

PRÁCTICA 2.
INSTALANDO POSTGRESQL – POSTGIS EN LINUX

La práctica se entrega de forma individual.

1.1. OBJETIVO

El objetivo de esta práctica es realizar las actividades necesarias para instalar un ambiente de desarrollo con una base de datos PostgreSQL con su extensión espacial PostGIS empleando plataforma Linux. Las instrucciones que se describen a continuación ilustran los pasos requeridos para instalar PostgreSQL 10+ en un sistema operativo GNU/Linux.

1.2. INSTALACIÓN DEL SISTEMA OPERATIVO.

El primer punto a desarrollar es la instalación del sistema operativo. Para ello, se deberán seguir las instrucciones del documento ubicado en la carpeta compartida (google drive):

- `comun/manual-instalacion-linux-mint.pdf` Si se desea emplear Linux Mint
- `comun/manual-instalacion-linux-ubuntu.pdf` Si se desea emplear Ubuntu.
- `comun/manual-instalacion-linux-fedora.pdf` Si se desea emplear Fedora.

C1. Incluir en el reporte la pantalla de usuario configurada en el manual.

1.3. INSTALACIÓN DE POSTGRESQL

Posterior a la instalación del sistema operativo, el siguiente paso es la instalación de la base de datos. Para ello se deberán descargar los siguientes archivos:

- Código fuente de postgresQL
 - Obtener el archivo `postgresql-X.X.X.tar.gz` de la ruta <https://www.postgresql.org/ftp/source/> (versión más reciente). Página principal: <https://www.postgresql.org/>
- Código fuente de PostGIS:
 - Obtener el archivo `postgis-2.x.x.tar.gz` de <http://postgis.net/source> (versión más reciente). Página principal: <http://postgis.net>

En algunos casos es necesario ejecutar las instrucciones como administrador (root). Para ello, se pueden emplear 2 técnicas:

- Anteponiendo el comando `sudo` antes de la instrucción. Esto permitirá ejecutar solo esa instrucción en modo de administrador, se deberá especificar para cada instrucción que requiera permisos de `root`. El comando `sudo` solo puede ser invocado por un usuario que pertenezca al grupo de administradores del sistema, por ejemplo, el usuario que se crea durante el proceso de instalación del sistema operativo se agrega al grupo de administradores por ser el primero. Sin embargo, si se crea otro usuario posterior a la instalación, dicho usuario ya no pertenece al grupo de administradores y no podrá hacer uso del comando `sudo`.
- Escribiendo el comando `sudo su` Esta instrucción cambiará la sesión del usuario actual en la terminal al usuario `root` (notar el cambio del cursor a #). Con esta modalidad no es necesario escribir el comando `sudo` en cada instrucción. Solo tener cuidado de no ejecutar instrucciones que no requieren ser ejecutadas por el usuario `root`. Para salir del modo `root`, ejecutar el comando `exit`.

A. Compilando PostgreSQL.

La instalación se realizará compilando el código fuente de postgresQL, no se emplearán los paquetes (.deb). Antes de iniciar el proceso de compilación, se deberán instalar las siguientes librerías requeridas. Ejecutar las siguientes instrucciones:

- Para Ubuntu/Mint

```
sudo apt-get install libreadline-dev  
sudo apt-get install zlib1g-dev
```

- Para Fedora

```
sudo dnf install -y bison-devel
sudo dnf install -y readline-devel
sudo dnf install -y zlib-devel
sudo dnf install -y openssl-devel
sudo dnf install -y wget
sudo dnf groupinstall -y 'Development Tools'
```

B. Descomprimir el archivo `tar.gz` que contiene el código fuente de PostgreSQL en cualquier directorio, por ejemplo, en el escritorio.

```
tar -xvf postgresql-X.X.X.tar.gz
```

C. Cambiarse al directorio generado, y ejecutar:

```
./configure --prefix=/opt/postgresql-X.X.X/pgsql
```

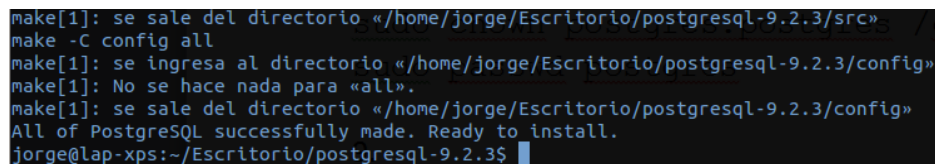
Donde X.X.X corresponde al número de versión de PostgreSQL. Por ejemplo 10.5

La instrucción anterior, verifica que el sistema cuente con todo lo necesario para poder compilar. El valor de `--prefix`, indica el directorio donde se instalará el software.

