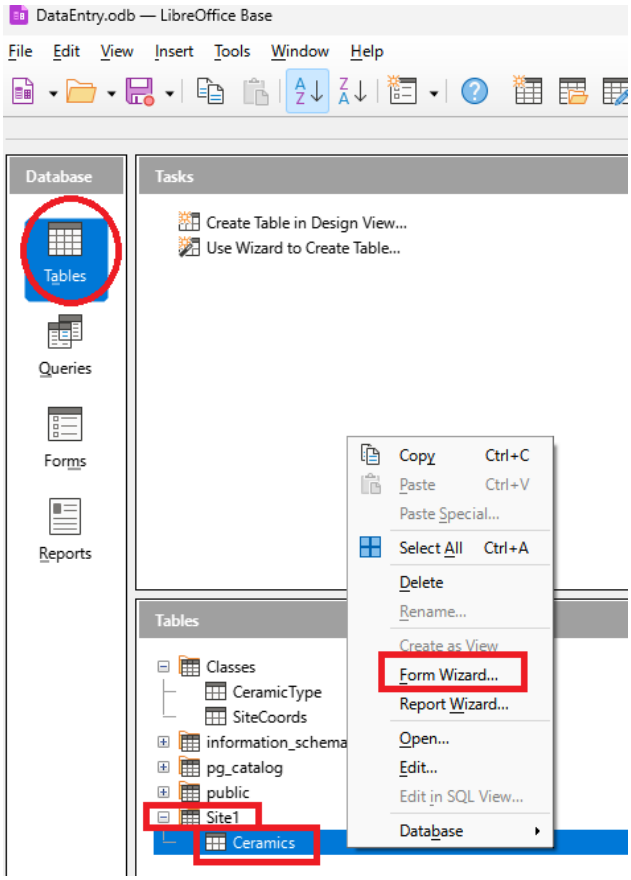


**Figure 62**

14. From this window, you may now navigate to where your blank table was just created. We can now create a form. ***NOTE THAT THIS WORKFLOW ALSO APPLIES TO EXISTING TABLES THAT YOU ARE CREATING FORMS FOR***. Click on Tables->Expand the Schema->Right click on the table->Click on Form Wizard (Figure 62)

15. The Form Wizard screen will appear. Click on the double arrows to add all fields to the form. They will migrate from the left box to the right box, signifying that they will be included in the form. Click Next (Figure 63).

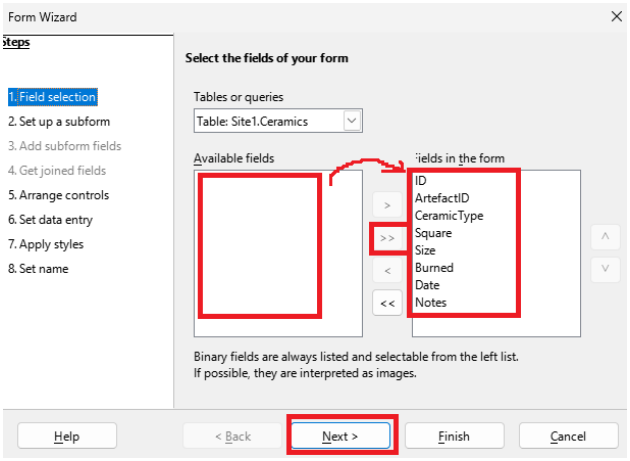


**Figure 63**

16. Do not add a subform. Click Next.

17. We can now begin the process of customising the look of the form. Choose any label alignment and arrangement, and click Next.

18. In the following screen, choose an appropriate option. In our example, we have ticked "The form is to be used for entering new data only. Click Next.

19. In the following page, choose any color and border look. Click Next.

20. In the final page of the form wizard, set the name of the form, and tick Modify the form, as we will be making several changes. Click Finish (Figure 64).

