

LABORATORY NO.05 - Databases and Network Protocols



ELABORADO POR:

ESTEBAN AGUILERA CONTRERAS
JUAN DAVID RODRIGUEZ RODRIGUEZ

PROFESOR(ES):

JOHN PACHON

RECO
2025-1

OBJECTIVE

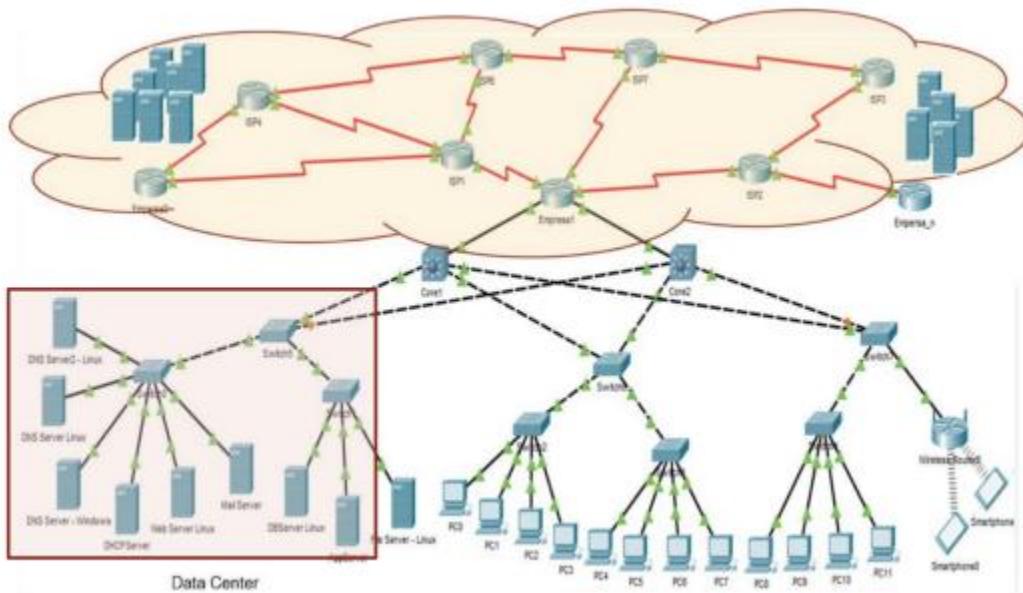
Continue learning the installation of basic software and the operation of network protocols.

TOOLS TO BE USED

- Computers
- Internet access
- Virtualization software
- Wireshark

CONTEXT

We are continuing to work within a company's infrastructure, which typically includes various IT services. This infrastructure features both wired and wireless user workstations and servers (both physical and virtualized), all connected through switches (Layer 2 and Layer 3), wireless equipment, and routers connecting them to the Internet. Cloud infrastructures are also common, provisioning resources as needed by the organization. The servers may host services such as web, DNS, email, databases, storage, and applications, among others.



In this part of the lab, we will focus on continuing to set up our servers

INTRODUCTION

En este laboratorio, se trabajará con herramientas de análisis de red y bases de datos para entender su funcionamiento e interacción dentro de un entorno empresarial.

Se analizarán los protocolos de red mediante la captura de tráfico con Wireshark y se instalarán distintos motores de bases de datos en servidores virtualizados, permitiendo evaluar diferencias entre soluciones locales y en la nube. Además, se explorará la importancia de los conceptos de escalabilidad, seguridad y disponibilidad en la gestión de bases de datos empresariales.

THEORETICAL FRAMEWORK

1. Protocolos de Red

- **DNS (Domain Name System):** Protocolo que permite la resolución de nombres de dominio en direcciones IP. Se analizarán los mensajes DNS para comprender su estructura y funcionamiento.
- **HTTP (Hypertext Transfer Protocol):** Protocolo utilizado para la transferencia de datos en la web. Se examinarán los encabezados HTTP y su papel en la comunicación entre clientes y servidores.
- **Ethernet:** Tecnología de red basada en tramas que permite la comunicación dentro de redes locales (LAN). Se estudiará el formato de los encabezados Ethernet en las comunicaciones web.

2. Bases de Datos y Motores de Bases de Datos

- **SQL Server:** DBMS desarrollado por Microsoft, diseñado para entornos Windows Server con alta integración en sistemas empresariales.
- **Azure SQL Database:** Servicio en la nube de Microsoft basado en SQL Server, que ofrece alta disponibilidad y escalabilidad sin necesidad de administrar hardware físico.

3. Computación en la Nube y Azure

- **Cloud Computing:** Modelo que permite el acceso a recursos de cómputo bajo demanda a través de internet. Microsoft Azure ofrece servicios en modalidades IaaS (Infraestructura como Servicio), PaaS (Plataforma como Servicio) y SaaS (Software como Servicio).
- **Regiones y Zonas de Disponibilidad:** Concepto clave en la infraestructura de la nube para garantizar la disponibilidad de los servicios mediante la distribución geográfica de recursos.
- **Escalabilidad Vertical vs. Horizontal:** La escalabilidad vertical implica mejorar los recursos de una máquina existente, mientras que la escalabilidad horizontal consiste en agregar más máquinas a la infraestructura.
- **Seguridad en la Nube:** Tecnologías como TLS (Transport Layer Security) garantizan la seguridad en la comunicación de datos en entornos de nube, comparado con bases de datos locales.

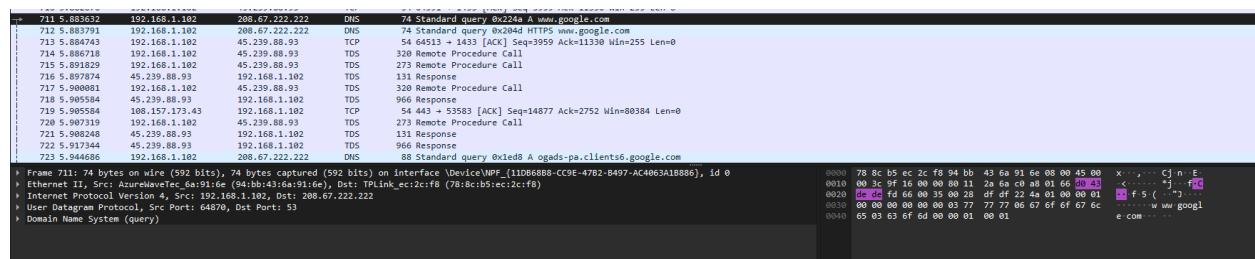
4. Configuraciones Avanzadas de Base de Datos

- **Arranque automático del DBMS:** Configuración que permite que los servicios de base de datos se inicien automáticamente al arrancar el sistema operativo.
- **Conexión Remota a Bases de Datos:** Uso de herramientas como DBeaver para acceder y gestionar bases de datos alojadas en servidores remotos.

REVIEW OF NETWORK PROTOCOLS

1. Review of DNS Messages

Using Wireshark, make a query to the webpage <http://www.google.com/>, filter DNS messages, and examine the name resolution. Identify the fields and explain each part.



1) Información de la Capa de Enlace de Datos (Ethernet II)

- Src (Origen): AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e) → Es la dirección de mi dispositivo que envió la consulta
- Dst (Destino): TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8) → Es la dirección MAC del router o dispositivo de red que recibe la consulta.

2) Información de la Capa de Red (IP - Internet Protocol)

- Versión: IPv4
- Origen: 192.168.1.102 → Es la dirección IP de mi dispositivo que hizo la consulta DNS.
- Destino: 208.67.222.222 → Es la dirección IP del servidor DNS de OpenDNS.
- Protocolo: UDP

3) Información de la Capa de Transporte (UDP - User Datagram Protocol)

- Src Port (Puerto de Origen): 64870 → Puerto asignado dinámicamente por la computadora solicitante.
- Dst Port (Puerto de Destino): 53 → El puerto 53 es el estándar para el servicio DNS.

4) Información de la Capa de Aplicación (DNS - Domain Name System)

- Tipo: Query (Consulta)
- ID de Transacción: 0x224a → Número único que identifica la consulta y su respuesta correspondiente.
- Nombre de Dominio Consultado: www.google.com
- Tipo de Consulta: A → Significa que está pidiendo la dirección IPv4 del dominio.

2. Review of HTTP Messages

Using Wireshark, make a query to the webpage

<http://profesores.is.escuelaing.edu.co/~csantiago/>, filter the messages, and examine the HTTP message header. Identify the fields and explain each part.

The screenshot shows a Wireshark capture window. At the top, there's a green bar labeled 'http'. Below it is a table with columns: No., Time, Source, Destination, Protocol, Length, and Info. The table lists several network frames, mostly HTTP requests from source 45.239.88.86 to destination 192.168.1.102. Frame 2142 is highlighted in yellow and expanded. It shows the raw hex and ASCII data for an unencrypted HTTP request. The ASCII dump includes headers like 'HTTP/1.1 400 Bad Request' and 'Content-Type: text/html; charset=iso-8859-1'. The detailed info pane at the bottom provides a breakdown of the frame's structure, including layers like Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and Hypertext Transfer Protocol.

| No. | Time | Source | Destination | Protocol | Length | Info |
|------|-----------|---------------|---------------|----------|--------|--------------------------------------|
| 2142 | 20.719786 | 45.239.88.86 | 192.168.1.102 | HTTP | 470 | HTTP/1.1 400 Bad Request (text/html) |
| 2145 | 20.720120 | 192.168.1.102 | 45.239.88.86 | HTTP | 61 | Continuation |
| 2160 | 20.753584 | 45.239.88.86 | 192.168.1.102 | HTTP | 470 | HTTP/1.1 400 Bad Request (text/html) |
| 2163 | 20.753823 | 192.168.1.102 | 45.239.88.86 | HTTP | 61 | Continuation |
| 2770 | 26.115556 | 192.168.1.102 | 45.239.88.86 | HTTP | 511 | GET /~csantiago/ HTTP/1.1 |
| 2772 | 26.124230 | 45.239.88.86 | 192.168.1.102 | HTTP | 492 | HTTP/1.1 200 OK (text/html) |
| 2778 | 26.186703 | 192.168.1.102 | 45.239.88.86 | HTTP | 492 | GET /~csantiago/foto.jpg HTTP/1.1 |
| 2806 | 26.213875 | 45.239.88.86 | 192.168.1.102 | HTTP | 1451 | HTTP/1.1 200 OK (PNG) |

```
Frame 2142: 470 bytes on wire (3760 bits), 470 bytes captured (3760 bits) on interface \Device\NPF_{11DB68B8-CC9E-47B2-8497-AC4063A1B886}, id 0
Ethernet II, Src: TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8), Dst: AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e)
Internet Protocol Version 4, Src: 45.239.88.86, Dst: 192.168.1.102
Transmission Control Protocol, Src Port: 443, Dst Port: 55514, Seq: 1, Ack: 1838, Len: 416
Hypertext Transfer Protocol
[Expert Info (Warning/Security): Unencrypted HTTP protocol detected over encrypted port, could indicate a dangerous misconfiguration.]
HTTP/1.1 400 Bad Request\r\n
Date: Fri, 21 Mar 2025 23:31:04 GMT\r\n
Server: Apache/2.4.53 (Unix) PHP/8.1.4\r\n
Content-Length: 226\r\n
Connection: close\r\n
Content-Type: text/html; charset=iso-8859-1\r\n
\r\n
File Data: 226 bytes
Line-based text data: text/html (8 lines)
```

Primer paquete

- Muestra un paquete de red con 470 bytes de datos.
- Capa de Enlace (Ethernet II): Dirección MAC de origen (78:8c:b5:ec:2c:f8) y destino (94:bb:43:6a:91:6e).
- Capa de Red (IPV4): Origen 45.239.88.86, destino 192.168.1.102.
- Capa de Transporte (TCP):
 - Puerto de origen 443 (HTTPS, indica tráfico cifrado).
 - Puerto de destino 55514.
- Capa de Aplicación (HTTP):
 - Indica que el paquete pertenece a una comunicación HTTP segura.

```

> Frame 2772: 492 bytes on wire (3936 bits), 492 bytes captured (3936 bits) on interface \Device\NPF_{11DB68B8-CC9E-47B2-B497-AC4063A1B886}, id 0
> Ethernet II, Src: TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8), Dst: AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e)
> Internet Protocol Version 4, Src: 45.239.88.86, Dst: 192.168.1.102
> Transmission Control Protocol, Src Port: 80, Dst Port: 55517, Seq: 1, Ack: 458, Len: 438
> Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Fri, 21 Mar 2025 23:31:09 GMT\r\n\r\n
    Server: Apache/2.4.53 (Unix) PHP/8.1.4\r\n
    Last-Modified: Thu, 01 Jul 2021 02:57:02 GMT\r\n\r\n
    ETag: "92-5c606fe4b3b80"\r\n\r\n
    Accept-Ranges: bytes\r\n\r\n
    Content-Length: 146\r\n\r\n
    Keep-Alive: timeout=5, max=100\r\n\r\n
    Connection: Keep-Alive\r\n\r\n
    Content-Type: text/html\r\n\r\n
    \r\n\r\n
    [Request in frame: 2770]
    [Time since request: 0.008674000 seconds]
    [Request URI: /~csantiago/]
    [Full request URL: http://profesores.is.escuelaing.edu.co/~csantiago/]
    File Data: 146 bytes
> Line-based text data: text/html (9 lines)

```

Segunda paquete

- Similar a la primera, pero aquí el tráfico usa HTTP en el puerto 80 en lugar de HTTPS.
- TCP:
 - Puerto de origen 80 (HTTP, tráfico sin cifrar).
 - Puerto de destino 55517.
- Capa de Aplicación (HTTP):
 - Se está transmitiendo una página web en texto plano.
- Retorna un mensaje 200 un OK

```

> Frame 2806: 1451 bytes on wire (11608 bits), 1451 bytes captured (11608 bits) on interface \Device\NPF_{11DB68B8-CC9E-47B2-B497-AC4063A1B886}, id 0
> Ethernet II, Src: TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8), Dst: AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e)
> Internet Protocol Version 4, Src: 45.239.88.86, Dst: 192.168.1.102
> Transmission Control Protocol, Src Port: 80, Dst Port: 55517, Seq: 23479, Ack: 896, Len: 1397
> [17 Reassembled TCP Segments (24437 bytes): #2782(1440), #2783(1440), #2786(1440), #2785(1440), #2787(1440), #2788(1440), #2789(1440), #2791(1440), #2792(1
> Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Fri, 21 Mar 2025 23:31:09 GMT\r\n\r\n
    Server: Apache/2.4.53 (Unix) PHP/8.1.4\r\n
    Last-Modified: Thu, 01 Jul 2021 02:58:24 GMT\r\n\r\n
    ETag: "5e4d-5c607032e7400"\r\n\r\n
    Accept-Ranges: bytes\r\n\r\n
    Content-Length: 24141\r\n\r\n
    Keep-Alive: timeout=5, max=99\r\n\r\n
    Connection: Keep-Alive\r\n\r\n
    Content-Type: image/jpeg\r\n\r\n
    \r\n\r\n
    [Request in frame: 2778]
    [Time since request: 0.027172000 seconds]
    [Request URI: /~csantiago/foto.jpg]
    [Full request URL: http://profesores.is.escuelaing.edu.co/~csantiago/foto.jpg]
    File Data: 24141 bytes
  > [Expert Info (Note/Malformed): HTTP body subdissector failed, trying heuristic subdissector]
> Portable Network Graphics

```

Tercer paquete

- Muestra la respuesta a una solicitud HTTP.
- HTTP/1.1 200 OK: Indica que el servidor respondió correctamente.
- Cabeceras HTTP:
 - Servidor: Apache/2.4.53 con PHP.
 - Fecha de respuesta: Viernes, 21 de marzo de 2025.
 - Contenido: Imagen JPEG de 24,411 bytes.

- URL completa de la imagen:
<http://profesores.is.escuelaing.edu.co/wcsantiago/foto.jpg>.

3. Review of Ethernet Frames

Using Wireshark, make a query to the webpage

<http://profesores.is.escuelaing.edu.co/~csantiago/>, filter the GET messages and examine the Ethernet frame header. Identify the fields and explain each part.

| No. | Time | Source | Destination | Protocol | Length Info |
|------|-----------|---------------|---------------|----------|--|
| 2142 | 20.719786 | 45.239.88.86 | 192.168.1.102 | HTTP | 470 HTTP/1.1 400 Bad Request (text/html) |
| 2145 | 20.720120 | 192.168.1.102 | 45.239.88.86 | HTTP | 61 Continuation |
| 2160 | 20.753584 | 45.239.88.86 | 192.168.1.102 | HTTP | 470 HTTP/1.1 400 Bad Request (text/html) |
| 2163 | 20.753823 | 192.168.1.102 | 45.239.88.86 | HTTP | 61 Continuation |
| 2770 | 26.115556 | 192.168.1.102 | 45.239.88.86 | HTTP | 511 GET /~csantiago/ HTTP/1.1 |
| 2772 | 26.124230 | 45.239.88.86 | 192.168.1.102 | HTTP | 492 HTTP/1.1 200 OK (text/html) |
| 2778 | 26.186703 | 192.168.1.102 | 45.239.88.86 | HTTP | 492 GET /~csantiago/foto.jpg HTTP/1.1 |
| 2806 | 26.213875 | 45.239.88.86 | 192.168.1.102 | HTTP | 1451 HTTP/1.1 200 OK (PNG) |


```

> Frame 2770: 511 bytes on wire (4088 bits), 511 bytes captured (4088 bits) on interface \Device\NPF_{11DB68B8-CC9E-47B2-B497-AC4063A1B886}, id 0
> Ethernet II, Src: AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e), Dst: TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8)
> Internet Protocol Version 4, Src: 192.168.1.102, Dst: 45.239.88.86
> Transmission Control Protocol, Src Port: 55517, Dst Port: 80, Seq: 1, Ack: 1, Len: 457
  Hypertext Transfer Protocol
    > GET /~csantiago/ HTTP/1.1\r\n
      Host: profesores.is.escuelaing.edu.co\r\n
      Connection: keep-alive\r\n
      Cache-Control: max-age=0\r\n
      Upgrade-Insecure-Requests: 1\r\n
      User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36\r\n
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8\r\n
      Sec-GPC: 1\r\n
      Accept-Language: es-419,es;q=0.6\r\n
      Accept-Encoding: gzip, deflate\r\n
    \r\n
    [Response in frame: 2772]
    [Full request URI: http://profesores.is.escuelaing.edu.co/~csantiago/]
  
```

Encabezado de Ethernet (Ethernet II):

- MAC de origen (Src): AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e).
- MAC de destino (Dst): TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8)

Encabezado de la capa de red (IP - Protocolo Internet v4):

- Dirección IP de origen (Src): 192.168.1.102
- Dirección IP de destino (Dst): 45.239.88.86 < Dirección IP del servidor que aloja la página profesores.is.escuelaing.edu.co.

Encabezado de la capa de transporte (TCP - Protocolo de Control de Transmisión):

- Puerto de origen: 55517 < Puerto asignado temporalmente en el dispositivo cliente para esta conexión.
- Puerto de destino: 80 < Puerto estándar de HTTP en el servidor.
- Número de secuencia (Seq): 1 < Número inicial de secuencia en la transmisión.

- Número de acuse de recibo (Ack): 1 < Indica el número de confirmación de paquetes.

Encabezado HTTP (Protocolo de Transferencia de Hipertexto):

- Método: GET
 - Sigue el recurso /~csantiago/.
- Host: profesores.is.escuelaing.edu.co
 - Servidor web al que se está accediendo.

```

> Frame 2778: 492 bytes on wire (3936 bits), 492 bytes captured (3936 bits) on interface \Device\NPF_{11DB6888-CC9E-47B2-B497-AC4063A18886}, id 0
> Ethernet II, Src: AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e), Dst: TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8)
> Internet Protocol Version 4, Src: 192.168.1.102, Dst: 45.239.88.86
> Transmission Control Protocol, Src Port: 55517, Dst Port: 80, Seq: 458, Ack: 439, Len: 438
- Hypertext Transfer Protocol
  > GET /~csantiago/foto.jpg HTTP/1.1\r\n
    Host: profesores.is.escuelaing.edu.co\r\n
    Connection: keep-alive\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36\r\n
    Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8\r\n
    Sec-GPC: 1\r\n
    Accept-Language: es-419,es;q=0.6\r\n
    Referer: http://profesores.is.escuelaing.edu.co/~csantiago/\r\n
    Accept-Encoding: gzip, deflate\r\n
  \r\n
[Response in frame: 2806]
[Full request URI: http://profesores.is.escuelaing.edu.co/~csantiago/foto.jpg]
```

Encabezado de Ethernet (Ethernet II):

- MAC de origen (Src): AzureWaveTec_6a:91:6e (94:bb:43:6a:91:6e).
- MAC de destino (Dst): TPLink_ec:2c:f8 (78:8c:b5:ec:2c:f8).