D. Ejecutar la siguiente instrucción para comenzar el proceso de compilación. Nota: si la instrucción anterior genera error alguno, no se podrá continuar con el proceso de instalación, se deberá revisar el error, corregir y reintentar.

```
make
```

Al final del proceso de compilación, se muestran mensajes similares a los de la siguiente imagen. **C2: Incluir en el reporte una imagen similar** (pequeño extracto del final de la salida).



```
make[1]: se sale del directorio «/home/jorge/Escritorio/postgresql-9.2.3/src»
make -C config all
make[1]: se ingresa al directorio «/home/jorge/Escritorio/postgresql-9.2.3/config»
make[1]: No se hace nada para «all».
make[1]: se sale del directorio «/home/jorge/Escritorio/postgresql-9.2.3/config»
All of PostgreSQL successfully made. Ready to install.
jorge@lap-xps:~/Escritorio/postgresql-9.2.3$
```

E. Si todo va bien, sin errores, ejecutar la siguiente instrucción como root, para instalar el software en la ruta indicada:

```
sudo make install
```

F. Creación del usuario `postgres` (usuario del sistema operativo)

PostgreSQL requiere de un usuario “postgres” de sistema operativo, el cual será designado como dueño de todas las bases de datos que se generen. Ejecutar las siguientes instrucciones para crear a este usuario:

```
sudo mkdir /opt/postgresql-X.X.X/pgsql/data
```

(Reemplazar **X.X.X** por la versión correspondiente en todas las instrucciones que lo requiera).

Observar que el comando anterior crea un directorio “data” dentro del directorio donde se instaló PostgreSQL. En este directorio se guardará toda la información y archivos de configuración de las bases de datos que se crearán. Ejecutar las siguientes instrucciones :

```
sudo adduser postgres
```

En algunos casos, el comando `adduser` puede solicitar información del nuevo usuario como nombre, email, etc. Se puede proporcionar cualquier valor o simplemente presionar `enter` en cada caso. Continuar con la siguiente instrucción la cual modifica el dueño y grupo de la carpeta `data` para que pertenezca al usuario `postgres`.

```
sudo chown postgres:postgres /opt/postgresql-X.X.X/pgsql/data
```

Ejecutar la siguiente instrucción para asignarle un password al usuario `postgres` (se recomienda el valor “postgres” para efectos del curso).

```
sudo passwd postgres
```

G. Para tener disponibles los comandos de PostgreSQL en la variable de entorno `$PATH`, se deberá editar el archivo `/etc/bash.bashrc`, al final del archivo agregar la siguiente línea:

- Para Ubuntu/Mint

```
sudo nano /etc/bash.bashrc
```

- Para Fedora:

```
sudo nano /etc/bashrc
```

Agregar estas líneas al final:

```
#Línea agregada para PostgreSQL
export PATH=/opt/postgresql-X.X.X/pgsql/bin:$PATH
```

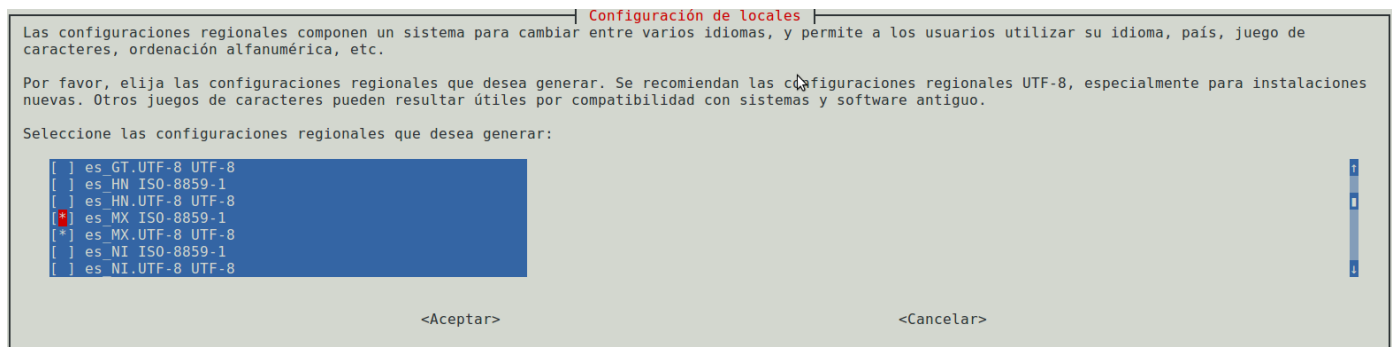
Guardar el archivo, y salir del editor (`ctrl -X`). Para verificar, cerrar la terminal, abrir una nueva (o en su defecto ejecutar `source /etc/bash.bashrc` o `source /etc/bashrc` para actualizar los cambios), ejecutar la instrucción `echo $PATH`. En la salida de dicho comando se deberá apreciar la ruta configurada anteriormente.