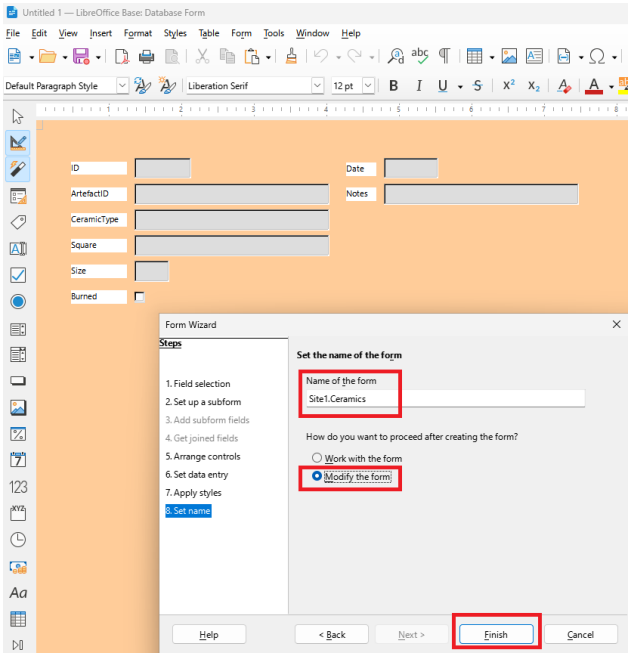


**Figure 64**

21. Here, we can make a wide variety of changes to the form. The fields and data entry boxes can be moved around or changed in size. Text, graphics, hyperlinks to useful resources, can all be added from this window. Use your creativity to make the most useful form that you can, for yourself, and for future investigators carrying out data entry.

22. One powerful way to modify the form is to convert some fields to Combo Boxes: This will allow you to make drop down lists based on other tables elsewhere in the database. Here is an example.

23. As mentioned in Step 10, we want CeramicType and Square to be drop down lists, drawing from two other tables.

24. The data entry boxes for those two fields are currently not suited for purpose and need to be deleted. Right click on CeramicType and click Ungroup (Figure 65).

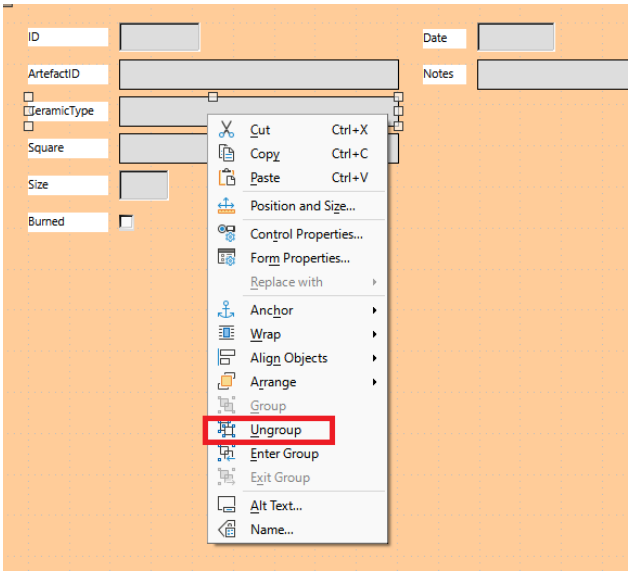


**Figure 65. After ungrouping, click on the grey box and delete it**

25. Next, click on the grey box next to CeramicType and press the delete key on your keyboard to get rid of it.

26. Repeat steps 23 and 25 to delete the grey box next to the field named Square.

27. On the toolbar, click on Form–> Combo Box (Figure 66)

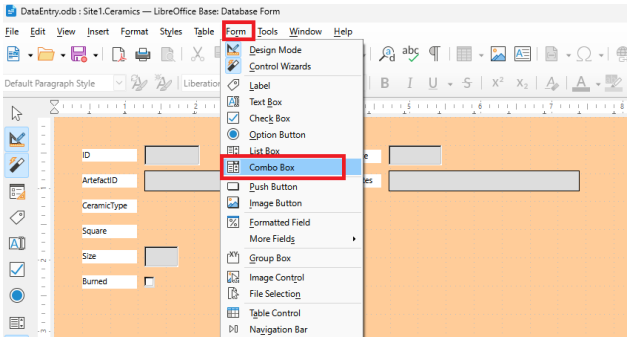


**Figure 66**

28. Hold down the left mouse button and drag to specify the size and location of the combo box. I have put mine next to the CeramicType field. In the resulting window, I have specified that the list should be populated from

an entirely different table—the CeramicType table in the "classes" schema in the database. Click next (Figure 67).