Encabezado de la capa de red (IP - Protocolo Internet v4):

- Dirección IP de origen (Src): 192.168.1.102.
- Dirección IP de destino (Dst): 45.239.88.86 < Dirección IP del servidor que aloja la página profesores.is.escuelaing.edu.co.

Encabezado de la capa de transporte (TCP - Protocolo de Control de Transmisión):

- Puerto de origen: 55517 < Puerto asignado temporalmente en el dispositivo cliente para esta conexión.
- Puerto de destino: 80 < Puerto estándar de HTTP en el servidor.
- Número de secuencia (Seq): 458 < Número de secuencia de la transmisión.
- Número de acuse de recibo (Ack): 439 < Indica el número de confirmación de paquetes.

Encabezado HTTP (Protocolo de Transferencia de Hipertexto):

- Método: GET
 - Sigue el recurso /~csantiago/foto.jpg.
- Host: profesores.is.escuelaing.edu.co.

- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36.
- Referer: <http://profesores.is.escuelaing.edu.co/~csantiago/>
- Full request URI: <http://profesores.is.escuelaing.edu.co/~csantiago/foto.jpg>.

BASE SOFTWARE INSTALLATION

1. PostgreSQL - Linux Slackware

- Install the PostgreSQL DBMS on a virtual machine running Linux Slackware.
 - Instalamos PostgreSQL con los siguientes comandos

```
wget https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz
tar -xvzf postgresql.tar.gz
cd postgresql
sudo ./postgresql.SlackBuild
sudo installpkg /tmp/postgresql-*.tgz
```

Donde:

- wget <https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz>
 - Descarga el paquete SlackBuild de PostgreSQL.
- tar -xvzf postgresql.tar.gz
 - Extrae el contenido del archivo descargado.
- cd postgresql
 - Entra en la carpeta extraída.
- sudo ./postgresql.SlackBuild
 - Ejecuta el script de compilación y empaquetado de PostgreSQL.
- sudo installpkg /tmp/postgresql-*.tgz
 - Instala el paquete compilado en el sistema.

- Verificamos que este instalado y en funcionamiento

```
postgres@juanito:~$ ps aux | grep postgres
root      2297  0.0  0.4  5980  3696  tty1      S    20:17  0:00 su - postgres
postgres   2301  0.0  0.5  7044  3764  tty1      S    20:17  0:00 -su
root      17909  0.0  0.5  5980  3780  tty1      S    20:36  0:00 su - postgres
postgres   17915  0.0  0.5  7048  3824  tty1      S    20:36  0:00 -su
postgres   17949  0.0  0.3  4720  2388  tty1     R+   20:41  0:00 ps aux
postgres   17950  0.0  0.2  3864  2056  tty1     S*   20:41  0:00 grep postgres
postgres@juanito:~$
```

- Creamos un usuario para acceder a la base de datos y le damos permisos sobre esta en el archivo visudo

```
root@juanito:~# visudo_
```

```

## Uncomment to enable logging of a command's output, except for
## sudoreplay and reboot. Use sudoreplay to play back logged sessions.
# Defaults log_output
# Defaults!/usr/bin/sudoreplay !log_output
# Defaults!/usr/local/bin/sudoreplay !log_output
# Defaults!REBOOT !log_output

##
## Runas alias specification
##

##
## User privilege specification
##
root ALL=(ALL:ALL) ALL

## Uncomment to allow members of group wheel to execute any command
# %wheel ALL=(ALL:ALL) ALL

## Same thing without a password
# %wheel ALL=(ALL:ALL) NOPASSWD: ALL

## Uncomment to allow members of group sudo to execute any command
# %sudo ALL=(ALL:ALL) ALL

## Uncomment to allow any user to run sudo if they know the password
## of the user they are running the command as (root by default).
# Defaults targetpw # Ask for the password of the target user
# ALL ALL=(ALL:ALL) ALL # WARNING: only use this together with 'Defaults targetpw'

## Read drop-in files from /etc/sudoers.d
@includedir /etc/sudoers.d

postgres ALL=(ALL) NOPASSWD: /bin/systemctl start postgresql, /bin/systemctl stop postgresql, /bin/systemctl restart postgresql
/etc/sudoers.tmp: 98 lines, 3277 characters

```

- Entramos al usuario postgres

```
$ su -postgres
```

- Inicializamos la base de datos

```

postgres@juanito:~$ initdb -D /var/lib/pgsql/data
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locales
    COLLATE: C
    CTYPE: en_US.UTF-8
    MESSAGES: en_US.UTF-8
    MONETARY: en_US.UTF-8
    NUMERIC: en_US.UTF-8
    TIME: en_US.UTF-8
The default database encoding has accordingly been set to "UTF8".
The default text search configuration will be set to "english".

Data page checksums are disabled.

creating directory /var/lib/pgsql/data ... ok
creating subdirectories ... ok
selecting dynamic shared memory implementation ... posix
selecting default max_connections ... 100
selecting default shared_buffers ... 128MB
selecting default time zone ... America/Bogota
creating configuration files ... ok
running bootstrap script ... ok
performing post-bootstrap initialization ... ok
syncing data to disk ... ok

initdb: warning: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the option -A, or
--auth-local and --auth-host, the next time you run initdb.

Success. You can now start the database server using:

    pg_ctl -D /var/lib/pgsql/data -l logfile start

```

- Iniciamos el servicio de pgsql

```

postgres@juanito:~$ pg_ctl -D /var/lib/pgsql/data start
waiting for server to start.....2025-03-18 21:33:16.427 -05 [18339] LOG:  starting PostgreSQL 14.17
on i586-slackware-linux-gnu, compiled by gcc (GCC) 11.2.0, 32-bit
2025-03-18 21:33:16.716 -05 [18339] LOG:  listening on IPv6 address "::1", port 5432
2025-03-18 21:33:16.717 -05 [18339] LOG:  listening on IPv4 address "127.0.0.1", port 5432
2025-03-18 21:33:16.760 -05 [18339] LOG:  listening on Unix socket "/tmp/.PGSQL.5432"
2025-03-18 21:33:16.944 -05 [18340] LOG:  database system was shut down at 2025-03-18 20:36:59 -05
.2025-03-18 21:33:17.565 -05 [18339] LOG:  database system is ready to accept connections
done
server started
postgres@juanito:~$ ss

```

- Ingresamos al servicio mediante el comando:

```

postgres@juanito:~$ psql
psql (14.17)
Type "help" for help.

```

- Create a user for each group member. Use the students' names as the username.
 - Creamos los dos usuarios
 - Esteban
 - Juan

```
postgres=# CREATE USER estebana with PASSWORD 'root123';
CREATE ROLE
postgres=# CREATE USER juanR with PASSWORD 'root123';
CREATE ROLE
postgres=# _
```

- Miramos la lista de usuarios con el siguiente comando

| List of roles | | |
|---------------|--|-------------|
| Role name | Attributes | I Member of |
| estebana | | Ø |
| juanr | | Ø |
| postgres | Superuser, Create role, Create DB, Replication, Bypass RLS | Ø |

- Create a database to store information about tourist sites in Colombia you wish to visit. The database must have at least 3 tables. Each student should have access only to their own database.
 - Creamos la base de datos para guardar los datos de turismo

```
postgres=# CREATE DATABASE turismo;
CREATE DATABASE
```

- Cambiamos el owner de la base de datos con ALTER TABLE

```
postgres=# ALTER DATABASE turismo OWNER TO estebana;
ALTER DATABASE
postgres=# \l
```

- Nos conectamos a la base de datos desde nuestro usuario

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

- Empezamos a crear nuestra estructura de la base de datos
 - Sitios

```
turismo=# CREATE TABLE sitios (
  id SERIAL PRIMARY KEY,
  nombre VARCHAR(50) NOT NULL,
  ubicacion VARCHAR(50) NOT NULL,
  descripcion VARCHAR(255)
);
CREATE TABLE
turismo=#
```

- Actividades

```
turismo=# CREATE TABLE actividades (
  id SERIAL PRIMARY KEY,
  sitioId INT REFERENCES sitios(id),
  nombre VARCHAR(50) NOT NULL,
  tipo VARCHAR(50)
);
CREATE TABLE
```

- Opiniones

```
turismo=# CREATE TABLE opiniones (
  id SERIAL PRIMARY KEY,
  sitioId INT REFERENCES sitios(id),
  autor VARCHAR(50) NOT NULL,
  comentario VARCHAR (255)
);
CREATE TABLE
```

- Insert data into the databases

- Insertando datos en turismo

```
turismo=# INSERT INTO sitios (nombre, ubicacion, descripcion) VALUES
turismo-# ('sitio 1', 'ubicacion 1', 'descripcion 1'),
turismo-# ('sitio 2', 'ubicacion 2', 'descripcion 2'),
turismo-# ('sitio 3', 'ubicacion 3', 'descripcion 3');
INSERT 0 3
turismo=#

```

- Insertando datos en actividades

```
turismo=# INSERT INTO actividades (sitioId, nombre, tipo ) VALUES
(1, 'nombre 1', 'tipo 1'),
(2, 'nombre 2', 'tipo 2'),
(3, 'nombre 3', 'tipo 3');
INSERT 0 3
```

- Insertando datos en opiniones

```
turismo=# INSERT INTO opiniones (sitioId, autor, comentario ) VALUES
(1, 'autor 1', 'comentario 1'),
(2, 'autor 2', 'comentario 2'),
(3, 'autor 3', 'comentario 3');
INSERT 0 3
```

- Probamos que los datos hayan sido ingresados correctamente con select

```
INSERT 0 3
turismo=# select * FROM sitios;
 id | nombre | ubicacion | descripcion
-----+-----+-----+
 1 | sitio 1 | ubicacion 1 | descripcion 1
 2 | sitio 2 | ubicacion 2 | descripcion 2
 3 | sitio 3 | ubicacion 3 | descripcion 3
(3 rows)

turismo=# select * FROM actividades;
 id | sitioId | nombre | tipo
-----+-----+-----+
 1 |       1 | nombre 1 | tipo 1
 2 |       2 | nombre 2 | tipo 2
 3 |       3 | nombre 3 | tipo 3
(3 rows)

turismo=# select * FROM opiniones;
 id | sitioId | autor | comentario
-----+-----+-----+
 1 |       1 | autor 1 | comentario 1
 2 |       2 | autor 2 | comentario 2
 3 |       3 | autor 3 | comentario 3
(3 rows)

turismo=#

```

- Repetimos el mismo proceso con el usuario Juan y conservamos la misma estructura.
- Miramos que las bases de datos esten creadas con el comando \dt

| LIST OF DATABASES | | | | | | | |
|-------------------|----------|----------|---------|-------------|-----------------------|------------|---|
| Name | Owner | Encoding | Collate | Ctype | Access | privileges | |
| postgres | postgres | UTF8 | C | en_US.UTF-8 | =c/postgres | | + |
| template0 | postgres | UTF8 | C | en_US.UTF-8 | postgres=CTc/postgres | | |
| template1 | postgres | UTF8 | C | en_US.UTF-8 | =c/postgres | | + |
| turismo | estebana | UTF8 | C | en_US.UTF-8 | postgres=CTc/postgres | | |
| turismodos | juanr | UTF8 | C | en_US.UTF-8 | =Tc/juanr | | + |
| | | | | | estebana=CTc/estebana | | |
| | | | | | juanr=CTc/juanr | | |
| (5 rows) | | | | | | | |

- Bases de datos finales
 - estebanA

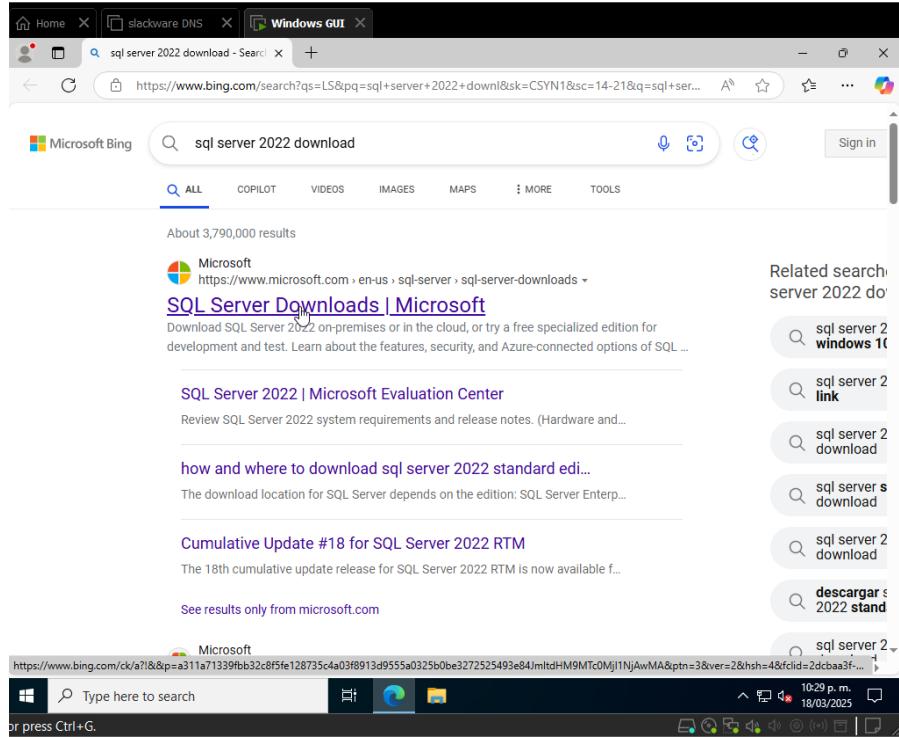
```
postgres=# \c turismo estebana
You are now connected to database "turismo" as user "estebana".
turismo=> \dt
      List of relations
 Schema |   Name    | Type | Owner
-----+-----+-----+
 public | actividades | table | estebana
 public | opiniones  | table | estebana
 public | sitios     | table | estebana
(3 rows)
```

- JuanR

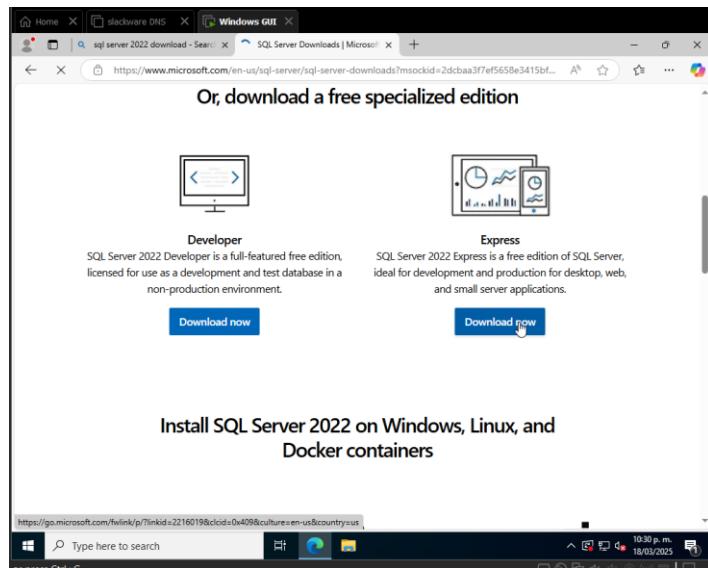
```
postgres=# \c turismodos juanr
You are now connected to database "turismodos" as user "juanr".
turismodos=> \dt
      List of relations
 Schema |   Name    | Type | Owner
-----+-----+-----+
 public | actividades | table | juanr
 public | opiniones  | table | juanr
 public | sitios     | table | juanr
(3 rows)
```

SQL SERVER - WINDOWS SERVER

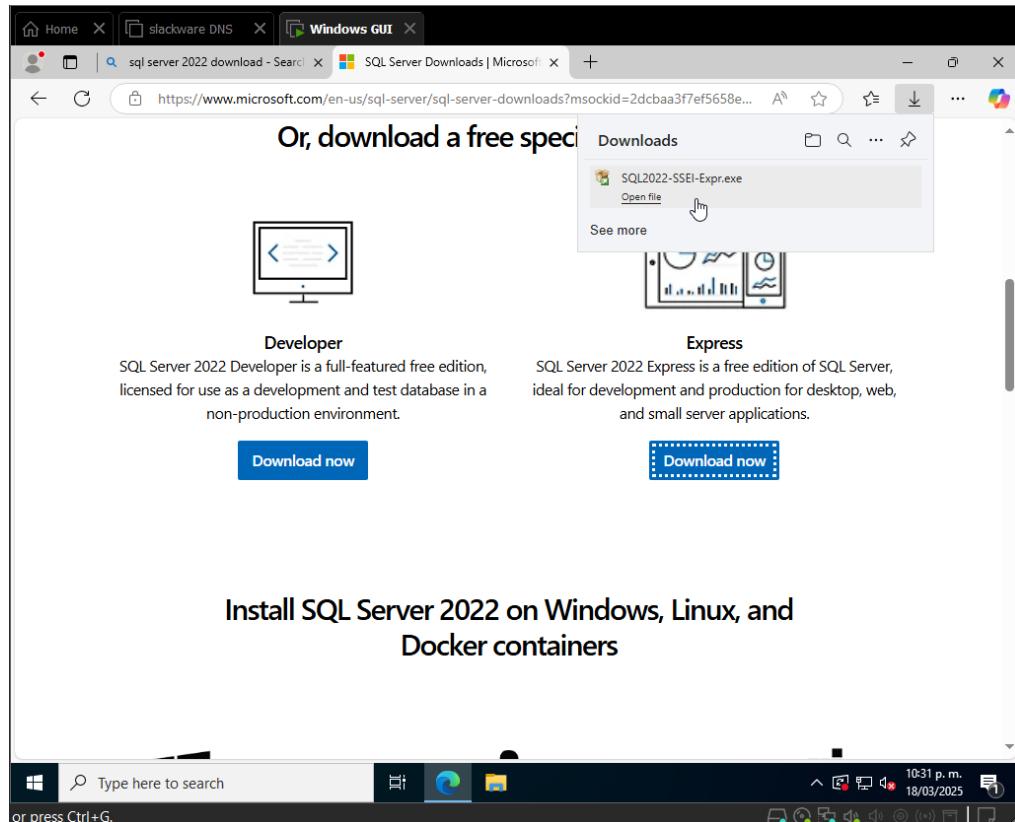
- Install the SQL Server DBMS on a virtual machine running Windows Server.
 - Buscamos SQL Server 2022 en el navegador y abrimos el primer link



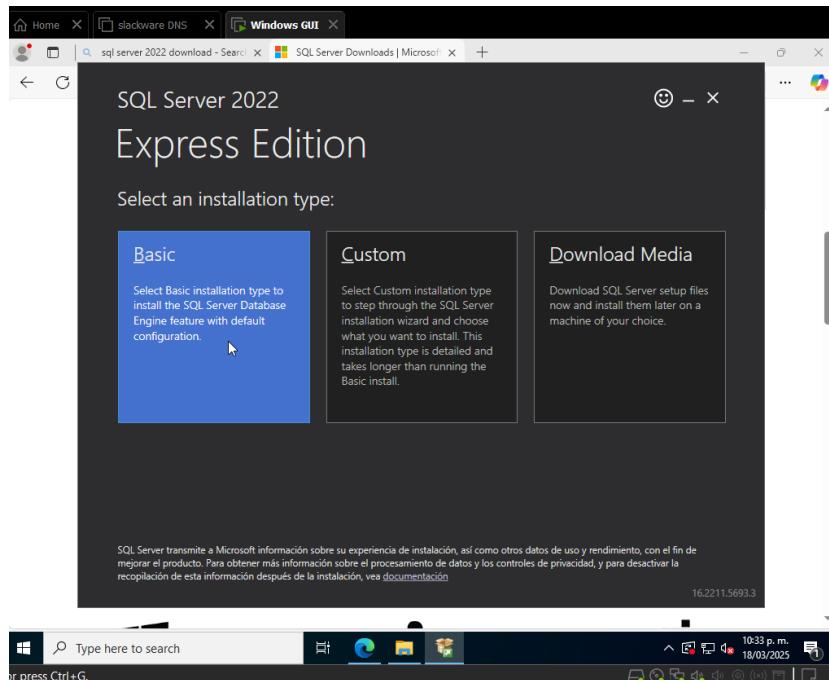
- Buscamos la opción express y damos click