H. Juego de caracteres (*Ejecutar este punto únicamente para sistemas Ubuntu /Mint*)

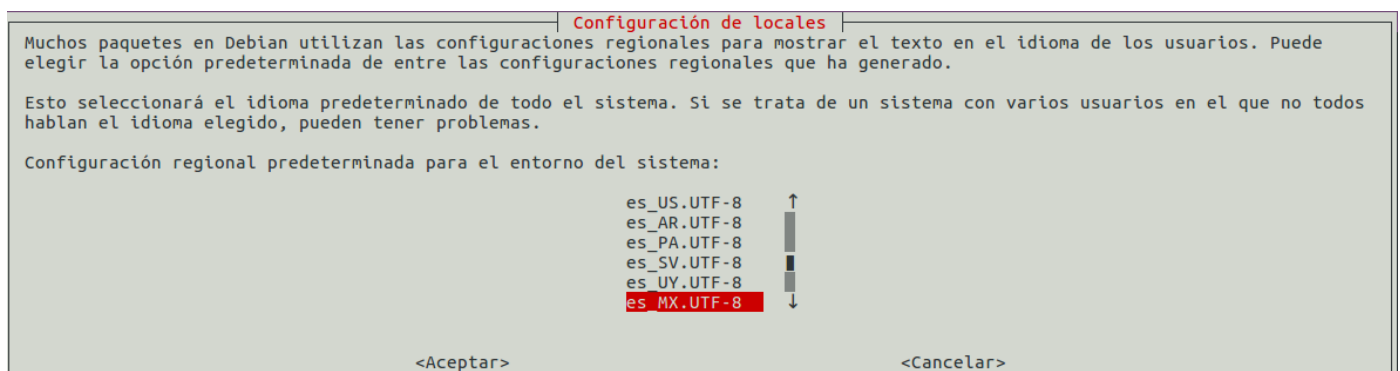
Por default todas las bases de datos creadas en Linux con PostgreSQL emplean como juego de caracteres UTF-8, el cual es el juego de caracteres por default de Linux. Sin embargo, en prácticas posteriores se realizará la importación de datos en un juego de caracteres ISO-8859-1, que contiene acentos, “ñ”, etc. Para evitar que se guarden caracteres extraños debido al juego de caracteres empleados, se deberán ejecutar las siguientes instrucciones para poder crear bases de datos con ISO-8859-1:

```
sudo dpkg-reconfigure locales
```

Aparecerá una pantalla similar a la siguiente. Navegar sobre la lista de juegos de caracteres hasta encontrar el valor `es_MX ISO-8859-1`. Para seleccionar este juego, presionar la barra de espacio.



- Presionar Tabulador para seleccionar la opción <Aceptar>, presionar “Enter” para continuar.
- A continuación, aparecerá una pantalla similar a la siguiente que ofrece modificar el juego de caracteres a emplear durante la instalación de algún paquete para un usuario determinado. Dejar la opción marcada (“es_MX.UTF-8”), seleccionar la opción <Aceptar>



Para verificar que se agregó correctamente, ejecutar la instrucción `locale -a`, deberá aparecer dentro de la lista el valor `"es_MX.iso88591"`.

1.4. INICIALIZAR ÁREA DE ALMACENAMIENTO, INICIANDO LA INSTANCIA.

Antes de realizar cualquier acción sobre la base de datos, es necesario inicializar el área de almacenamiento donde caerán todos los objetos que se crearán (directorio "data" creado anteriormente). En términos de PostgreSQL, a esta área se le conoce como **Database Cluster**.

Un cluster es una colección de bases de datos que son administradas por una instancia simple que se ejecuta en un servidor. Una vez que el cluster es creado, se generan por default las siguientes bases de datos:

- postgres (no confundir con los nombres de usuario).
- template1 (como su nombre lo indica, se emplea como template o plantilla para crear otras bases de datos).

A. Inicializar el cluster. Ejecutar la siguiente instrucción estando como usuario `postgres` (usuario de sistema operativo).

```
su -l postgres
initdb --encoding LATIN1 --locale es_MX.iso88591 -D /opt/postgresql-X.X.X/pgsql/data/
```

Observar que se inicializa el cluster empleando el directorio "data" creado anteriormente. **C3: Incluir en el reporte la parte final de la salida de este comando.**

Una vez ejecutada la instrucción, cambiarse al directorio "data", ejecuta la instrucción `ls -l`, **C4: Incluir en el reporte la salida de este comando.**

B. Creación de los archivos de log. Es importante tener presente donde se guardarán los archivos de log que genera PostgreSQL. En caso de generarse algún error, estos archivos pueden ser de gran utilidad. Ejecutar la siguiente instrucción estando en sesión del usuario `postgres`:

```
mkdir /home/postgres/logs
```

C. Iniciando la instancia. Ejecutar la siguiente instrucción para levantar la instancia (estando en sesión del usuario `postgres`):

```
pg_ctl -D /opt/postgresql-X.X.X/pgsql/data -l /home/postgres/logs/postgreSQL.log start 2>&1 &
```

Presionar "Enter" después de ejecutar la instrucción anterior para liberar la terminal.