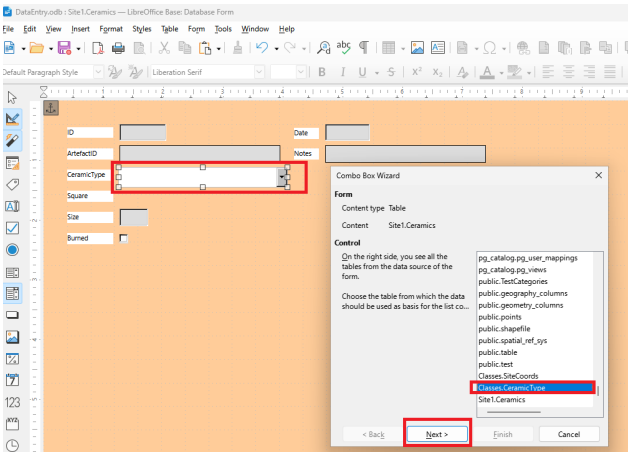


**Figure 67**

29. In the resulting window, I specify the exact **single** field from the other table to be displayed in the drop-down list. Then click next.

30. In the final page of the combo box wizard, I specify that YES, I want to save the values from the drop down list to the CeramicType field on the new table you have created. Click Finish (Figure 68).

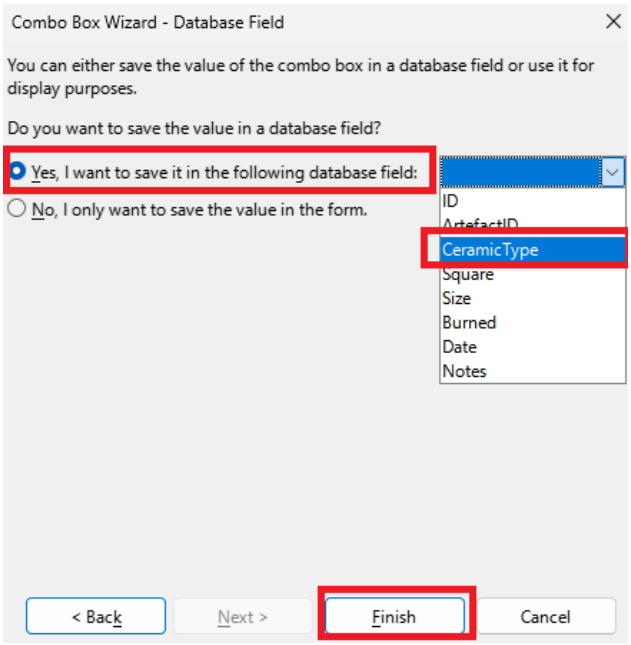


**Figure 68**

31. In my example I repeated the processs with the Square field, tying it to the table called SiteCoords.

32. With the form now complete and in a usable state, we can now switch to data entry mode. In the toolbar, click on Form–> Design mode to allow data entry.

33. The form is now fully functioning as a data entry interface, with drop down menus that are populated from other tables. Click the arrow as indicated in Figure 69 to move to the next record when done.