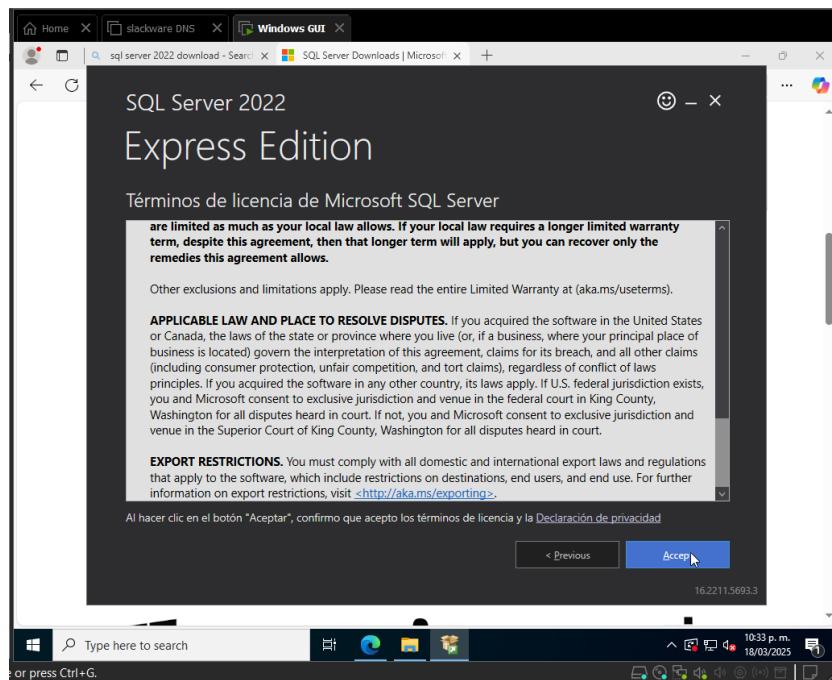
- Se generará una descarga. Una vez descargado el archivo, damos click en Open File



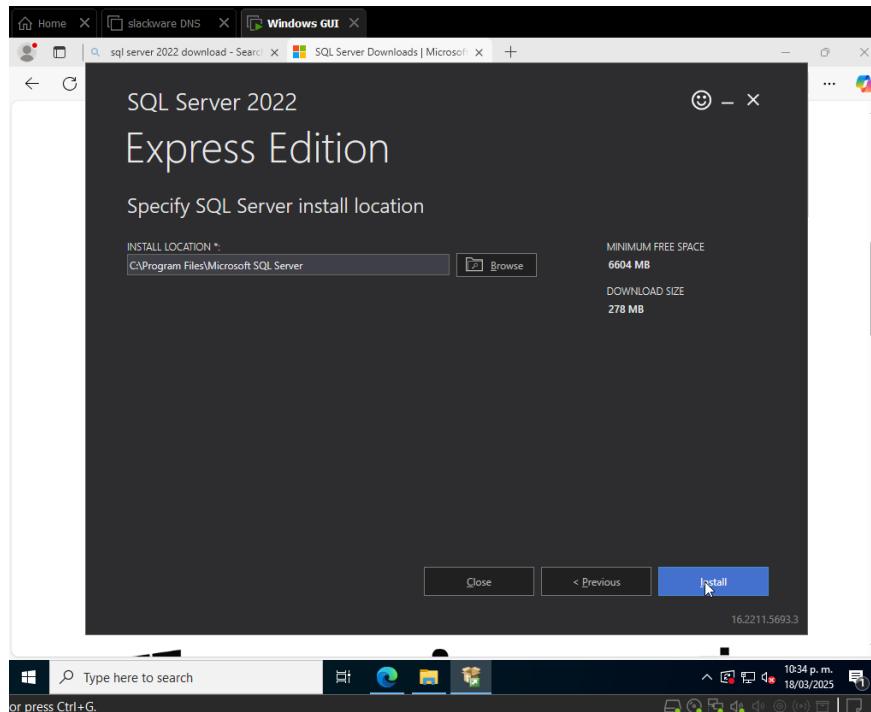
- Seleccionamos la opción basic



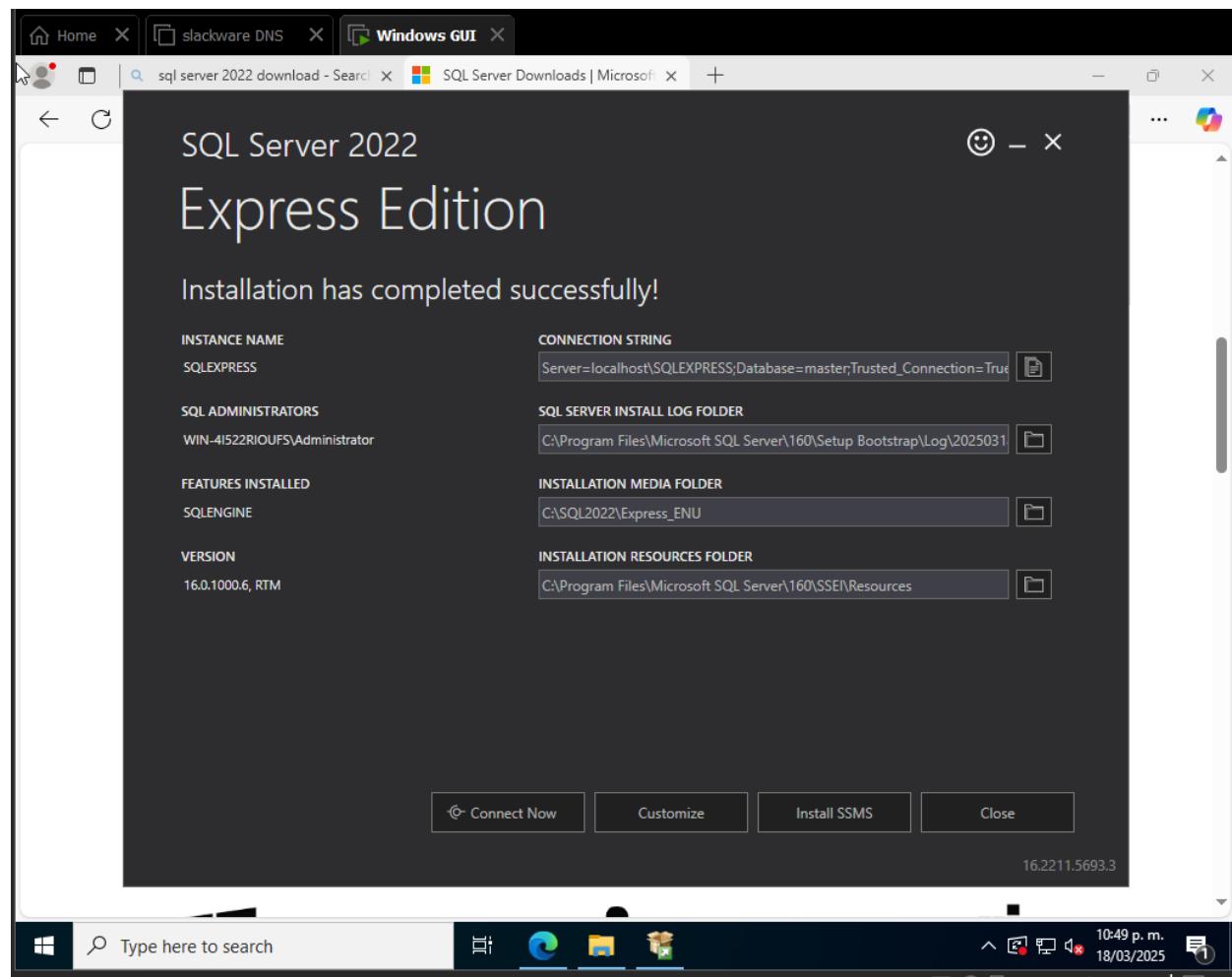
- Aceptamos los terminos y condiciones



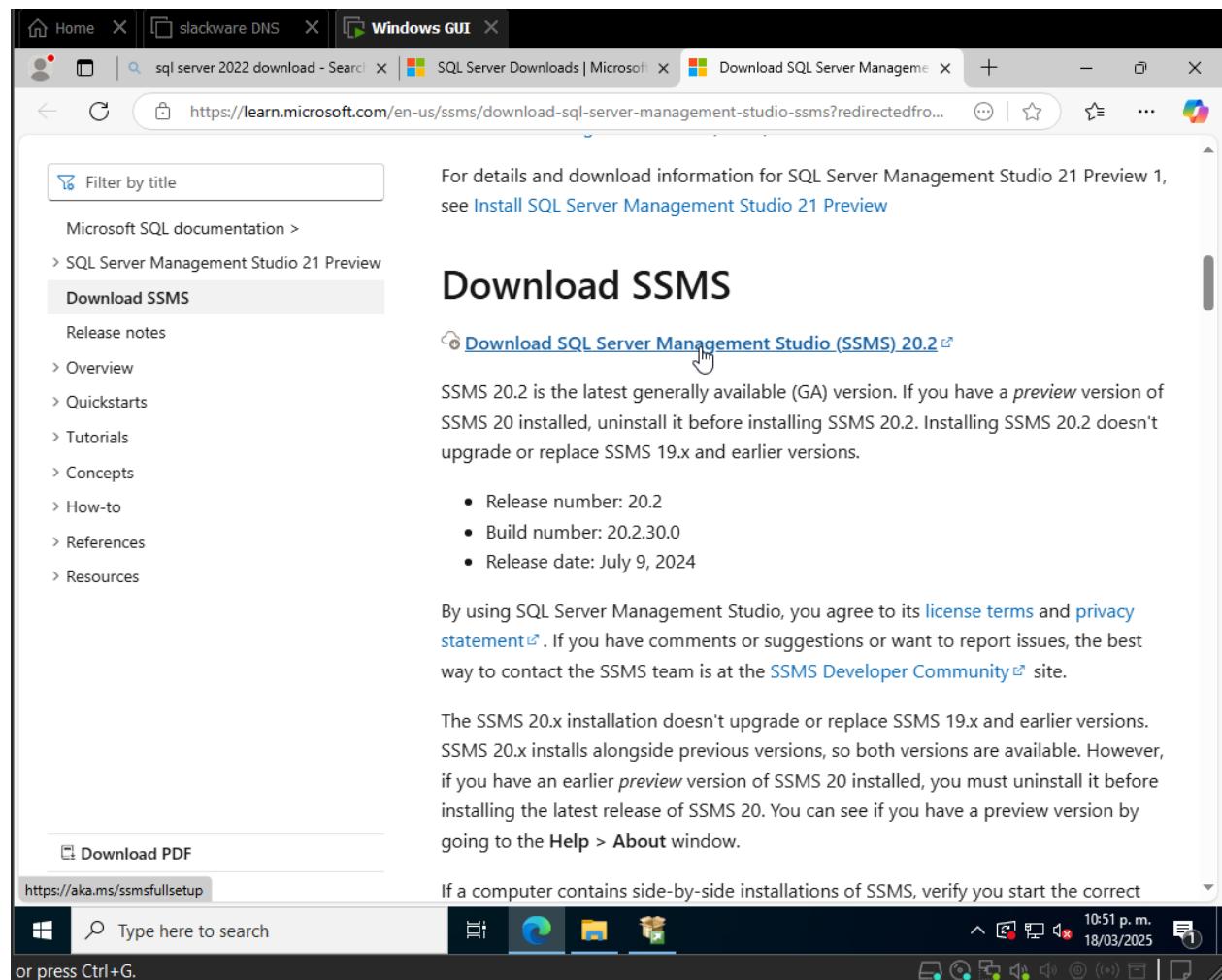
- Damos click en Install



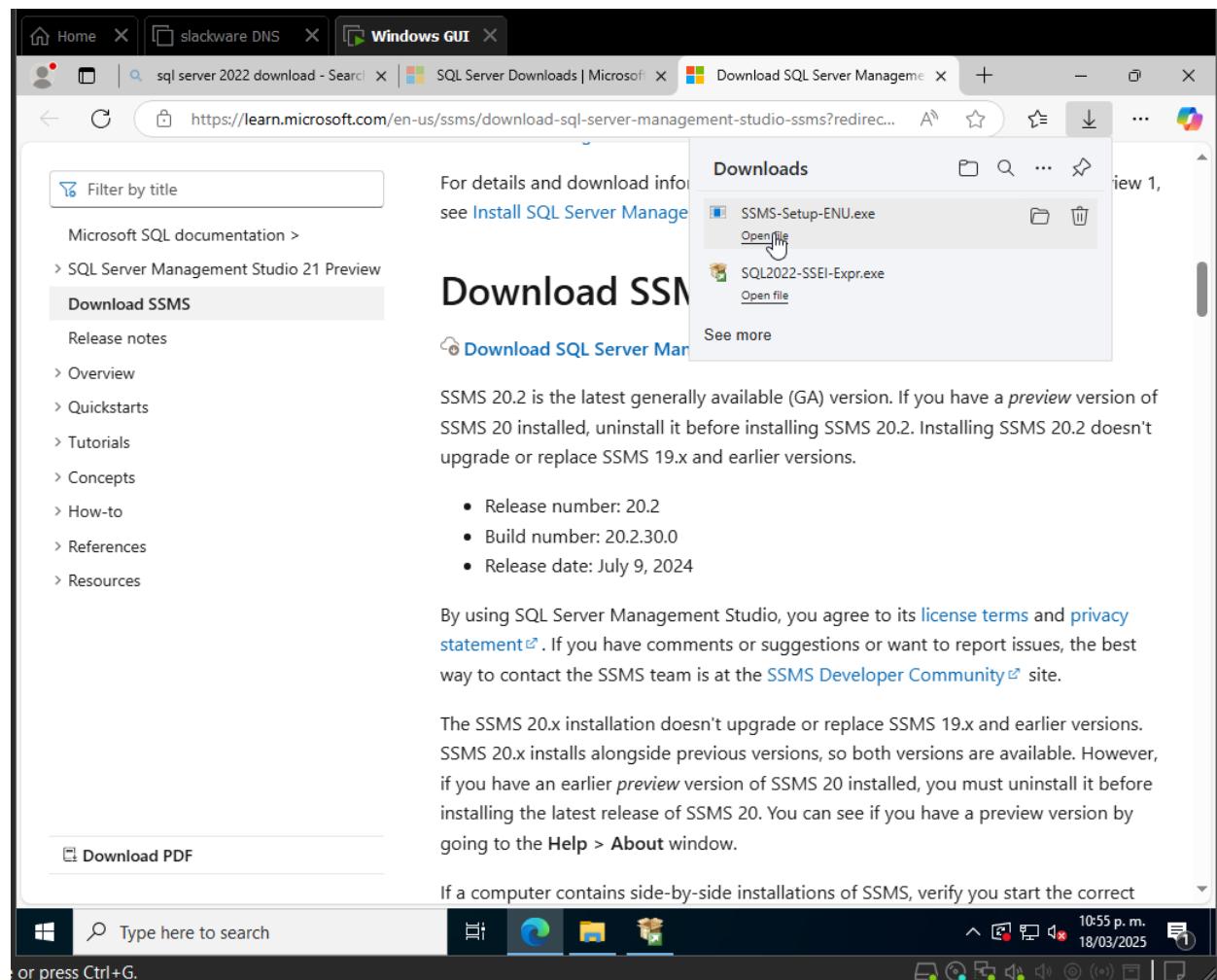
- Damos click en Install ssms



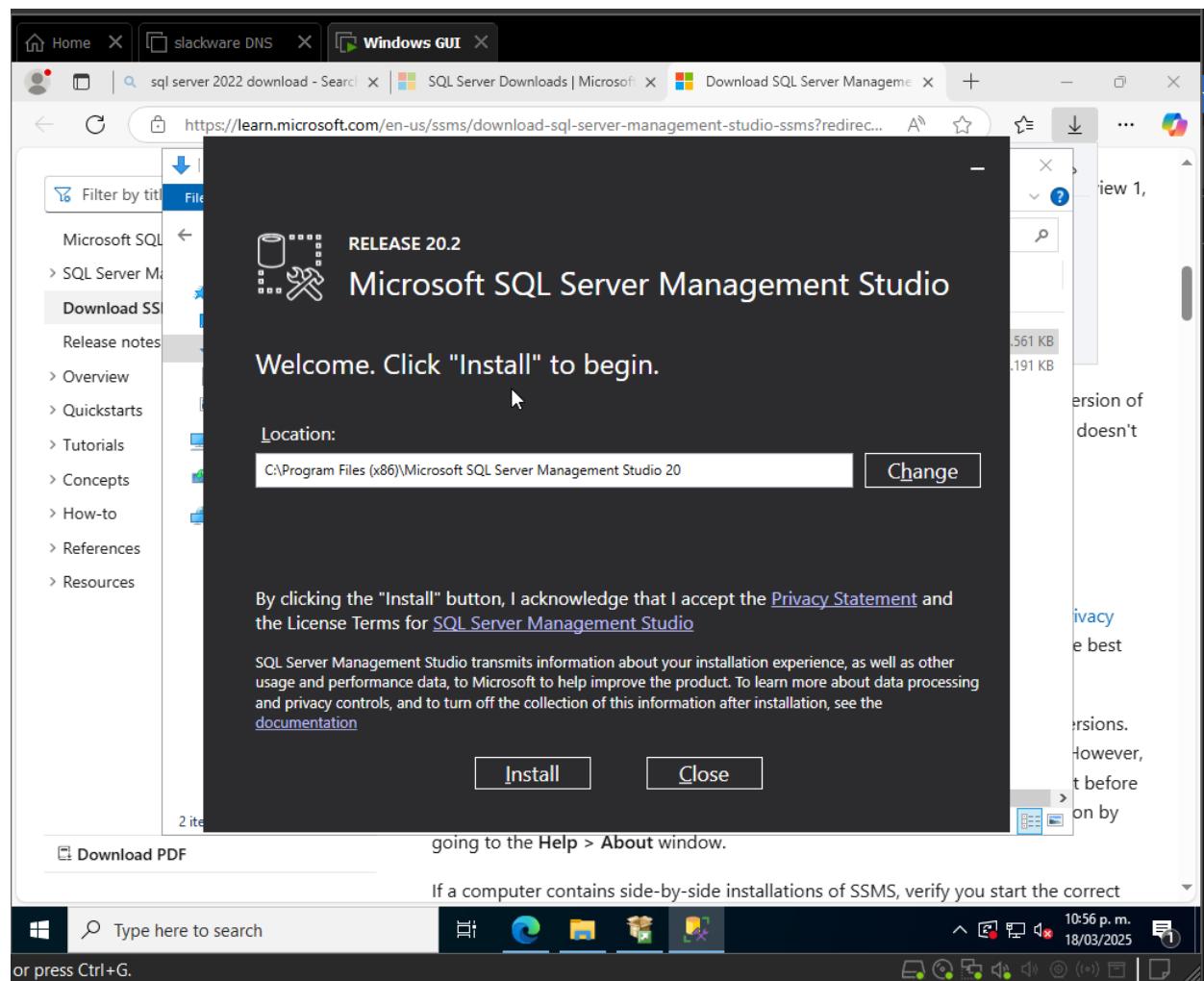
- Luego nos redirigira a una pagina en la cual daremos click en el siguiente apartado el cual nos va generar una descarga