La opción `-D` indica el directorio del cluster inicializado, `-l` indica la ubicación del archivo de log. Investigar el significado de la última parte del comando `"2>&1 &"`, **C5: Incluir en el reporte la explicación.**

Para verificar que la instancia está arriba ejecutar `ps -ef | grep postgres`, observar la existencia de varios procesos del sistema operativo que se crean para el correcto funcionamiento de la instancia. **C6: Incluir en el reporte la salida de este comando.**

Para detener la instancia, ejecutar:

```
pg_ctl -D /opt/postgresql-X.X.X/pgsql/data stop
```

1.5. INSTALACIÓN DE PGADMIN

PgAdmin es una herramienta altamente utilizada que permite administrar la base de datos de forma gráfica. Se empleará en el curso para administrar las bases de datos espaciales. PgAdmin 4 representa una evolución de sus versiones anteriores al ser reescrita por completo de C++ a Python. Lo anterior permite ofrecer nuevas funcionalidades como son: acceso vía navegador web entre otras. PgAdmin 4 hace uso extensivo frameworks Javascript: JQuery, Backbone, Bootstrap. Existen 2 modos de operación:

- Desktop runtime (single user mode)
- A través del uso de un servidor Web (multi user mode).

Para efectos del curso se hará uso del modo "single user". Ejecutar las siguientes instrucciones para realizar su instalación.

1.5.1. Instalación en Ubuntu/Linux Mint

Ejecutar las siguientes instrucciones con el usuario principal y el comando `sudo`, o en su defecto como usuario `root`.

A. Importar las llaves de los certificados requeridas por el repositorio de paquetes de PostgreSQL.

```
sudo apt-get install wget ca-certificates
```

B. Obtener el certificado y almacenarlo en el sistema:

```
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
```

C. Crear el archivo `/etc/apt/sources.list.d/pgdg.list` en que se agregará la configuración del repositorio de PostgreSQL:

```
sudo nano /etc/apt/sources.list.d/pgdg.list
```

- Se abrirá un archivo en blanco.

D. Agregar la siguiente línea:

```
deb http://apt.postgresql.org/pub/repos/apt/ <distribution>-pgdg main
```

- Notar la existencia de un espacio después de la última diagonal. (apt/)
- La expresión marcada en rojo se deberá reemplazar con el nombre del código de la distribución Linux que se está empleando. Para determinar este valor se puede revisar el valor del atributo `UBUNTU_CODENAME` en el archivo `/etc/os-release`:

```
more /etc/os-release
...
UBUNTU_CODENAME=bionic
...
```

E. En este ejemplo, la ruta quedará así:

```
deb http://apt.postgresql.org/pub/repos/apt/ bionic-pgdg main
```

F. Guardar cambios y salir del editor `nano`.

G. Ejecutar las siguientes instrucciones para actualizar la lista de repositorios de paquetes en el sistema:

```
sudo apt-get update
sudo apt-get upgrade
```

H. Ejecutar la siguiente instrucción para instalar `pgAdmin4`

```
sudo apt-get install pgadmin4
```

Con estas instrucciones concluye la instalación. Mas adelante se regresará a `pgAdmin`.

1.5.2. Instalación en Fedora.

- A. Similar a Ubuntu, en Fedora se requiere instalar un nuevo repositorio para poder descargar los paquetes de `pgAdmin 4`. Para ello, abrir un navegador web en la siguiente dirección: <https://yum.postgresql.org/repopackages.php>
- B. Ubicar la versión de PostgreSQL instalada y la versión del sistema operativo instalada. Hacer clic en la liga correspondiente.
- C. Al hacer clic, se descargará un archivo `rpm` que será el encargado de configurar el sistema con el nuevo repositorio. Por default el archivo será descargado en la carpeta `Descargas`.
- D. Abrir una terminal y cambiarse al directorio descargas (no usar `root` ni usuario `postgres`):

```
cd ~/Descargas
```

- E. Ejecutar los siguientes comandos para visualizar el archivo descargado y para cambiar los permisos para que pueda ser ejecutado. Los valores `XX` representan a la versión descargada. Sustituir por los valores que aparecen al ejecutar el comando `ls`

```
ls
chmod 755 pgdg-fedoraXX-XX-X.noarch.rpm
```

F. Instalar el archivo `rpm` para crear en nuevo repositorio. Emplear `root` o `sudo`.

```
sudo rpm -ivh pgdg-fedoraXX-XX-X.noarch.rpm
```

G. Ejecutar los siguientes comandos para actualizar la lista de repositorios del sistema:

```
sudo dnf update
```

H. Ejecutar la siguiente instrucción para instalar los paquetes que requiere pgAdmin4.

```
sudo yum install pgadmin4-*
```

Con estas instrucciones concluye la instalación. Mas adelante se regresará a pgAdmin.