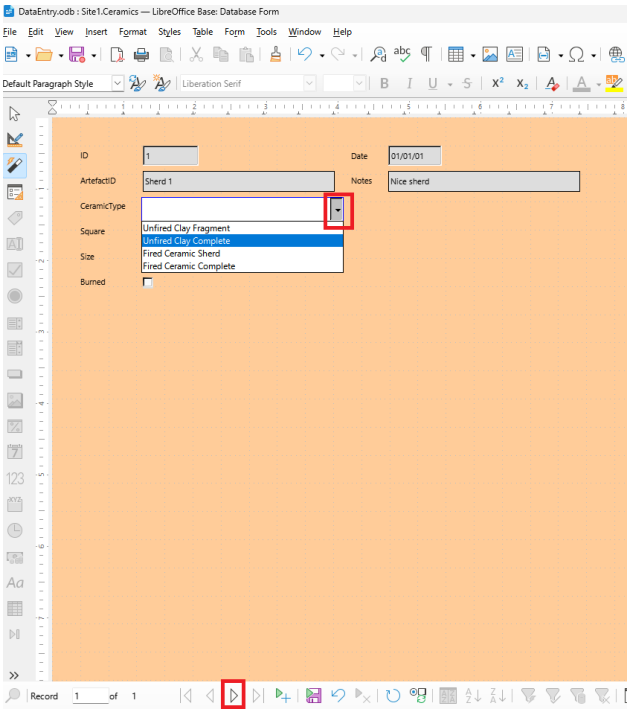


**Figure 69**

34. Exit the table, and saving the data entry when prompted. In the future, this form will be available from the LibreOffice Base main window (Figure 70)

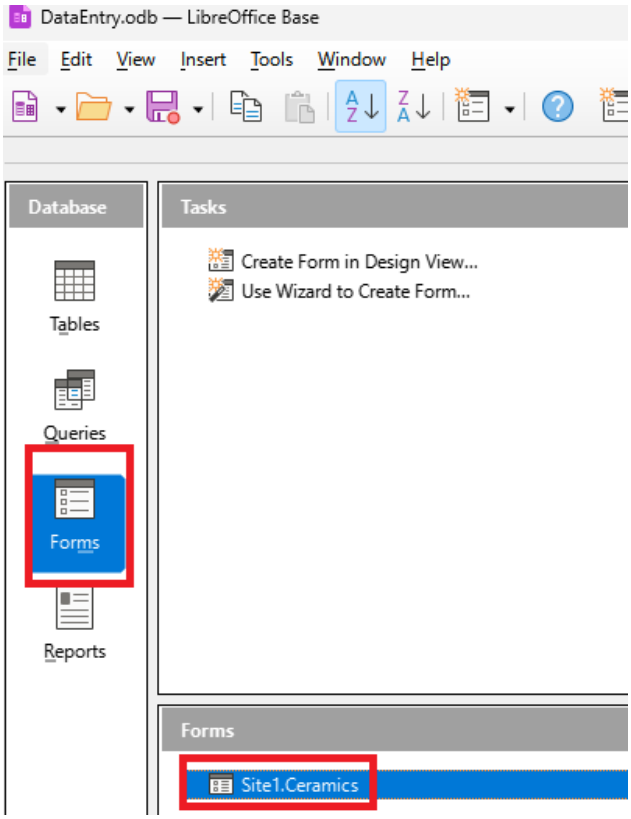


**Figure 70**

## 6.2 QGIS to create new spatial datasets

1. Creating new spatial data for storage in the database is extremely easy using QGIS. When the QGIS client is connected to the database, vector data that is produced can be saved directly to it.

2. To demonstrate this, we will digitize a polygon and save it to the database.

3. Open QGIS, and ensure the client is connected to the database. In the toolbar, click Layer–> Create Layer–> New shapefile layer (Figure 71)