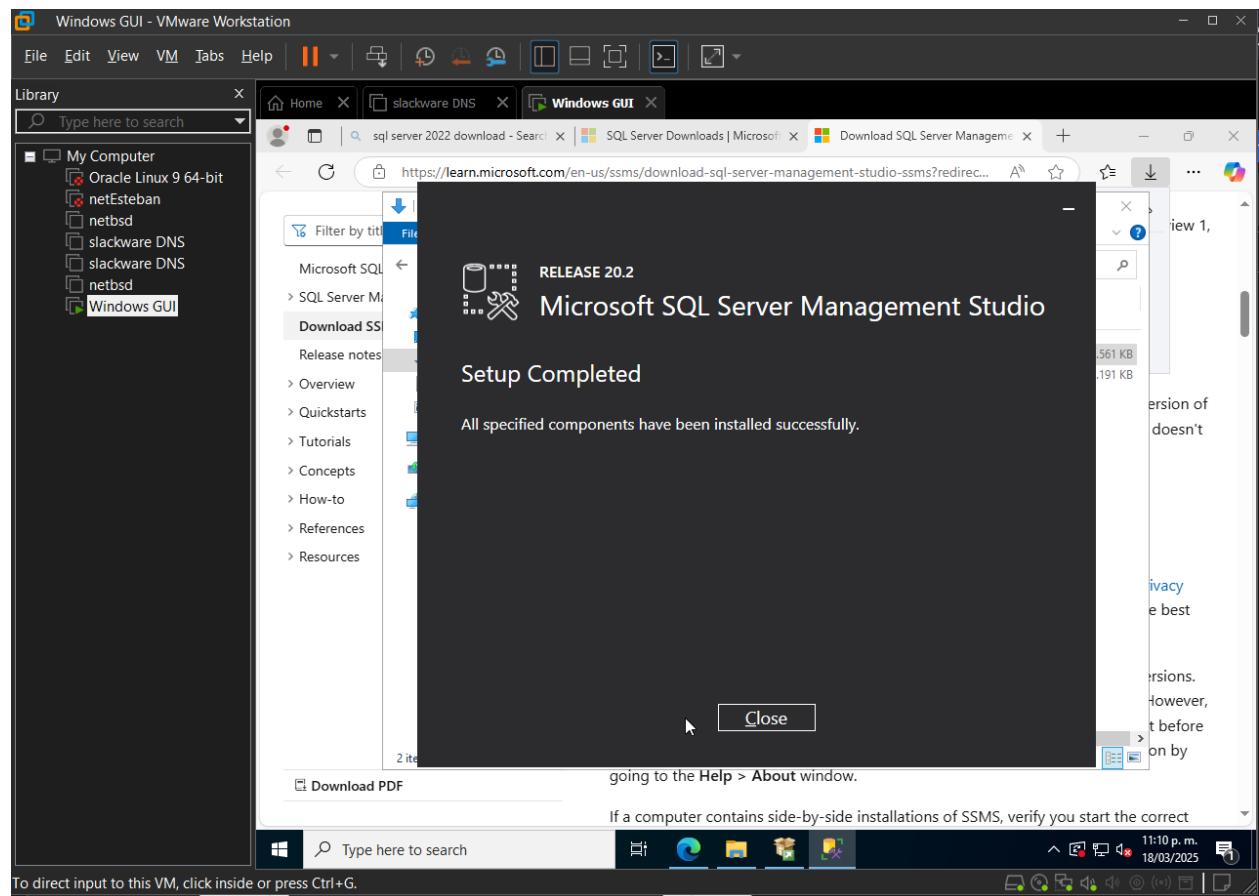
- Abrimos el archivo descargado



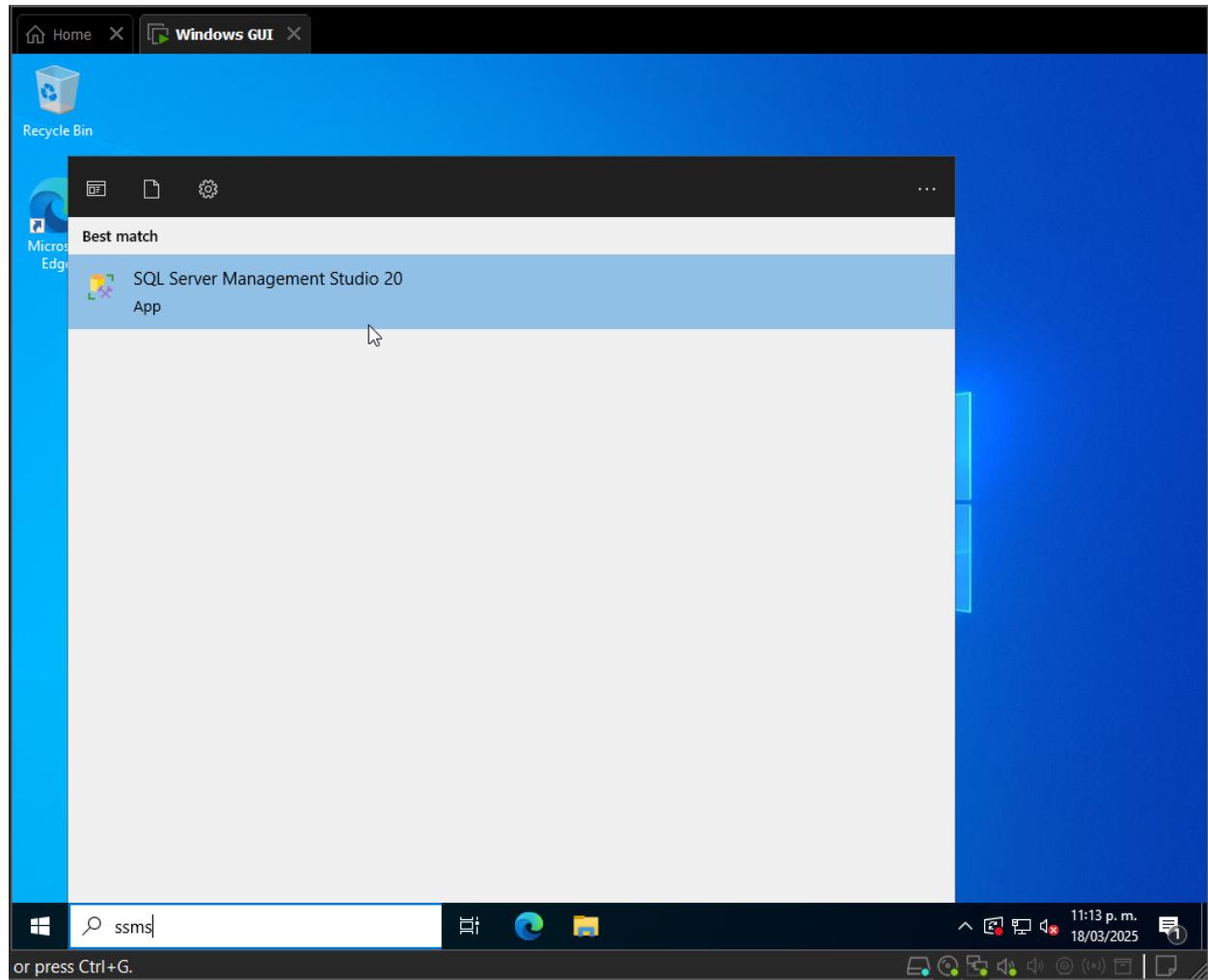
- Damos click en install



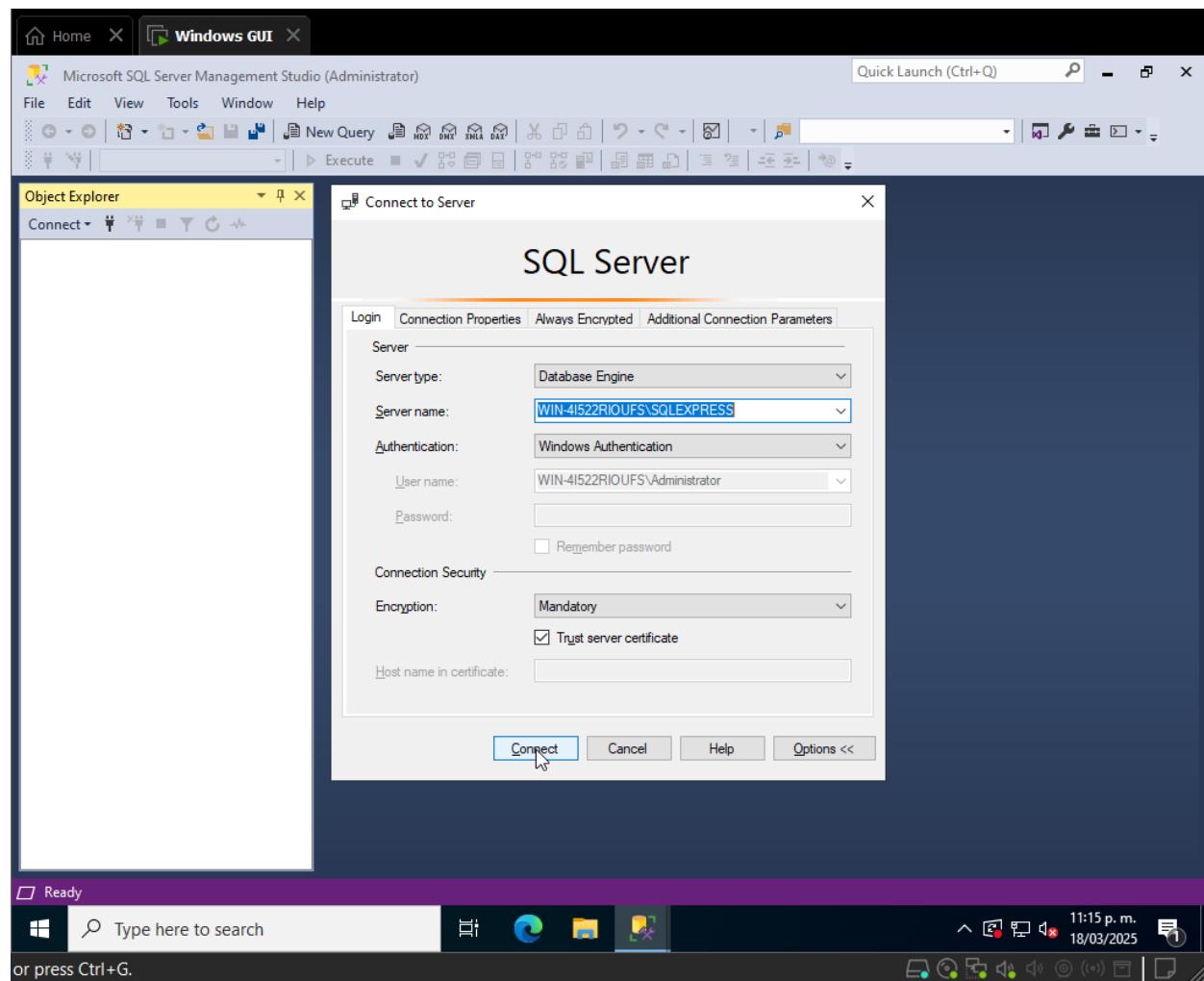
- Damos click en close



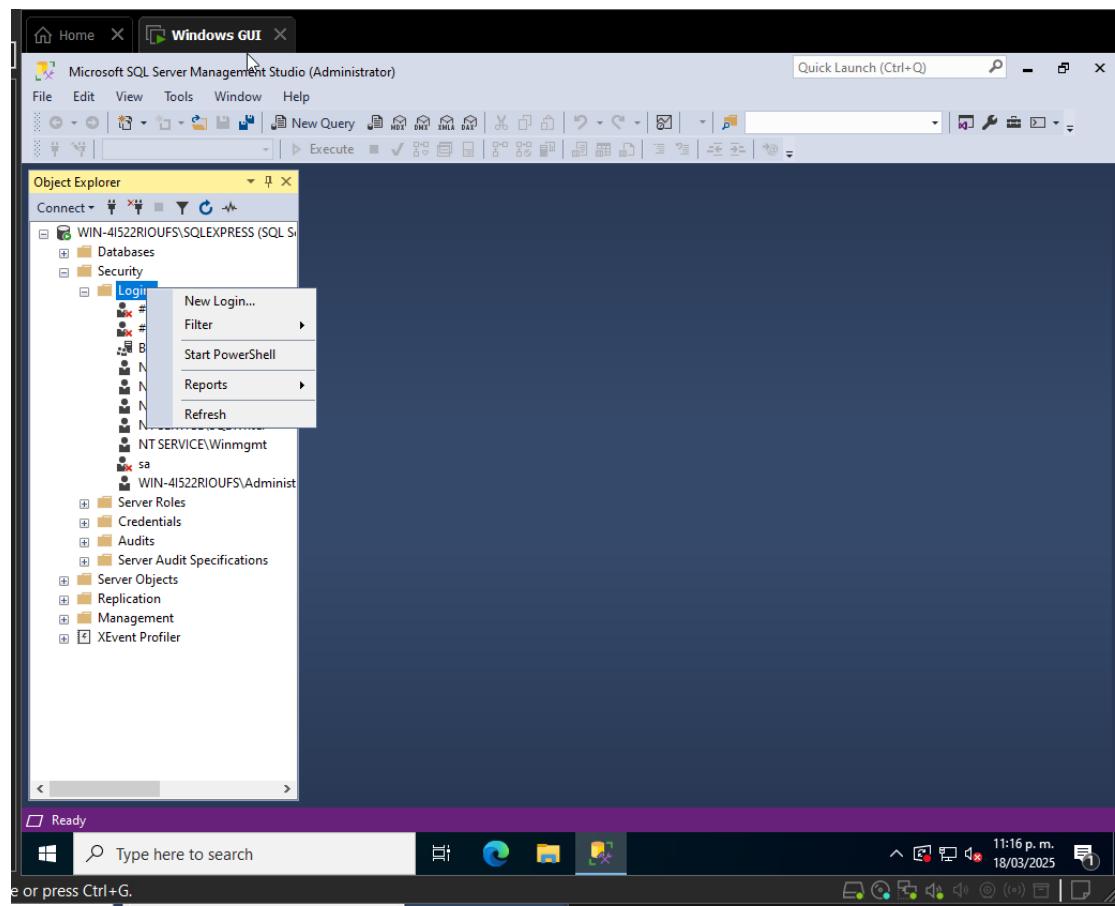
- Abrimos la aplicación buscandola en la barra de búsqueda (ssms)



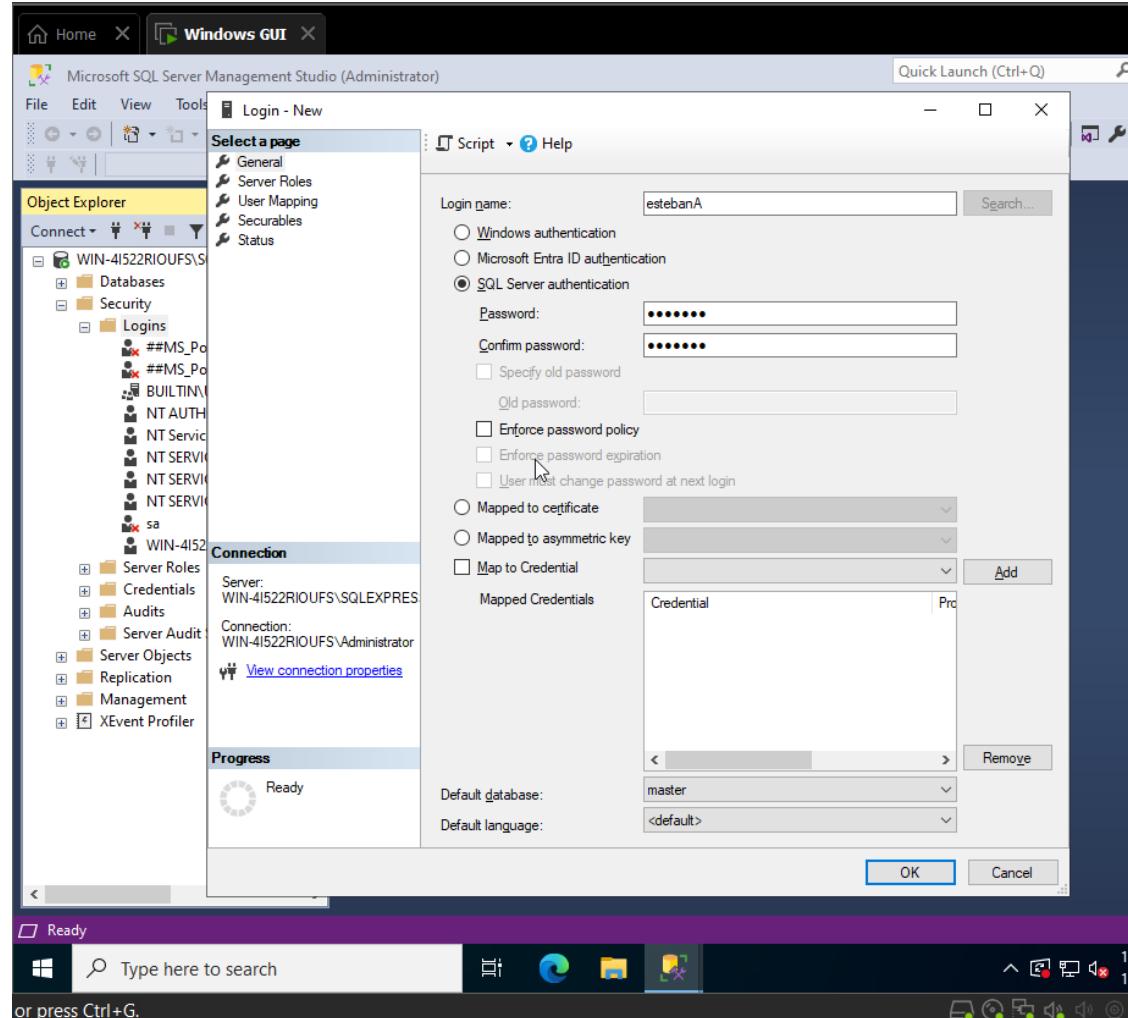
- Seleccionamos la opcion Connect



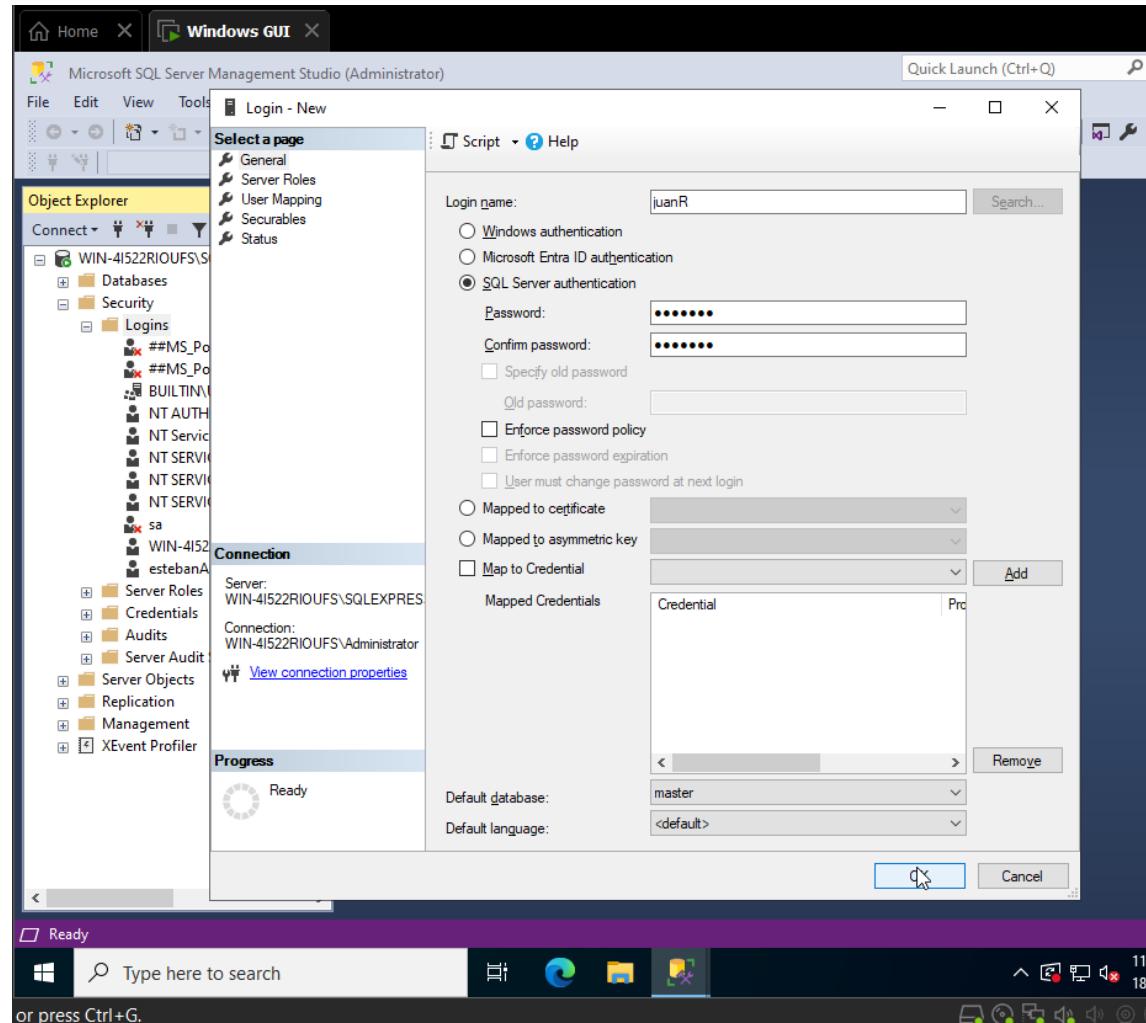
- Create a user for each group member. Use the students' names as the username.
 - Abrimos el apartado de Security y damos click derecho en Login para crear un nuevo usuario