1.6. INSTALACIÓN DE POSTGIS

El siguiente paso de esta práctica es la instalación de la extensión espacial de PostgreSQL llamada PostGIS. De forma similar a PostgreSQL, la instalación se realizará a partir del código fuente. Ejecutar las siguientes instrucciones:

A. Instalar librerías requeridas:

1.6.1. Dependencias para Ubuntu/Mint

```
sudo apt-get install libgeos++-dev
sudo apt-get install libxml2-dev
sudo apt-get install libproj-dev
sudo apt-get install libjson-c-dev
sudo apt-get install xsltproc
sudo apt-get install docbook-xsl
sudo apt-get install docbook-mathml
sudo apt-get install libgdal-dev
```

1.6.2. Dependencias para Fedora.

```
sudo dnf install geos-devel
sudo dnf install libxml2-devel
sudo dnf install proj
sudo dnf install proj-devel
sudo dnf install gdal
sudo dnf install gdal-devel
sudo dnf install gdal-libs
```

B. Extraer el contenido del archivo que contiene el código fuente de **PostGIS** en cualquier directorio, por ejemplo, en el escritorio.

C. Cambiarse al directorio creado, ejecutar las siguientes instrucciones para configurar, compilar e instalar:

```
./configure --with-pgconfig=/opt/postgresql-X.X.X/pgsql/bin/pg_config
make
sudo make install
sudo ldconfig
sudo make comments-install
```

Ojo, Sustituir X.X.X con la versión de Postgres.

Existe un archivo que se crea el cual causa conflictos con el comando `psql` de PostgreSQL. Para eliminar este conflicto se deberá renombrar. Ejecutar la siguiente instrucción. Si no existe el archivo, omitir este paso.

```
sudo mv /usr/bin/psql /usr/bin/psql-old
```

1.7. EXPLORANDO LA BASE DE DATOS POSTGRES

Como se mencionó anteriormente, por default, el cluster incluye una base de datos llamada `postgres`. Una vez que se ha levantado la instancia, el siguiente paso es entrar a la base de datos “postgres”.

A. Entrando a la base. Para ello, se emplea el comando `psql` el cual tiene la siguiente sintaxis:

```
psql [DBNAME [USERNAME]]
```

Si se omite el nombre de la base se toma como nombre, el username con el que se invoca al comando `psql`. Lo mismo ocurre con el valor de `USERNAME`. Por lo tanto, estando como usuario postgres a nivel de sistema operativo, se puede escribir:

```
psql
```

Es posible invocar al comando `psql` desde otro usuario, por ejemplo:

```
psql postgres postgres
```

Estando como usuario postgres, aparecerá una salida como la siguiente:

```
postgres@lap-xps:~$ psql
psql (X.X.X)
Type "help" for help.

postgres=#
```

Observar el símbolo `=#`, el cual indica que estamos dentro de la base de datos postgres como super usuario.

B. Desplegar la hora y la versión con postgres:

Ejecutar los siguientes comandos, **C7: Incluir la salida en el reporte.**

```
postgres=# select now();
postgres=# select version();
```

Para salir de la línea de comandos, escribimos `\q`

1.8. CONFIGURANDO SEGURIDAD EN POSTGRES

Antes de iniciar con la configuración de la seguridad, es necesario asignar un password al usuario postgres que se genera al crear el cluster (No confundirse, este usuario es un usuario interno de PostgreSQL, el creado anteriormente es un usuario a nivel de sistema operativo). Para realizar esta actividad, ejecutar la siguiente instrucción estando dentro de la base de datos postgres (observar el cursor):

```
postgres=# alter user postgres with encrypted password 'postgres';
```

Por default, la base de datos no cuenta con restricción alguna para acceder a cualquiera de las bases de datos del cluster. Por tal razón, fue posible ejecutar los comandos de la sección anterior sin emplear algún método de autenticación. El primer paso a realizar es proporcionar un medio de autenticación al usuario postgres.

Para realizar la configuración de accesos se deberá editar el archivo `pg_hba.conf` que se encuentra en el directorio `data`. Es decir, se deberá ejecutar el siguiente comando (como usuario postgres)

```
nano /opt/postgresql-X.X.X/pgsql/data/pg_hba.conf
```

1.8.1. Configurando acceso a través de línea de comandos empleando la identidad del sistema operativo.

Significa que el usuario postgres podrá tener acceso a la base de datos siempre y cuando el usuario que intente entrar haya iniciado sesión a nivel de sistema operativo con el usuario postgres. Para realizar lo anterior, editar el archivo arriba mencionado, y verificar que la siguiente línea coincida con lo siguiente (casi al final del archivo):

#	TYPE	DATABASE	USER	ADDRESS	METHOD
#	"local"	is for Unix domain socket connections only			
local	all		postgres		ident

`local`: se refiere a conexiones realizadas desde el servidor empleando una terminal.

`all`: El usuario podrá tener acceso a todas las bases de datos.

`postgres`: Solo el usuario llamado "postgres" podrá tener acceso empleando esta configuración.

`ident`: Verifica que el usuario que intenta conectarse a la base de datos haya entrado a sesión a nivel de sistema operativo con un usuario llamado "postgres". Si esto se cumple, se permite el acceso.

Importante: Asegurarse de que las líneas restantes del archivo comiencen con “#”, es decir, agregar # al inicio de la línea para comentarlas.