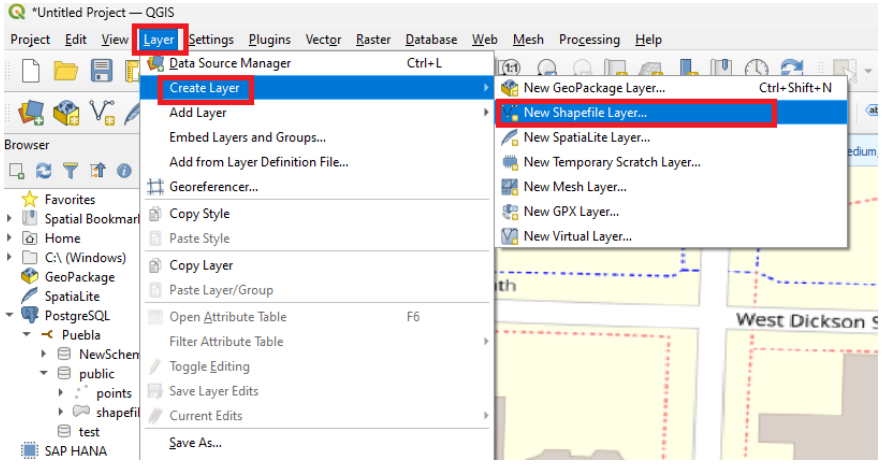


**Figure 71**

4. Assign a file name. Click the Geometry type drop-down arrow, and select any geometry type (Figure 72). Ensure the coordinate system is correct, then click OK

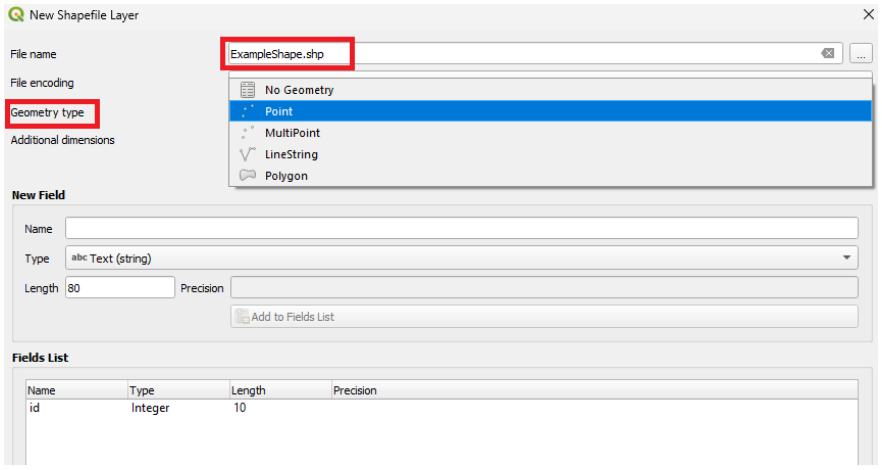


**Figure 72**

5. The empty vector feature layer should now be added to the Layers panel. Right click on the new layer, and then click Toggle Editing (Figure 73)

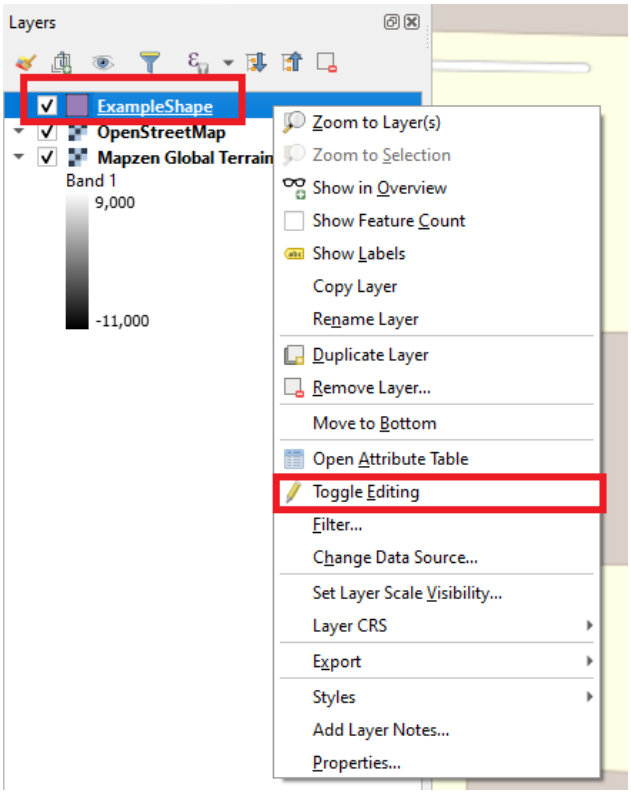


**Figure 73**

6. The feature should now be in editing mode. With the feature selected in the Layers pane, click on Add [Feature type] Feature symbol in the toolbar to begin digitizing (Figure 74)