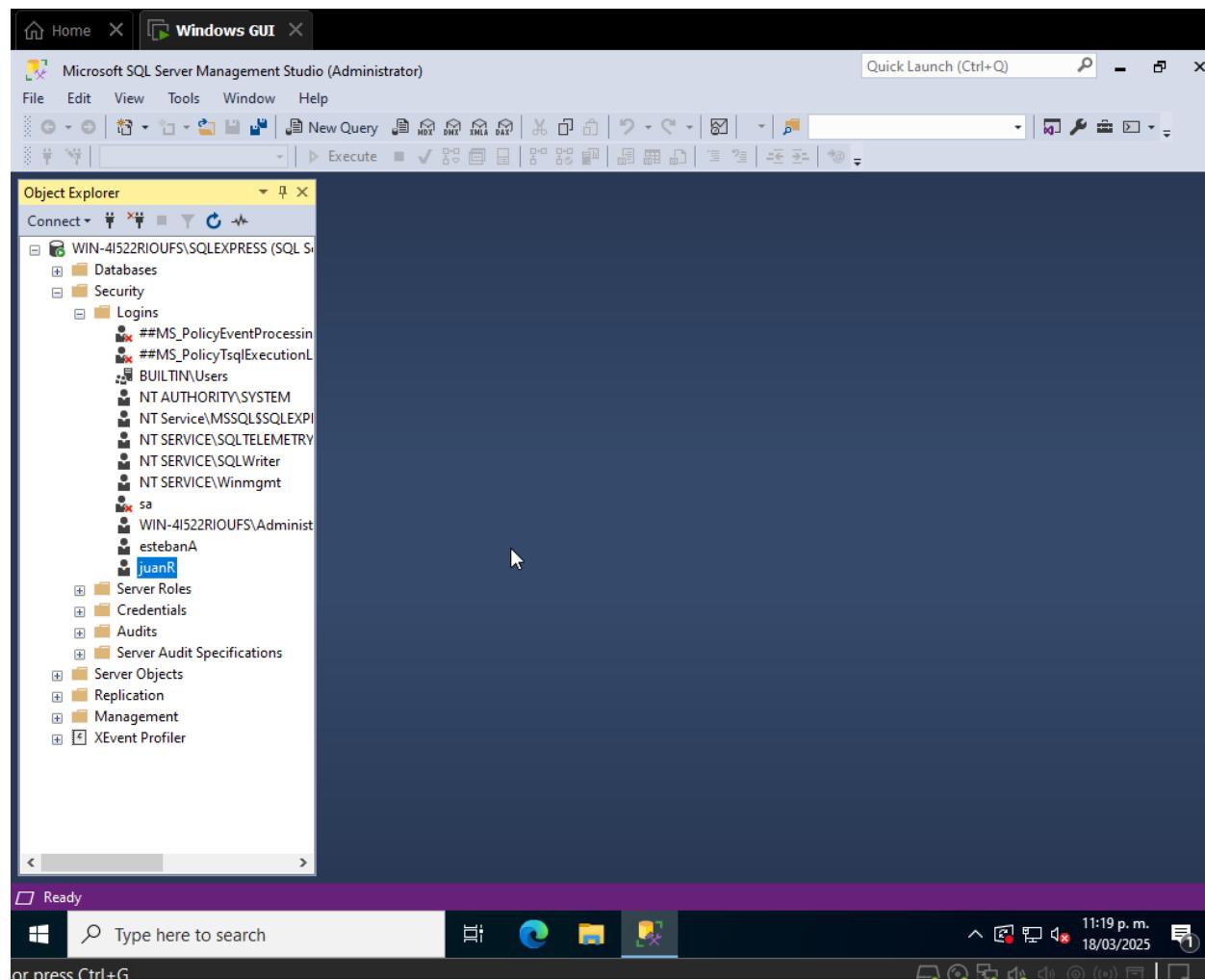
- Agregamos los usuarios con la contraseña root123
 - Esteban



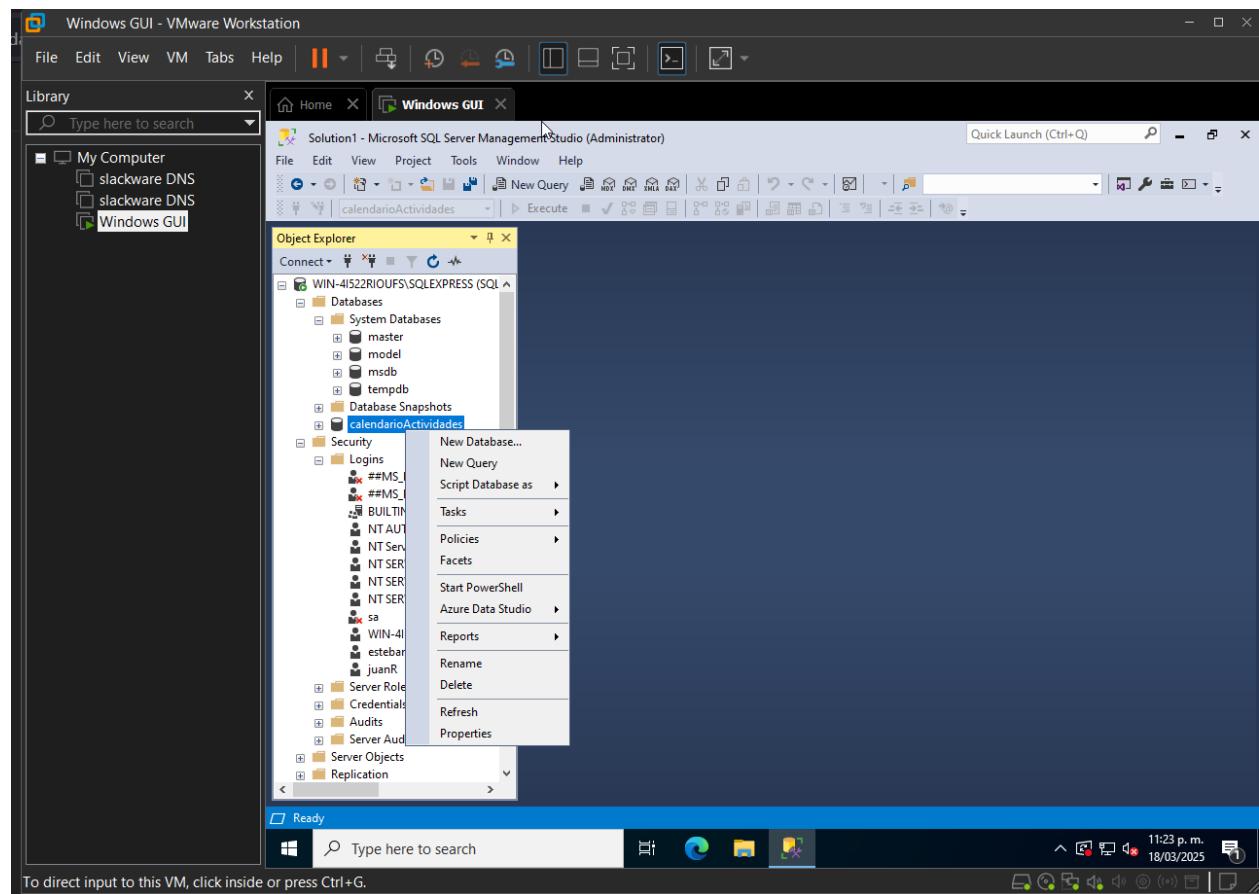
- Juan



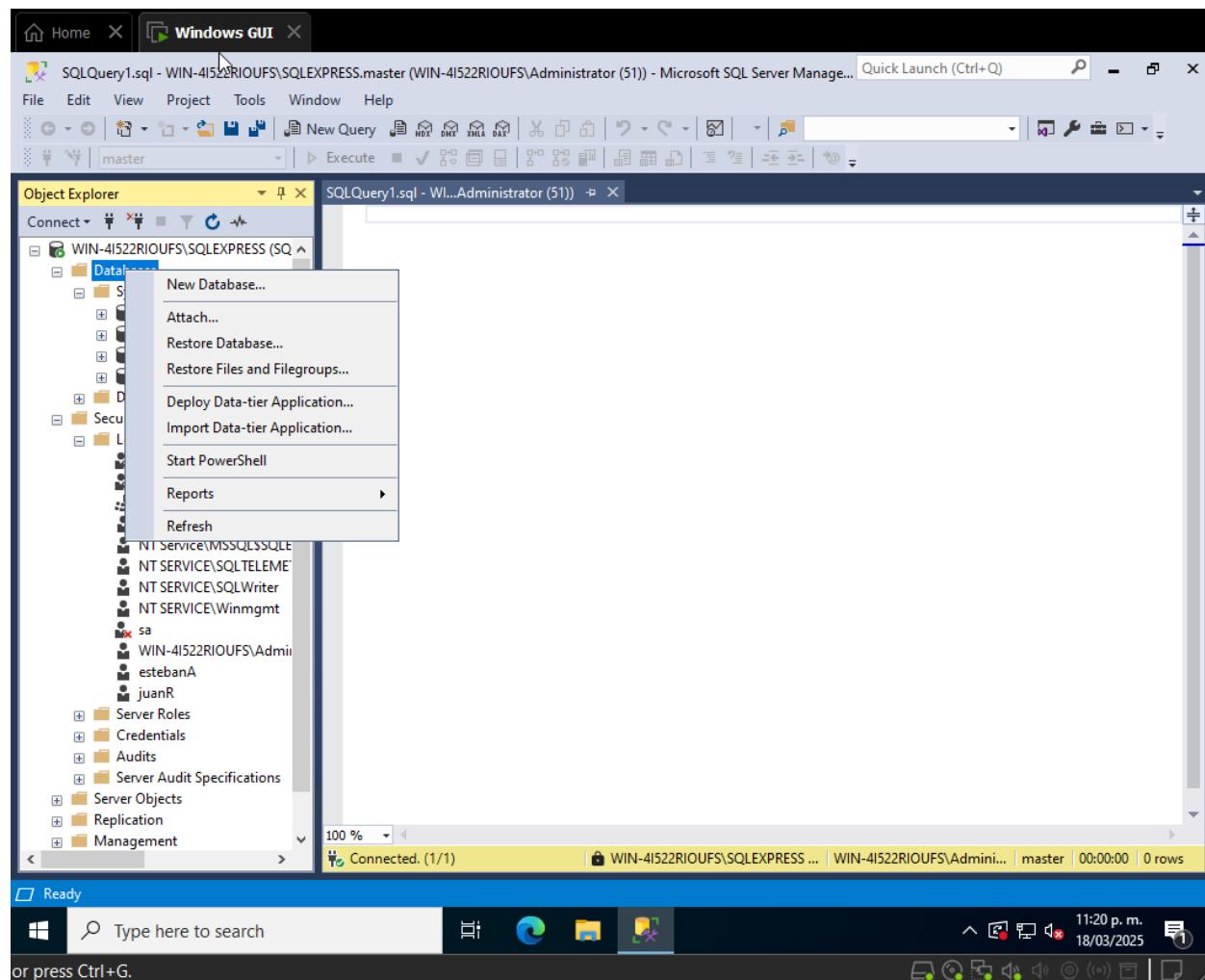
- Verificamos que los usuarios esten creados en la carpeta login



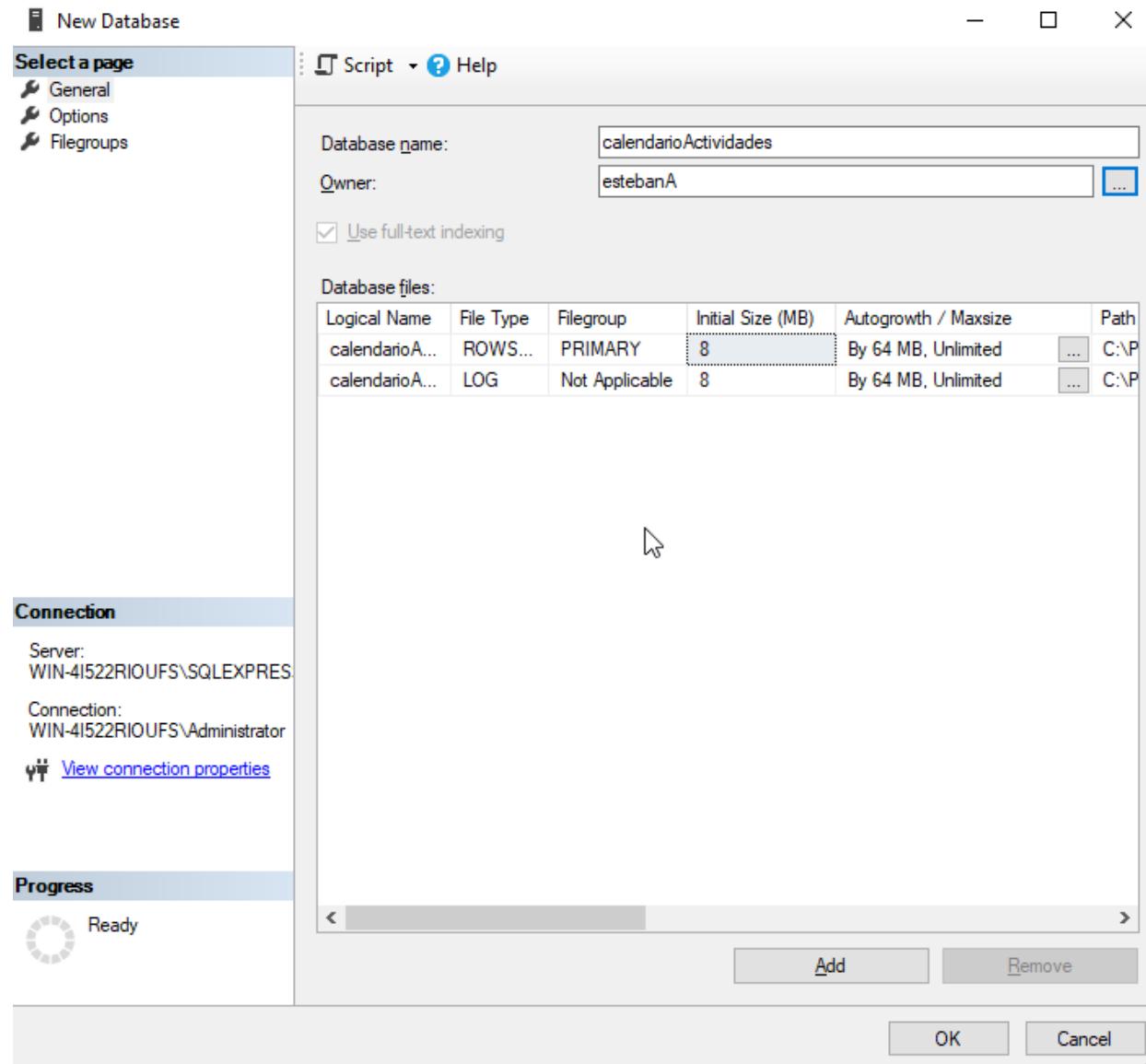
- Con la base de datos creada, damos click derecho en ella y seleccionamos la opción new query para crear las tablas



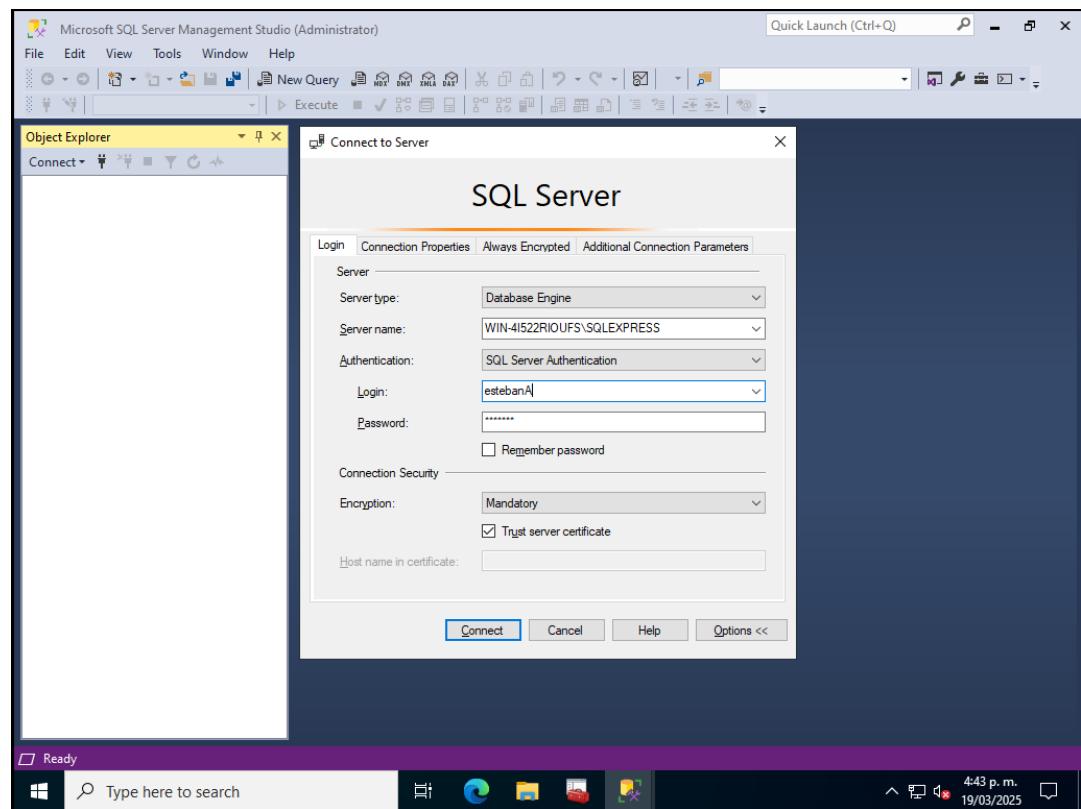
- Create a database to organize your monthly activity schedule. The database must have at least 3 tables. Each student should have access only to their own database.
 - Creamos la base de datos dandole click derecho a la carpeta Database



- Creamos la base de datos y asignamos de Owner a un usuario de los creados anteriormente (en este caso EstebanA)



- Agregamos una nueva conexión e ingresamos como nuestro usuario creado anteriormente



- Creamos las 3 tablas de la base de datos de calendario

```
GO
CREATE TABLE Usuarios (
    id INT IDENTITY PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    correo VARCHAR(100) UNIQUE NOT NULL
);
GO
CREATE TABLE Eventos (
    id INT IDENTITY PRIMARY KEY,
    titulo VARCHAR(50) NOT NULL,
    fecha DATE NOT NULL,
    descripcion VARCHAR(200),
    usuarioId INT FOREIGN KEY REFERENCES Usuarios(id) ON DELETE CASCADE
);
CREATE TABLE Recordatorios (
    id INT IDENTITY PRIMARY KEY,
    evento_id INT FOREIGN KEY REFERENCES Eventos(id) ON DELETE CASCADE,
    fechaHora DATETIME NOT NULL,
    mensaje VARCHAR(255) NOT NULL
);
GO
```

100 % ▶ Messages
Commands completed successfully.
Completion time: 2025-03-18T23:31:12.9140311-05:00