Para comprobar esta configuración:

- C. guardar cambios en el archivo (se recomienda hacer un respaldo)
- D. Reiniciar la instancia.
- E. Intentar ejecutar el siguiente comando estando en sesión del usuario `postgres`:
`psql` En este caso se deberá tener acceso a la base sin problema alguno.
- F. Cambiarse de sesión a nivel de sistema operativo empleando un usuario diferente a `postgres`.
- G. Ejecutar `psql postgres postgres`
- H. El sistema deberá mandar un mensaje de error indicando que no es posible conectarse, esto debido a que el usuario en sesión no es el usuario `postgres`. **C8: Incluir en el reporte** la salida de la pantalla que muestre el mensaje de error de autenticación

1.8.2. Autenticación local para todos los usuarios empleando password encriptado.

Para permitir que cualquier usuario pueda conectarse a la base de datos a línea de comandos empleando su password, se deberán agregar las siguientes líneas del archivo para que quede de la siguiente forma:

```
#Se agrega acceso a todos los usuarios a línea de comando
local all all md5
```

Como se puede observar, en esta configuración, todos los usuarios podrán entrar a la base de forma local empleando el método md5, que significa: acceso empleando un password encriptado.

1.8.3. Autenticación para conexiones TCP/IP

Se requiere agregar esta última configuración para permitir que todos los usuarios puedan conectarse de forma adicional a través de una conexión TCP/IP, en particular a partir de pgAdmin (más adelante se verá su uso). Para ello, editar el archivo, (no se requiere agregar líneas nuevas) verificar que coincidan con lo siguiente:

```
# IPv4 local connections:
host all all 127.0.0.1/32 md5
```

Como se puede observar, en esta configuración todos los usuarios podrán conectarse vía TCP/IP, únicamente desde la misma maquina donde está el servidor de la base de datos (127.0.0.1/32 localhost). Si se desea conectarse desde otra IP, se deberán especificar en este archivo.

Al terminar de configurar la seguridad, reiniciar la base de datos para que los cambios tengan efecto.

1.9. CREACIÓN DE UNA BASE DE DATOS CON EXTENSIÓN ESPACIAL

El siguiente paso es crear una base de datos con extensión espacial empleando la instalación de PostGIS realizada anteriormente. Ejecutar los siguientes comandos.

- A. Estando conectado a la base como usuario `postgres`, ejecutar la siguiente instrucción para crear al usuario que será el dueño de nuestra base de datos.

```
postgres=# create user espaciales with encrypted password 'espaciales';
```

- B. Crear una base de datos espacial llamada “espaciales”

```
postgres=# create database espaciales owner espaciales;
```

- C. Asignar al usuario `espaciales` como dueño del esquema `public` de la nueva base de datos.

```
postgres=# \c espaciales
You are now connected to database "espaciales" as user "postgres".
```

```
espaciales=# alter schema public owner to espaciales;
```

El comando `\c` permite cambiarse a otra base de datos estando en sesión como usuario `postgres`.

D. Agregar la extensión espacial a la base de datos espaciales:

```
espaciales=# CREATE EXTENSION postgis;  
espaciales=# CREATE EXTENSION postgis_topology;
```

C9: Incluir en el reporte la salida de estos 2 comandos (pequeño extracto).

E. Para acceder a nuestra nueva base de datos espacial empleando el usuario `espaciales`, ejecutar:

```
espaciales=# \q  
psql espaciales espaciales
```

F. Finalmente, para verificar que nuestra instalación es correcta, ejecutar las siguientes instrucciones, e **C10: Incluir la salida en el reporte.**

```
espaciales=> select postgis_lib_version();  
espaciales=> select now();
```

1.10. VISUALIZACIÓN DE LA BASE ESPACIAL EMPLEANDO PGADMIN

Para trabajar de forma gráfica la nueva base de datos, se hará uso de pgAdmin. Realizar las siguientes acciones para iniciar la aplicación.

A. Generalmente tanto en Ubuntu, Mint y Fedora, el proceso de instalación crea un acceso directo en alguno de los siguientes menús:

- Menu -> Programación -> pgAdmin4
- Menu -> Herramientas del sistema -> pgAdmin4
- En el campo de búsqueda de aplicaciones se puede buscar la aplicación pgAdmin4

B. Una vez encontrado el acceso directo, hacer clic para iniciar.

C. Aparecerá una pantalla similar a la siguiente.



D. Hacer clic derecho en el ícono Servers, seleccionar `create-> Server` para configurar el acceso a la base de datos proporcionando los datos como se muestra en la imagen (pestaña General y pestaña Connection). Presionar Save.