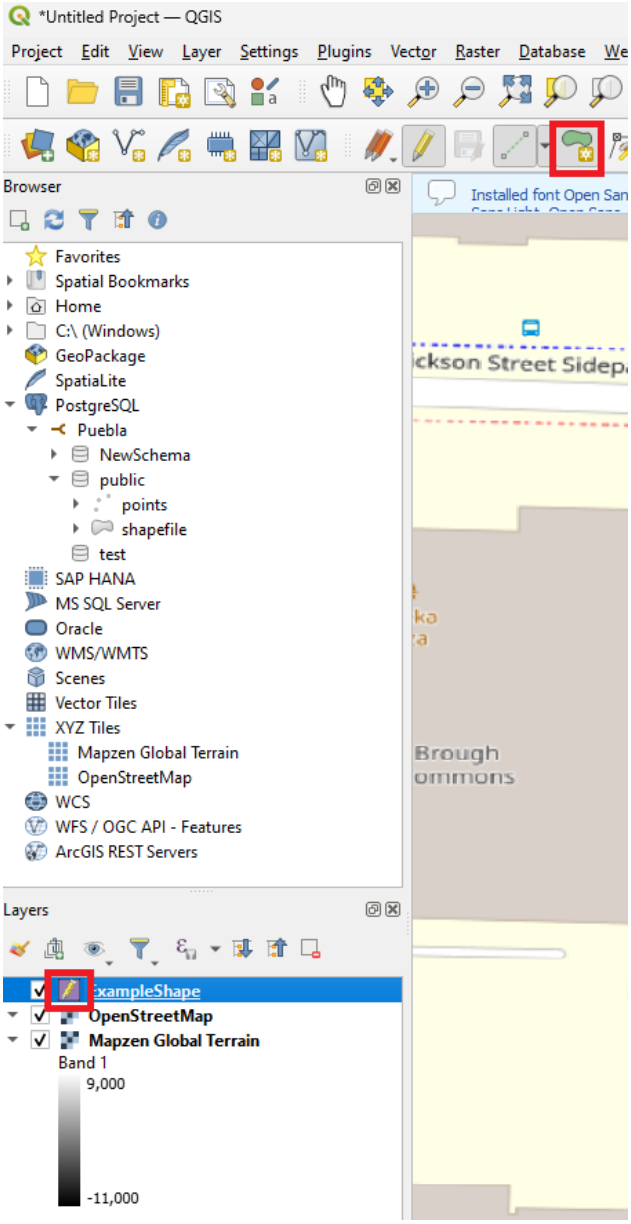


**Figure 74**

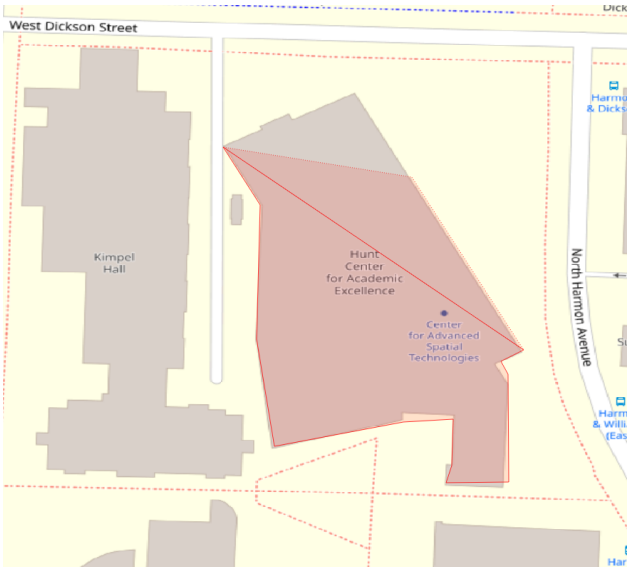


**Figure 75**

7. Start the process by left clicking to draw the feature (Figure 75).
8. To finish, right click to fill in fields (if necessary) and then press OK (Figure 76).