- Insert data into the databases.
 - Poblamos las 3 tablas creadas anteriormente

```
INSERT INTO Usuarios (nombre, correo) VALUES
('nombre 1', 'correo 1'),
('nombre 2', 'correo 2'),
('nombre 3', 'correo 3');

INSERT INTO Eventos (titulo, fecha, descripcion, usuarioId) VALUES
('titulo 1', '2025-03-18', 'descripcion 1', 1),
('titulo 1', '2025-03-19', 'descripcion 2', 1),
('titulo 1', '2025-03-20', 'descripcion 3', 1);

INSERT INTO Recordatorios (evento_id, fechaHora, mensaje ) VALUES
(1, '2025-03-18 09:00:00', 'mensaje 1'),
(2, '2025-03-19 09:00:00', 'mensaje 2'),
(3, '2025-03-20 09:00:00', 'mensaje 3');
```

% < Messages

(3 rows affected)

(3 rows affected)

(3 rows affected)

- Verificamos que se hayan creado los datos correctamente con una consulta en las tablas

```

SELECT * FROM Usuarios;
SELECT * FROM Eventos;
SELECT * FROM Recordatorios;

```

Results

| | id | nombre | correo |
|---|----|----------|----------|
| 1 | 1 | nombre 1 | correo 1 |
| 2 | 2 | nombre 2 | correo 2 |
| 3 | 3 | nombre 3 | correo 3 |

| | id | título | fecha | descripcion | usuarioId |
|---|----|----------|------------|---------------|-----------|
| 1 | 1 | título 1 | 2025-03-18 | descripcion 1 | 1 |
| 2 | 2 | título 1 | 2025-03-19 | descripcion 2 | 1 |
| 3 | 3 | título 1 | 2025-03-20 | descripcion 3 | 1 |

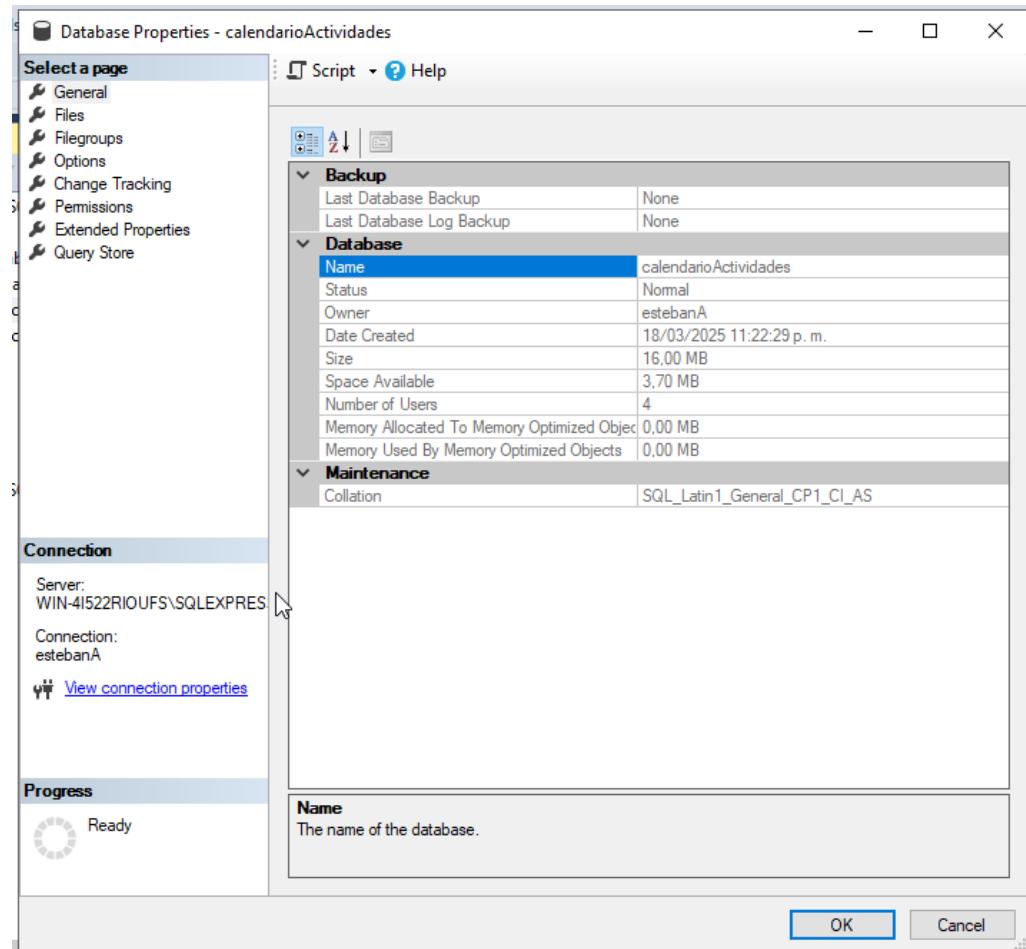
| | id | evento_id | fechaHora | mensaje |
|---|----|-----------|-------------------------|-----------|
| 1 | 1 | 1 | 2025-03-18 09:00:00.000 | mensaje 1 |
| 2 | 2 | 2 | 2025-03-19 09:00:00.000 | mensaje 2 |
| 3 | 3 | 3 | 2025-03-20 09:00:00.000 | mensaje 3 |

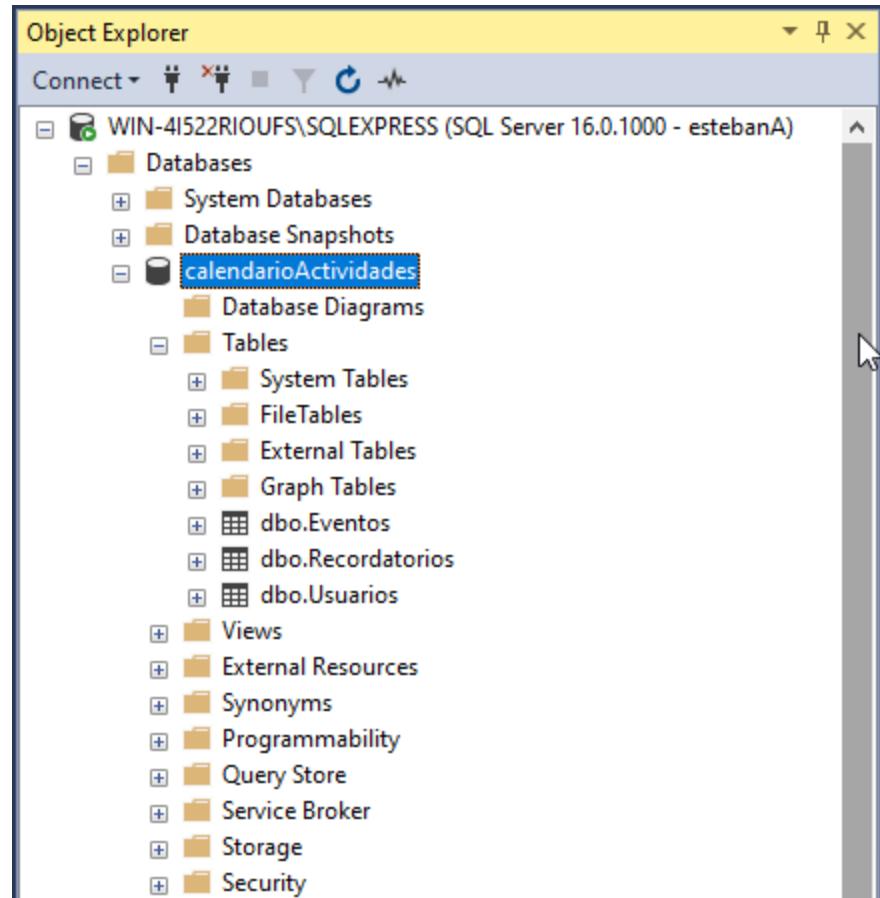
Query executed successfully | WIN-4I522RIOUFS\SQLEXPRESS ... | WIN-4I522RIOUFS\Administrador... | calendarioActividades | 00:00:00 | 9 rows

Ln:45 Col:29 Ch:29 INS

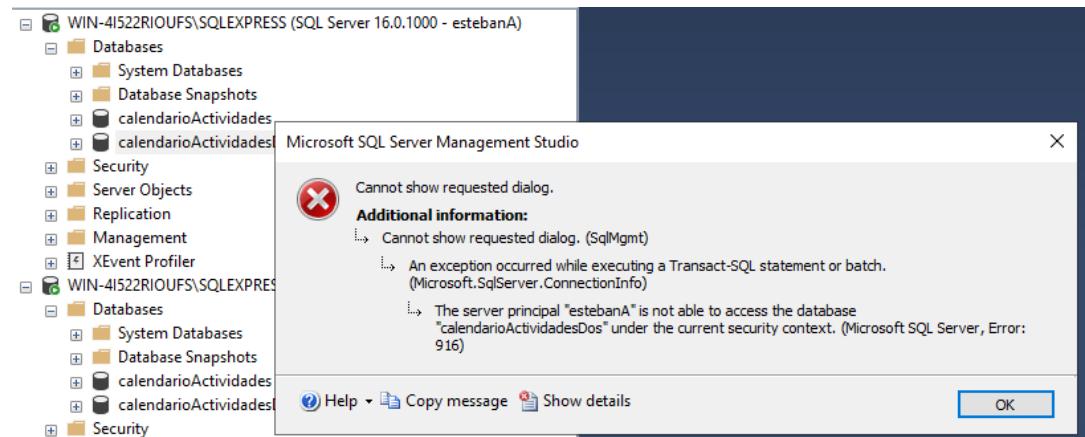
11:41 p.m. 18/03/2025

- Repetimos el mismo proceso para el otro usuario con otra base de datos diferentes
- Bases de datos finales
- Nota(las bases de datos aparecen en los dos usuarios pero no deja acceder a todas ya que estos solo tienen permisos en sus propias bases de datos)
 - Esteban

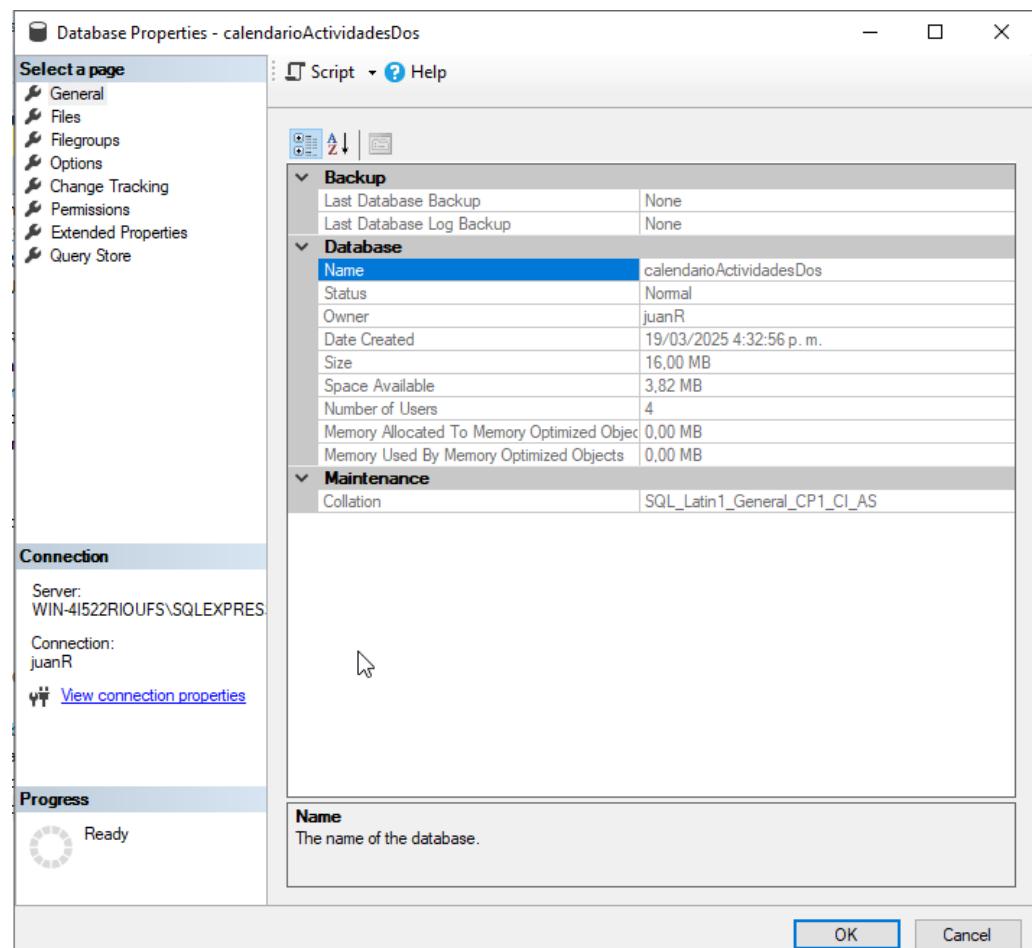


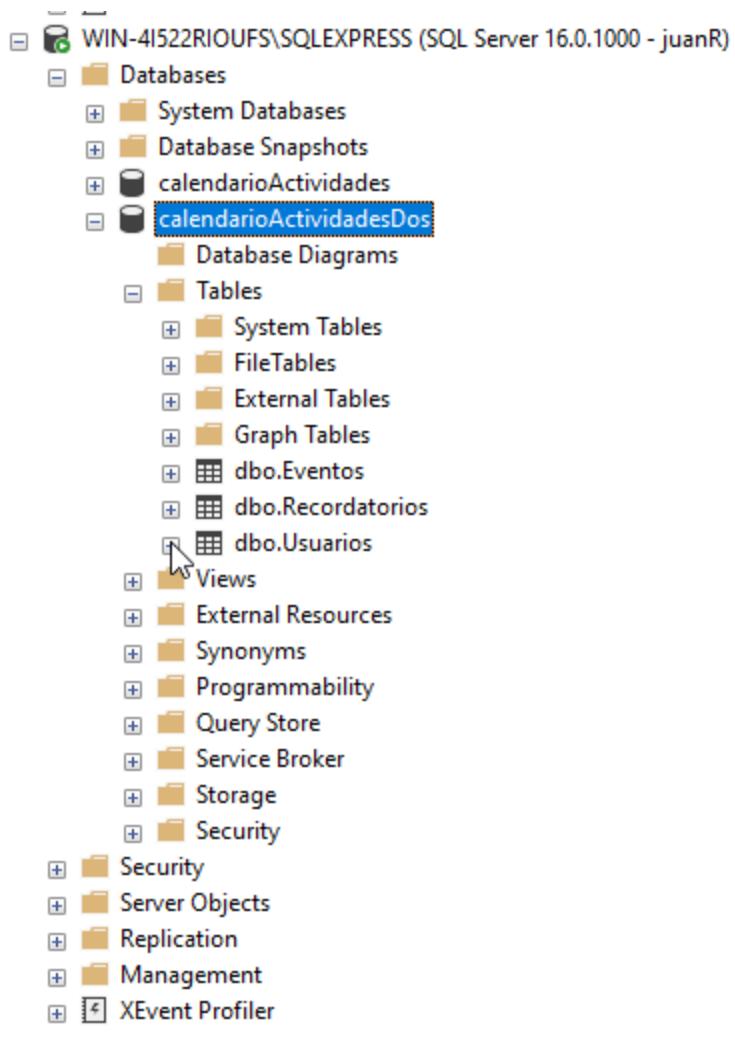


Intento de acceso a la base de datos de Juan

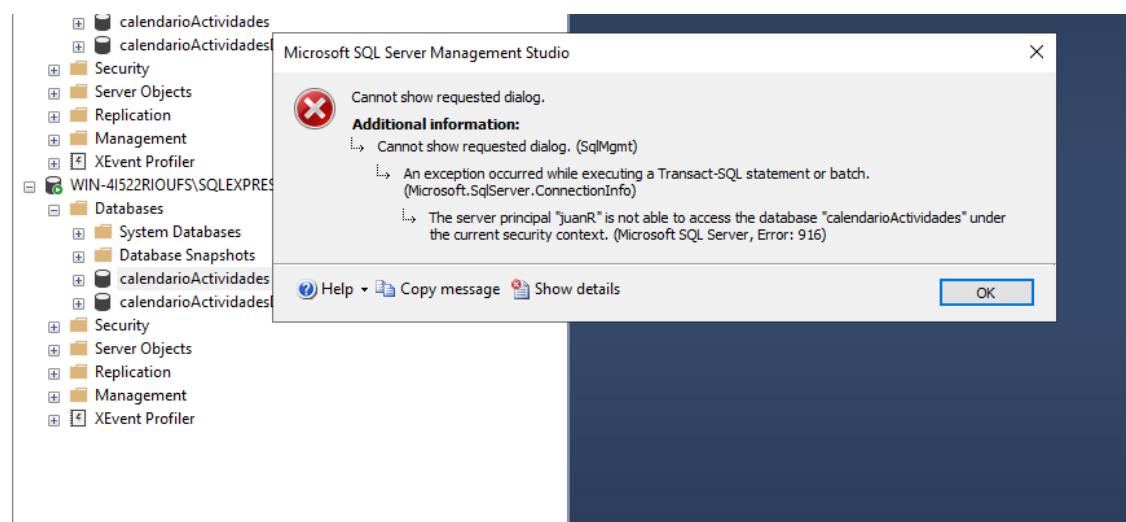


- Juan





Intento de acceso a la base de datos de esteban



SQL DATABASE - MICROSOFT AZURE

- a. Go to <https://azure.microsoft.com/en-us/pricing/purchase-options/azure-account/> and log in with your institutional email

The screenshot shows the Microsoft Azure portal interface. At the top, there's a header bar with the Microsoft Azure logo, a search bar, and a Copilot button. Below the header is a navigation bar with icons for creating a resource, App Services, Subscriptions, All resources, Azure DevOps organizations, Quick Start Center, AI services, Kubernetes services, Virtual Machines, and More services. The main content area is titled "Servicios de Azure". It features a "Recursos" section with tabs for Reciente (Recent) and Favorito (Favorite), displaying a list of resources like ReservasLaboratorios (App Service), AppReservasCvds, and CVDS (Resource Group). Below this is a "Navegar" section with links for Subscriptions, Groups of resources, All resources, and Panel. A "Herramientas" section follows, containing links to Microsoft Learn, Azure Monitor, Microsoft Defender for Cloud, and Cost Management. At the bottom, there are sections for Useful links (Technical documentation, Migration tools, Expert search, and Recent updates) and a mobile application download section for the Azure app.

- b. Once logged into the Azure portal, explore the available services.

- What is cloud computing, and what are some of the advantages of using a platform like Microsoft Azure compared to an on-premise infrastructure?

- La **computación en la nube** es un modelo de entrega de servicios informáticos (como servidores, almacenamiento, bases de datos, redes y software) a través de Internet.

Ventajas de Azure sobre una infraestructura local:

Escalabilidad: Permite aumentar o reducir los recursos según la demanda.

Costo basado en uso: No requiere grandes inversiones iniciales en hardware.

Alta disponibilidad: Azure tiene centros de datos en múltiples regiones para garantizar disponibilidad y recuperación ante desastres.

Seguridad y cumplimiento: Ofrece seguridad integrada con controles de acceso y cifrado.

Gestión simplificada: No es necesario mantener físicamente servidores ni hardware.

- What types of services does Microsoft Azure offer (IaaS, PaaS, SaaS), and how do they differ from one another?
 - Microsoft Azure ofrece los siguientes modelos de servicio:

IaaS (Infrastructure as a Service): Proporciona infraestructura virtualizada (máquinas virtuales, redes, almacenamiento). El usuario gestiona el software, mientras Azure administra la infraestructura. Ejemplo: **Azure Virtual Machines**.

PaaS (Platform as a Service): Proporciona un entorno de desarrollo listo para usar sin necesidad de administrar infraestructura. Ejemplo: **Azure App Services**.

SaaS (Software as a Service): Ofrece aplicaciones completamente gestionadas por Azure, listas para su uso. Ejemplo: **Microsoft 365**.