Create - Server

General Connection SSL Advanced

Name:

Server group:

Background: ☒

Foreground: ☒

Connect now?: ☒

Comments:

Create - Server

General Connection SSL Advanced

Host name/address:

Port:

Maintenance database:

Username:

Password:

Save password?: ☐

Role:

- E. Observar los esquemas y elementos de la nueva base de datos. Revisar que la configuración de la sentencia CREATE DATABASE coincida con la de la siguiente imagen. Para ello del lado derecho seleccionar la pestaña SQL.

pgAdmin 4 File Object Tools Help jorgerdc@gmail.com

Browser

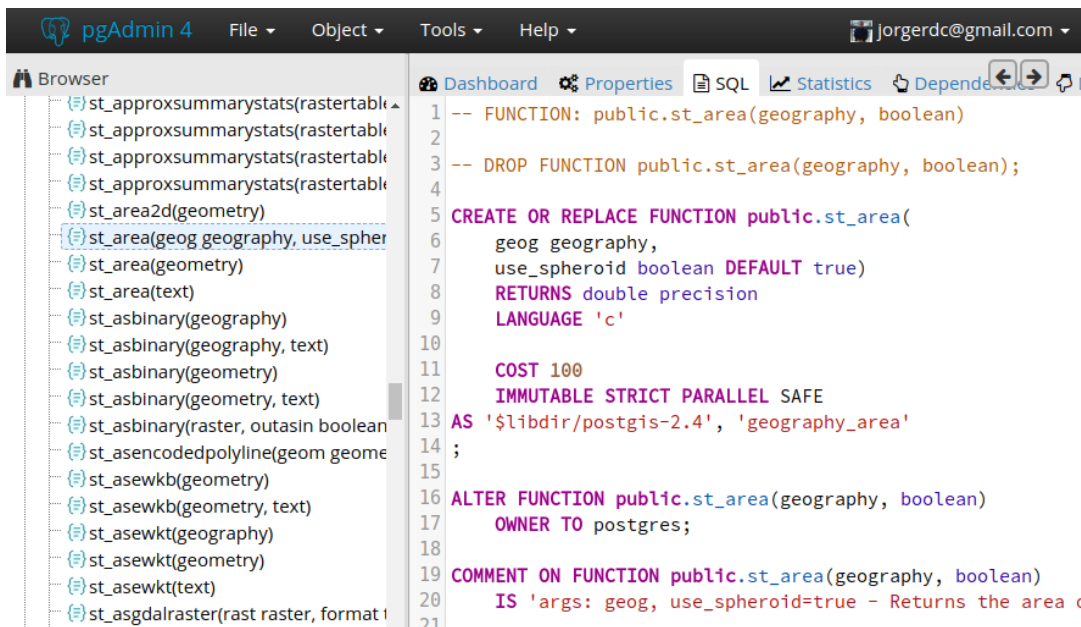
- Servers (2)
 - conexion-bde
 - Databases (2)
 - espaciales**
 - Casts
 - Catalogs
 - Event Triggers
 - Extensions
 - Foreign Data Wrappers
 - Languages
 - Schemas (2)
 - public
 - Collations
 - Domains
 - FTS Configurations
 - FTS Dictionaries
 - FTS Parsers
 - FTS Templates
 - Foreign Tables
 - Functions (1174)
 - Materialized Views
 - Sequences
 - Tables
 - Trigger Functions
 - Types
 - Views
 - topology

SQL

```

1 -- Database: espaciales
2
3 -- DROP DATABASE espaciales;
4
5 CREATE DATABASE espaciales
6 WITH
7     OWNER = espaciales
8     ENCODING = 'LATIN1'
9     LC_COLLATE = 'es_MX.iso88591'
10    LC_CTYPE = 'es_MX.iso88591'
11    TABLESPACE = pg_default
12    CONNECTION LIMIT = -1;
13
14 ALTER DATABASE espaciales
15     SET search_path TO "$user", public, topology;
  
```

- F. En postgresQL, por default se crea un schema llamado “public” que contiene todos los objetos tablas, vistas, y funciones, en especial las funciones vistas en clase que nos permiten trabajar con objetos espaciales. Observar la siguiente imagen, se ha seleccionado public -> functions -> st_area



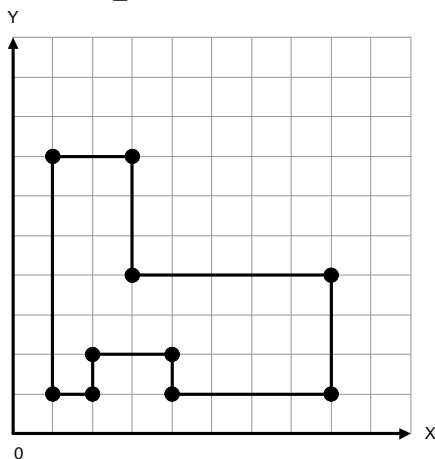
Observar la descripción del lado derecho, se muestra la forma en la que se define una función en nuestra base. La ruta `$libdir/postgis-x.x`, indica la librería que contiene la programación de la función.

C11: Incluir en el reporte una imagen similar a la anterior, pero seleccionando la función “`st_distance`”.