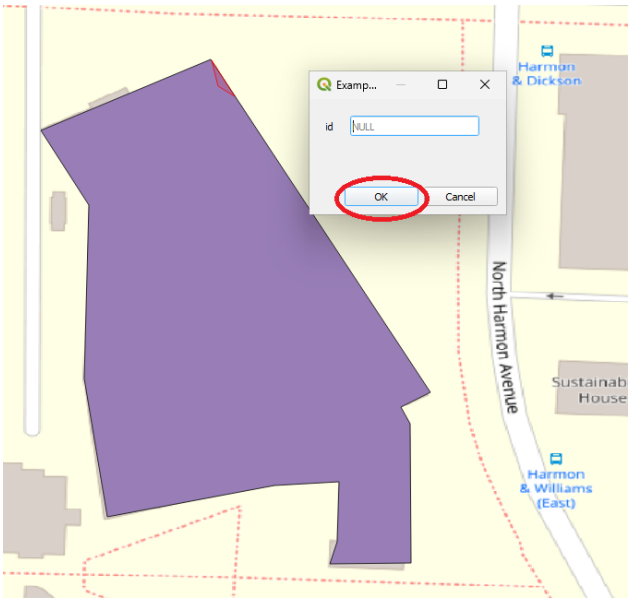


**Figure 76**

9. Right click on the feature layer, Click Toggle Editing to stop editing, and press Save in the resulting window to save edits (Figure 77).

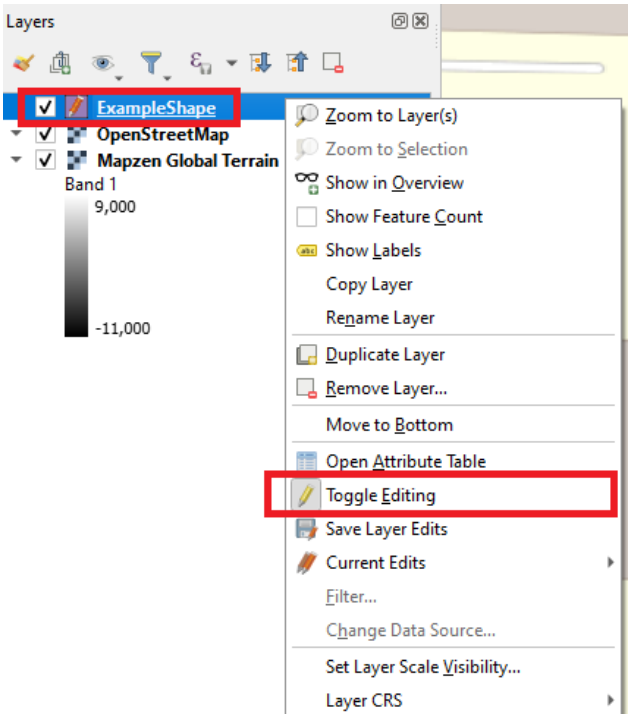


**Figure 77**

10. In the Browser pane, expand the database directory until the Schema you want to save the feature layer into is visible. Simply hold down left click and drag the feature layer to the Schema. An "Import was successful" confirmation window will appear. Click OK (Figure 78).