- What is the importance of regions and availability zones in Azure, and how do they affect service availability?
 - Azure organiza sus centros de datos en **regiones**, que son ubicaciones geográficas específicas. Dentro de cada región, hay zonas de disponibilidad, que son centros de datos físicamente separados para garantizar redundancia.
Importancia:
Alta disponibilidad: Si una zona falla, el servicio puede seguir funcionando desde otra.
Recuperación ante desastres: Permite replicar datos y aplicaciones para evitar pérdidas.
Reducción de latencia: Al seleccionar la región más cercana a los usuarios, se mejora el rendimiento.
- What is the difference between vertical scaling and horizontal scaling in Azure, and when would you choose one over the other?
 - **Escalado vertical:** Aumenta los recursos de una sola máquina (más CPU, RAM, almacenamiento). Se usa cuando la aplicación depende de una sola instancia.
Escalado horizontal: Agrega más instancias del servicio para distribuir la carga. Se usa cuando la aplicación debe manejar muchas conexiones simultáneas.
- How does using technologies like TLS (Transport Layer Security) on the transport layer affect accessing Azure SQL Database compared to a local virtual machine database?
 - Azure SQL Database requiere conexiones seguras mediante TLS (Transport Layer Security), lo que cifra los datos en tránsito para evitar ataques de interceptación.
Diferencias clave:
Azure SQL Database: Siempre usa TLS para conexiones seguras, lo que protege los datos en tránsito.
Base de datos en VM local: Puede configurarse con o sin TLS, pero requiere administración manual para habilitar seguridad.

- From a transport layer perspective, how does the handling of TCP connections differ between a SQL database hosted on a local virtual machine and Azure SQL Database?
 - **Máquina virtual local:** El servidor maneja directamente las conexiones TCP y la configuración de red.
 - Azure SQL Database:** Usa un balanceador de carga interno y conexión a través de una **pasarela de datos**, lo que permite alta disponibilidad y escalabilidad sin que el usuario gestione conexiones directamente.

c. Use the Azure SQL Database service to manage records of books and scientific articles. The database must have at least 3 tables.

The screenshot shows the Azure portal interface for the RecoDB database. At the top, there's a navigation bar with 'redescomputadores.database.windows.net/RecoDB' and a close button. Below it, there are tabs for 'Home', 'New Query', 'New Notebook', 'Refresh', and 'Learn More'. The main content area displays the database details: Edition: General Purpose, Compatibility Level: 160, Pricing Tier: GP_S_Gen5_1, and Owner: reco. A search bar is present above a table listing three tables: Books, Categories, and ScientificArticles, all belonging to the dbo schema and being Tables.

| Name | Schema | Type | Actions |
|--------------------|--------|-------|---------|
| Books | dbo | Table | ... |
| Categories | dbo | Table | ... |
| ScientificArticles | dbo | Table | ... |

d. Insert data into the database.

```

1  SELECT * FROM Books;
2  SELECT * FROM ScientificArticles;
3  SELECT * FROM Categories;
4
5

```

Results Messages

| | BookID | Title | Author | PublishedYear | CategoryID |
|---|--------|---------------------------|----------------|---------------|------------|
| 1 | 1 | El origen de las especies | Charles Darwin | 1859 | 1 |
| 2 | 2 | Inteligencia Artificial | Stuart Russell | 2010 | 2 |

| | ArticleID | Title | Author | Journal | PublicationYear | CategoryID |
|---|-----------|-------------------------------|-----------------|------------|-----------------|------------|
| 1 | 1 | Evolución y selección natural | Richard Dawkins | Nature | 1995 | 1 |
| 2 | 2 | Redes neuronales | Geoffrey Hinton | AI Journal | 2018 | 2 |

| | CategoryID | CategoryName |
|---|------------|--------------|
| 1 | 1 | Ciencia |
| 2 | 2 | Tecnología |
| 3 | 3 | Matemáticas |

- e. Record a video (no longer than 5 minutes) demonstrating the connection to the database, the created tables, and the inserted data.

https://www.canva.com/design/DAGiZy1w-x4/myszXZfLL2UfGps6b0YtvA/edit?utm_content=DAGiZy1w-x4&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton

OTHER DATABASE ENGINE CONFIGURATIONS

1. On the servers where you installed the database engines, configure the operating system so that the database engines automatically start when the OS boots.

- a. Slackware

- i. Ingresamos a la ruta /etc/rc.d/rc.local

```
root@juanito:~# nano /etc/rc.d/rc.local_
```

- ii. Agregamos el inicio automatico del postgresql

```
GNU nano 6.0                               /etc/rc.d/rc.local
#!/bin/bash
#
## /etc/rc.d/rc.local: Local system initialization script.
#
## Put any local startup commands in here. Also, if you have
## anything that needs to be run at shutdown time you can
## make an /etc/rc.d/rc.local_shutdown script and put those
## commands in there.

/etc/rc.d/rc.ntpd start
su - postgres -c "pg_ctl -D /var/lib/pgsql/data start"
```



- iii. Damos permisos de ejecución

```
root@juanito:~# chmod +x /etc/rc.d/rc.postgresql
root@juanito:~#
```

- iv. Verificamos que prenda dando reboot y mirando los mensajes de inicio

```

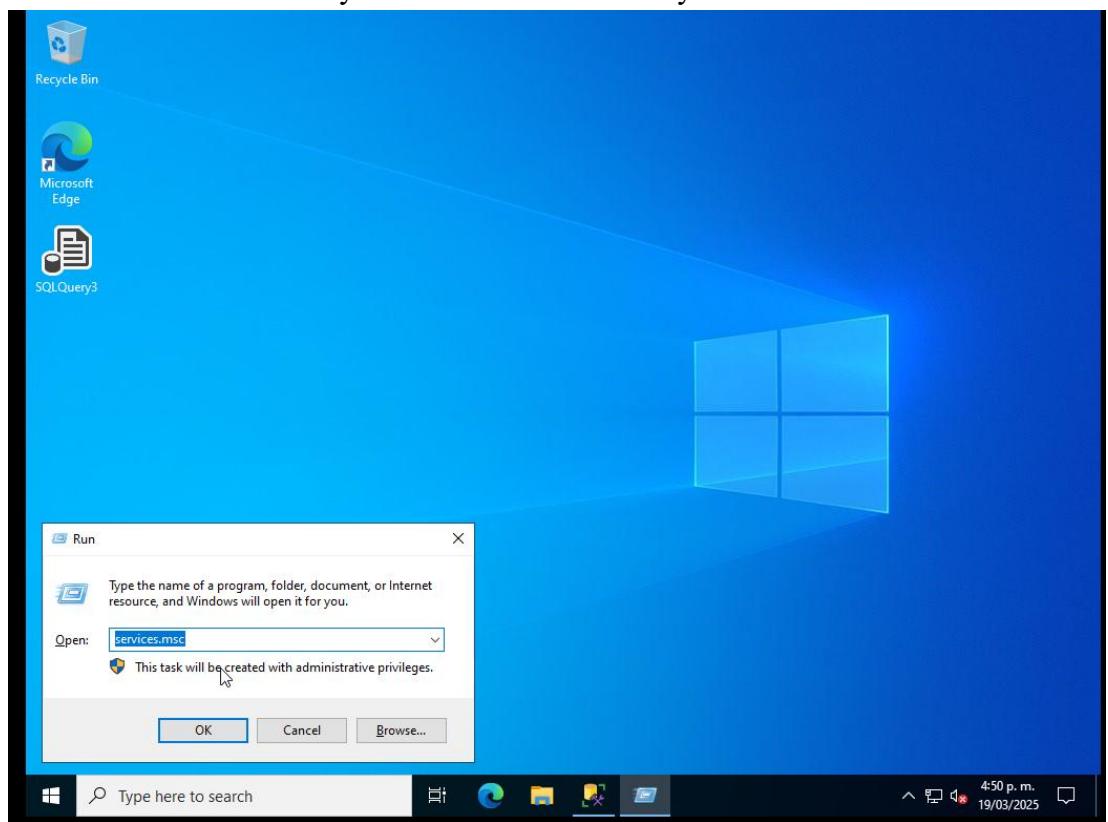
waiting for server to start....2025-03-19 17:29:23.634 -05 [1214] LOG:  starting PostgreSQL 14.17 on
i586-slackware-linux-gnu, compiled by gcc (GCC) 11.2.0, 32-bit
2025-03-19 17:29:23.636 -05 [1214] LOG:  listening on IPv6 address "::1", port 5432
2025-03-19 17:29:23.636 -05 [1214] LOG:  listening on IPv4 address "127.0.0.1", port 5432
2025-03-19 17:29:23.639 -05 [1214] LOG:  listening on Unix socket "/tmp/.s.PGSQL.5432"
2025-03-19 17:29:23.890 -05 [1215] LOG:  database system was shut down at 2025-03-19 17:25:37 -05
done
server started

Welcome to Linux 5.15.19-smp i686 (tty1)
juanito login: █

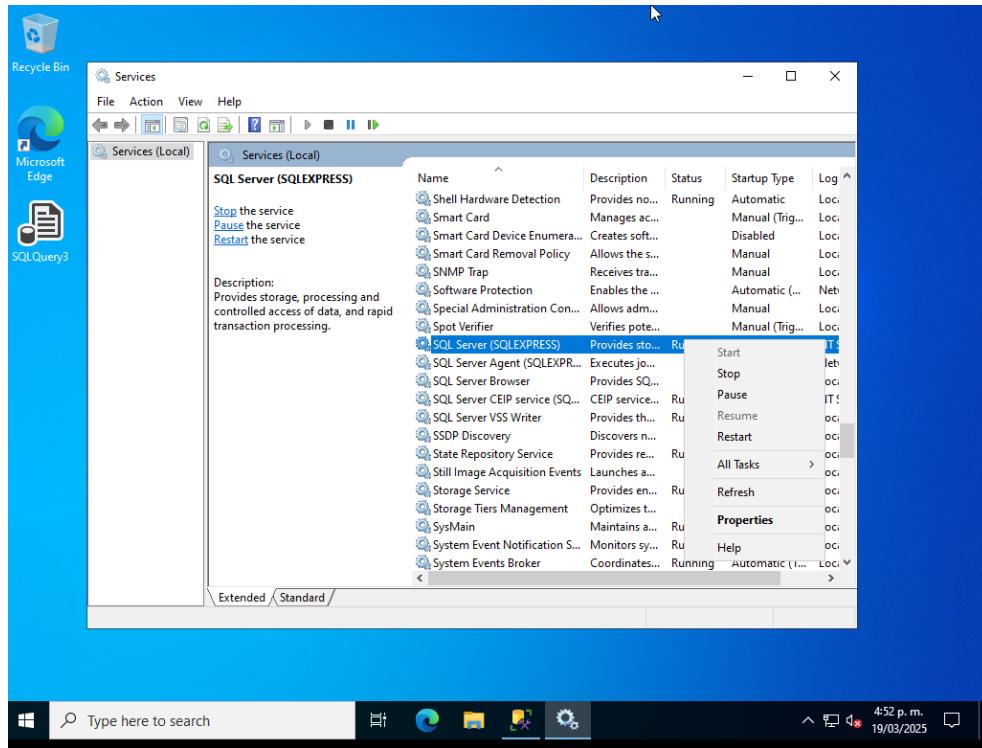
```

b. Windows

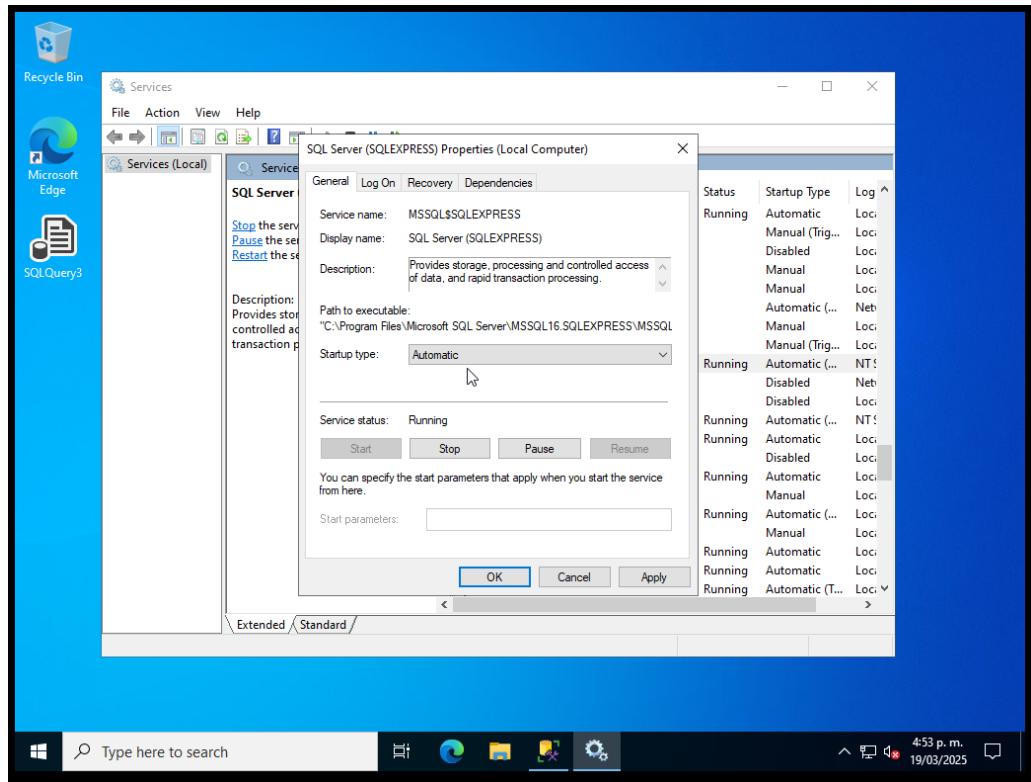
- Presionamos windows +r y escribimos servicios.ms y damos ENTER



- Buscamos el servicio de SQL server , presionamos click derecho y seleccionamos Properties



- iii. En startup tpe selecciónamos automatic y presionamos Apply



2. Using a database connection client (e.g., DBeaver), connect to your databases from a remote machine and view the table contents

- a. Slackware

- i. Ingresamos a la ruta /var/lib/pgsql/data/postgresql.conf

```
root@juanito:~# nano /var/lib/pgsql/data/postgresql.conf
```

- ii. Descomentamos la linea listen_addresses y agregamos el '*' el cual permite escuchar todas las direcciones de red

```
GNU nano 6.0          /var/lib/pgsql/data/postgresql.conf          Modified
# option or PGDATA environment variable, represented here as ConfigDir.

#data_directory = 'ConfigDir'          # use data in another directory
#                                         # (change requires restart)
#hba_file = 'ConfigDir/pg_hba.conf'    # host-based authentication file
#                                         # (change requires restart)
#ident_file = 'ConfigDir/pg_ident.conf' # ident configuration file
#                                         # (change requires restart)

# If external_pid_file is not explicitly set, no extra PID file is written.
#external_pid_file = ''                # write an extra PID file
#                                         # (change requires restart)

#-----
# CONNECTIONS AND AUTHENTICATION
#-----

# - Connection Settings -

listen_addresses = '*'          #_what IP address(es) to listen on;
#                               # comma-separated list of addresses;
#                               # defaults to 'localhost'; use '*' for all
#                               # (change requires restart)
#port = 5432
max_connections = 100
#superuser_reserved_connections = 3
#unix_socket_directories = '/tmp'
#unix_socket_group = ''
#unix_socket_permissions = 0777
#bonjour = off
#                               # advertise server via Bonjour

^K Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute   ^C Location  M-U Undo
^X Exit       ^R Read File   ^N Replace   ^U Paste     ^J Justify   ^- Go To Line M-E Redo
```

- iii. Ahora ingresamos a la ruta /var/lib/pgsql/data/pg_hba.conf

```
root@juanito:~# nano /var/lib/pgsql/data/pg_hba.conf
```

- iv. Agregamos la linea host all all 0.0.0.0/0 md5 para permitir conexiones remotas desde cualquier IP

```

GNU nano 6.0          /var/lib/pgsql/data/pg_hba.conf      Modified
# This file is read on server startup and when the server receives a
# SIGHUP signal. If you edit the file on a running system, you have to
# SIGHUP the server for the changes to take effect, run "pg_ctl reload",
# or execute "SELECT pg_reload_conf()".
#
# Put your actual configuration here
#
#
# If you want to allow non-local connections, you need to add more
# "host" records. In that case you will also need to make PostgreSQL
# listen on a non-local interface via the listen_addresses
# configuration parameter, or via the -i or -h command line switches.
#
# CAUTION: Configuring the system for local "trust" authentication
# allows any local user to connect as any PostgreSQL user, including
# the database superuser. If you do not trust all your local users,
# use another authentication method.

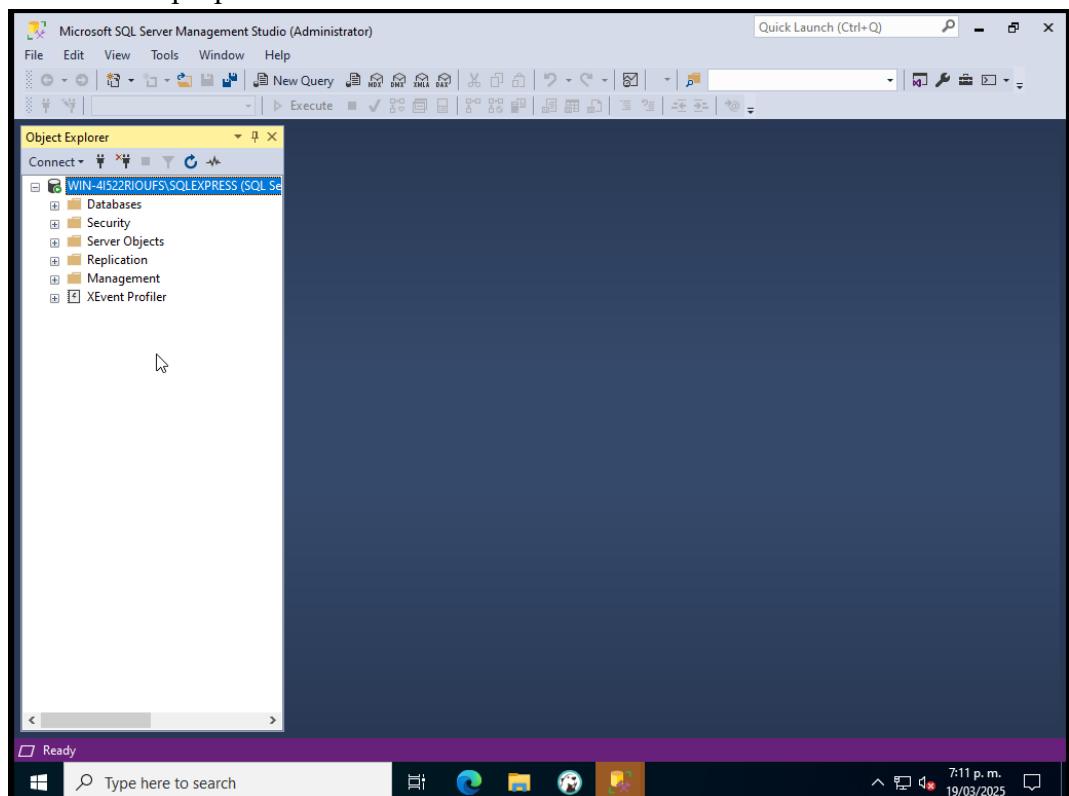
# TYPE   DATABASE      USER      ADDRESS      METHOD
#
# "local" is for Unix domain socket connections only
local   all            all          trust
# IPv4 local connections:
host    all            all          127.0.0.1/32    trust
# IPv6 local connections:
host    all            all          ::1/128       trust
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication   all          trust
host    replication   all          127.0.0.1/32    trust
host    replication   all          ::1/128       trust
host    all            all          0.0.0.0/0      md5

^G Help      ^O Write Out  ^W Where Is  ^K Cut      ^T Execute  ^C Location  M-U Undo
^X Exit      ^R Read File  ^P Replace   ^U Paste     ^J Justify  ^- Go To Line M-E Redo

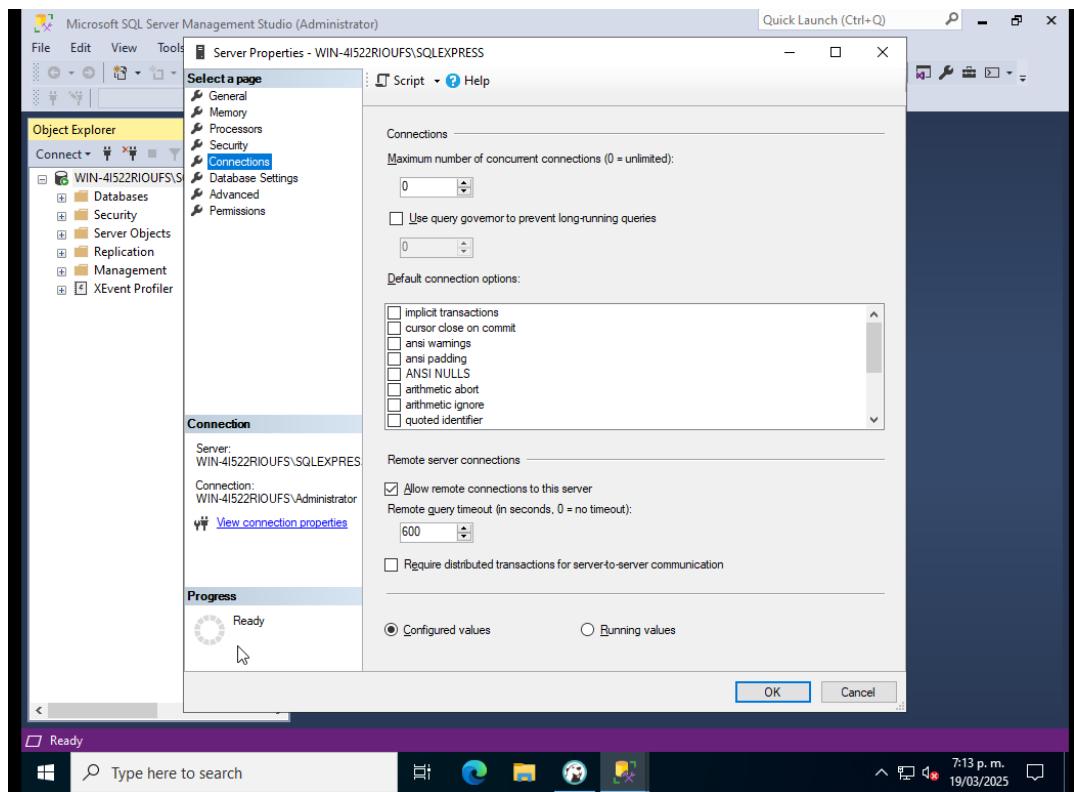
```