1.11. CREACIÓN DE TABLAS CON GEOMETRÍAS

Suponer que se desea guardar la información del siguiente objeto geográfico:

`Pais(país_id, nombre, capital, geometría): P1=(1, 'ITALIA', 'ROMA', geo)`, donde geo esta dada por la siguiente figura:



La representación en modo texto de esta figura es:

```
POLYGON((1 1,2 1,2 2,4 2,4 1,8 1,8 4,3 4,3 7,1 7,1 1))
```

Para realizar la inserción de este objeto en la base de datos, se deberá crear una tabla de forma normal y posteriormente agregar el ADT que en este caso es un POLYGON:

```

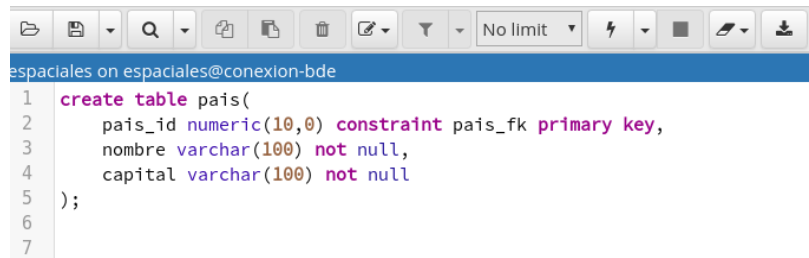
create table pais(
    pais_id numeric(10,0) constraint pais_fk primary key,
    nombre varchar(100) not null,
    capital varchar(100) not null
);

```

Para ejecutar sentencias SQL con pgAdmin:

A. Seleccionar del menú `Tools-> Query Tool`, escribir la sentencia SQL y ejecutarla.





```

1 create table pais(
2     pais_id numeric(10,0) constraint pais_fk primary key,
3     nombre varchar(100) not null,
4     capital varchar(100) not null
5 );
6
7

```

Posteriormente, agregamos el ADT empleando la función `addGeometryColumn`, la cual acepta los siguientes parámetros:

```

AddGeometryColumn (
    <schema_name>,
    <table_name>,
    <column_name>,
    <srid>,
    <type>,
    <dimension>
)

```

- Schema_name: en caso de que no se especifica, el valor del esquema se asigna a "public".
- Table_name: Nombre de la tabla a la que se le agregará el ADT.
- Column_name: Nombre de la columna que representa al ADT.
- SRID: Identificador del sistema de referencia asociado al ADT. Su valor corresponde con algún sistema registrado en la tabla `SPATIAL_REF_SYS`. Para efectos del curso, le asignaremos el valor 0 (sistema de referencia indefinido).
- Type: el tipo de ADT, corresponde con los nombres de los ADTs empleados para realizar su representación en formato WKT
- Dimension: La dimensión del sistema de referencia. Para efectos del curso es 2, es decir, un sistema de coordenadas X,Y (no confundir con la dimensión del objeto espacial).

```
select addGeometryColumn('public','pais','geo',0,'POLYGON',2);
```

Para realizar la inserción del registro nos apoyamos de la función `st_geomFromText`:

```

insert into pais(pais_id,nombre,capital,geo)
values(1,'ITALIA','ROMA',st_geomFromText(
'POLYGON((1 1,2 1,2 2,4 2,4 1,8 1,8 4,3 4,3 7,1 7,1 1))'
));

```

Para mostrar el registro creado empleamos la función `asText`

```
select pais_id,nombre,capital, st_asText(geo) as geo from pais;
```

Para verificar si la geometría que agregamos es válida, empleamos la función `st_isValid`

```
select pais_id,nombre,capital, st_asText(geo) as geo, st_isValid(geo) as valida from pais;
```

Ejecutar estas 2 instrucciones e **C12: Incluir la salida en el reporte.**

Finalmente (**C13 – C15**):

- Crear una sentencia SQL que calcule el área de la figura anterior, incluir sentencia y resultado.
- Crear una sentencia SQL que calcule el perímetro de la figura anterior, incluir sentencia y resultado.
- Dentro del esquema `public`, observar que existen una tabla llamada: `spatial_ref_sys`. Observar su estructura y los datos que contienen, ¿Cuál es su finalidad?

1.12. CONTENIDO DEL REPORTE

- Introducción
- Objetivo
- Desarrollo de la práctica:
 - C1: Pantalla con el nombre de usuario o indicación de instalación y usuario existente.
 - C2: Pantalla con extracto del proceso de compilación.

- C3: Pantalla de inicialización de cluster.
- C4: Pantalla contenido del directorio DATA
- C5: Significado del “2>&1 &”
- C6: Pantalla con la lista de procesos del usuario postgres.
- C7: Pantalla, versión, hora y fecha en `psql`
- C8: Pantalla de error de autenticación
- C9: Pantalla de creación de la extensión espacial
- C10: Pantalla Fecha del sistema y versión de PostGIS.
- C11: Pantalla descripción de la función `st_distance`.
- C12: Resultado de `ST_ISVALID`
- C13: Área del polígono
- C14: Perímetro del polígono
- C15: Significado y utilidad de la tabla `spatial_ref_sys`.
- Opcionalmente se pueden incluir algunas pantallas.
- Conclusiones, comentarios, recomendaciones.
- Bibliografía.