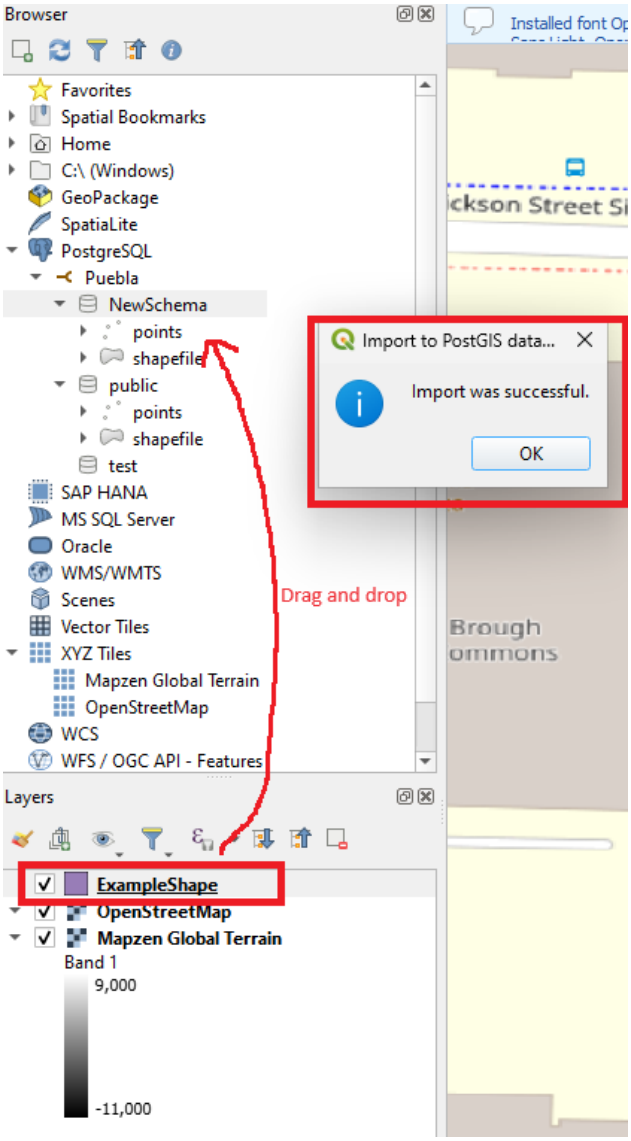


**Figure 78. Drag and drop!**

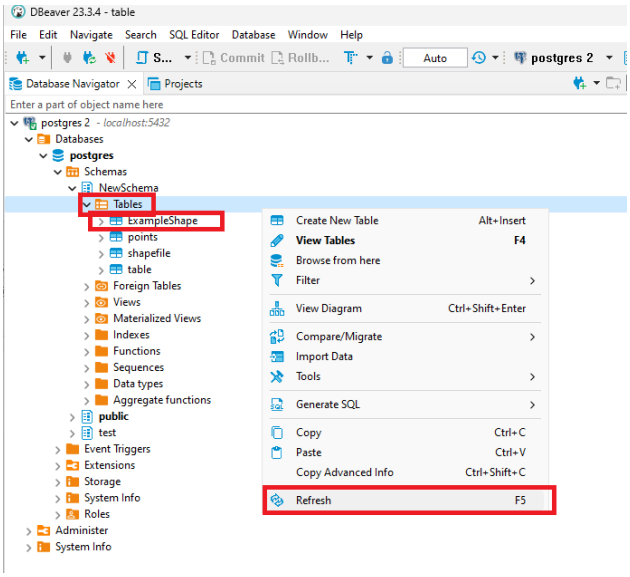


**Figure 79**

11. You will now note that the feature layer has now appeared in the desired schema in the database directory in the QGIS browser window. If needed, you can confirm this by navigating to the same schema in DBeaver. Right click on the Table container and press Refresh, and it should become visible (Figure 79).

# 7 More resources

1. Prof Wu Chuisheng (University of Tennessee) has made an excellent video series on establishing a spatial database with PostgreSQL, which includes more advanced functions like using SQL to query data within PGAdmin. His channel on Youtube is called Open Geospatial Solutions.

2. Data can be easily queried within QGIS https://docs.qgis.org/3.28/en/docs/training_manual/database_concepts/queries.html

3. An alternative method to this is to run queries within the DBeaver console by clicking on SQL Editor in the toolbar. A list a common commands can be found here: https://www.postgresqltutorial.com/