b. Windows

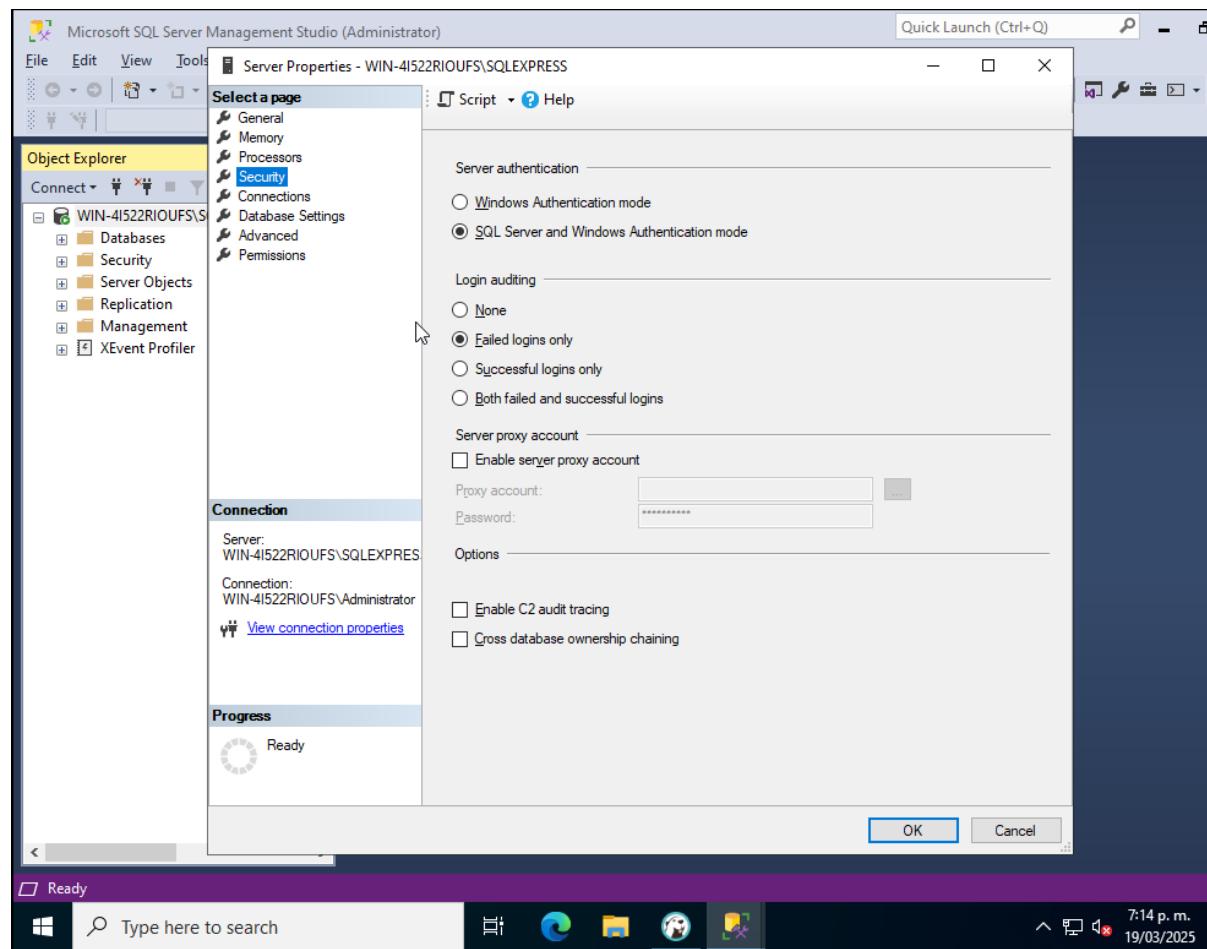
i. Abrimos las propiedades de la base de datos en ssms



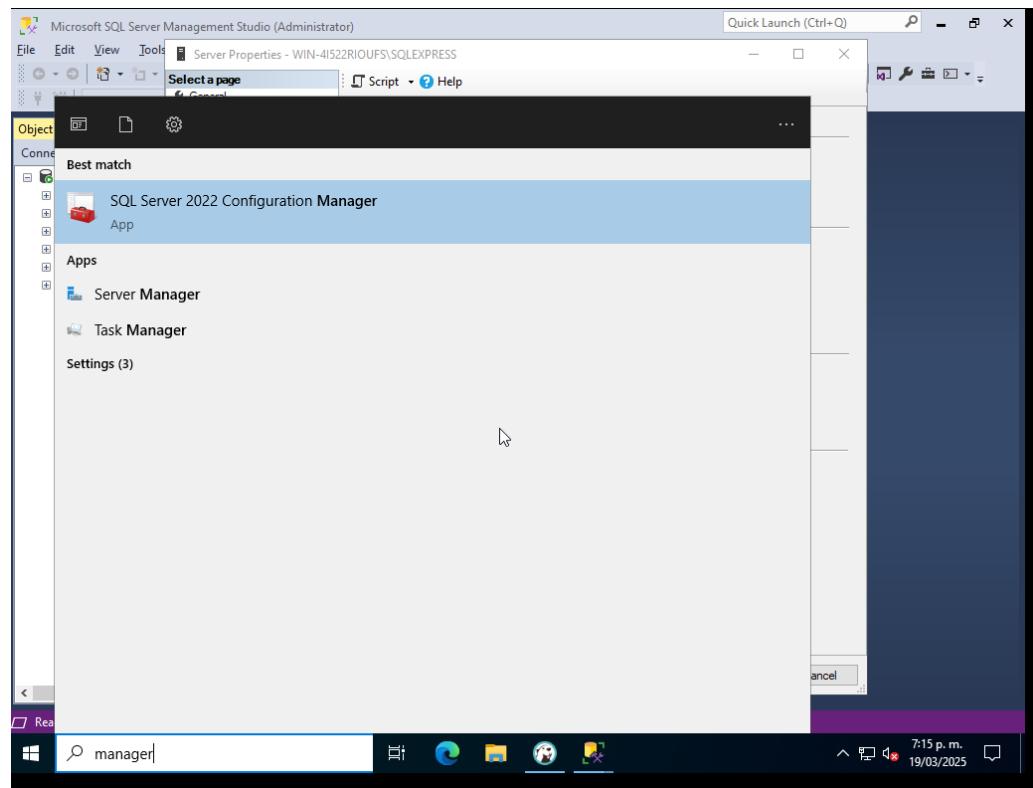
- ii. En el apartado connections seleccionamos la opción allow remote connections to this server



- iii. Ahora navegamos a seguridad y seleccionamos la opción SQL Server and Windows authentication mode

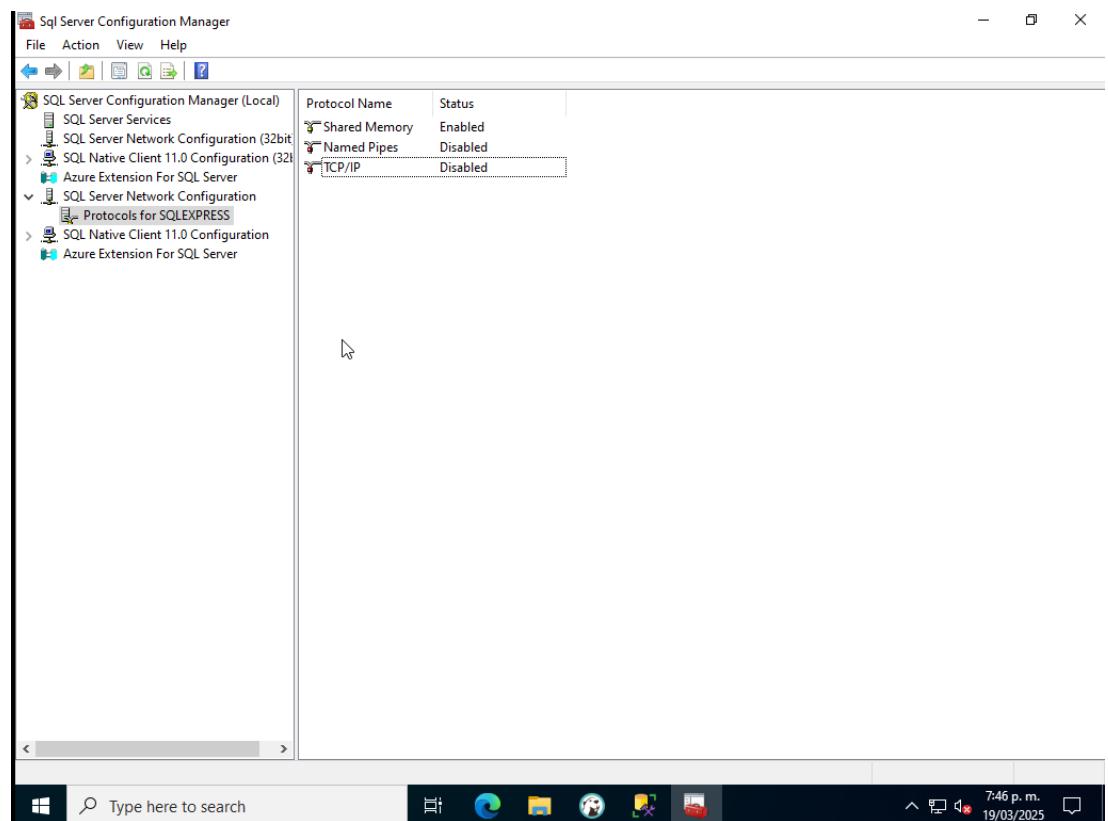


- iv. Ya permitimos las conexiones a la base de datos, ahora abrimos el sscm

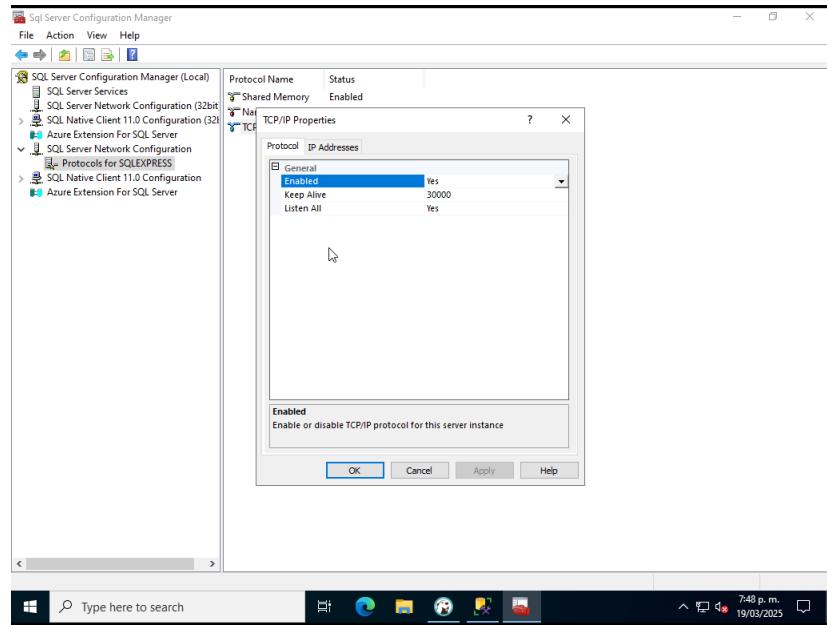


v. Buscamos el apartado de protocolos con el fin de activar los protocolos

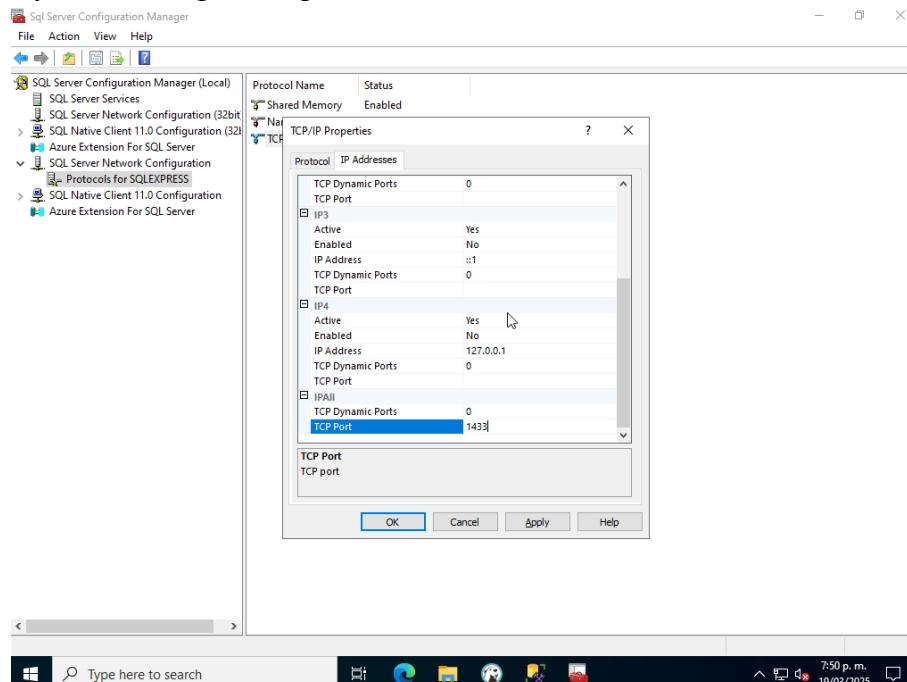
TCP/IP



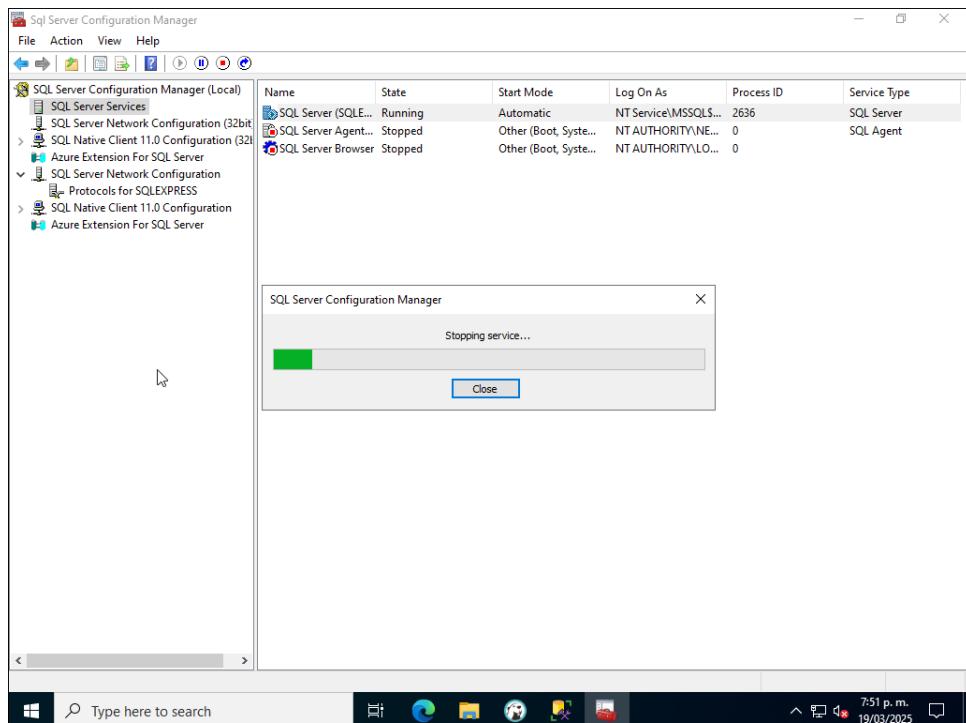
- vi. Activamos TCP/IP dandole click derecho , properties y seleccionando la opcion enabled



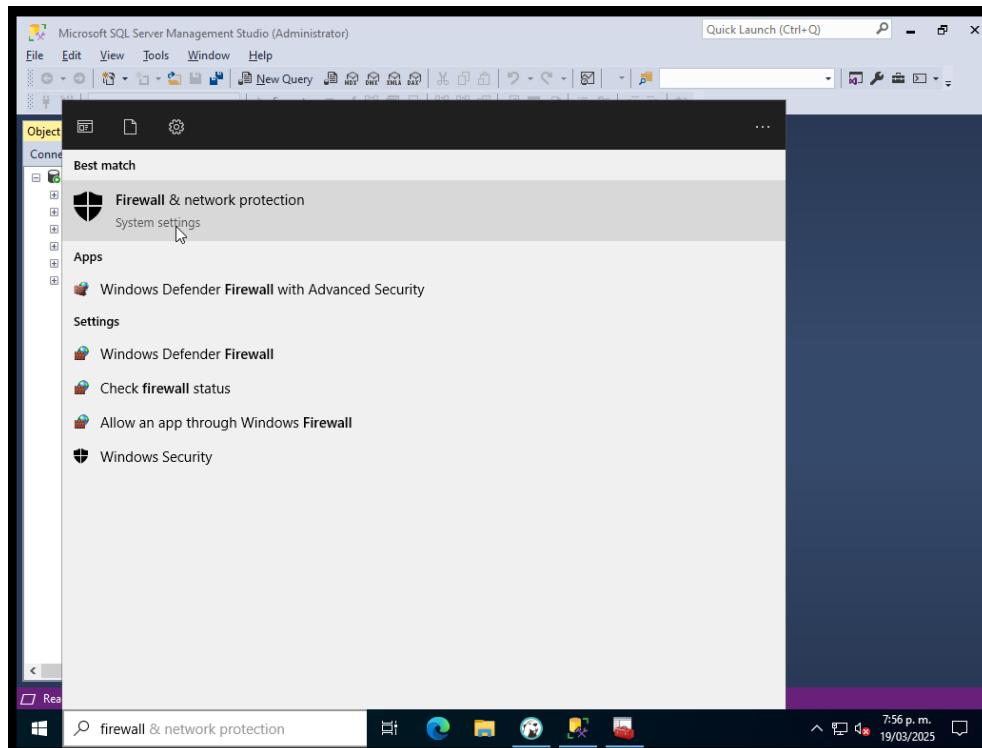
- vii. Luego, vamos a la ventana "Direcciones IP" y, en la opción "Puertos TCP Dynamics", ingrese el puerto de SQL Server: 1433.



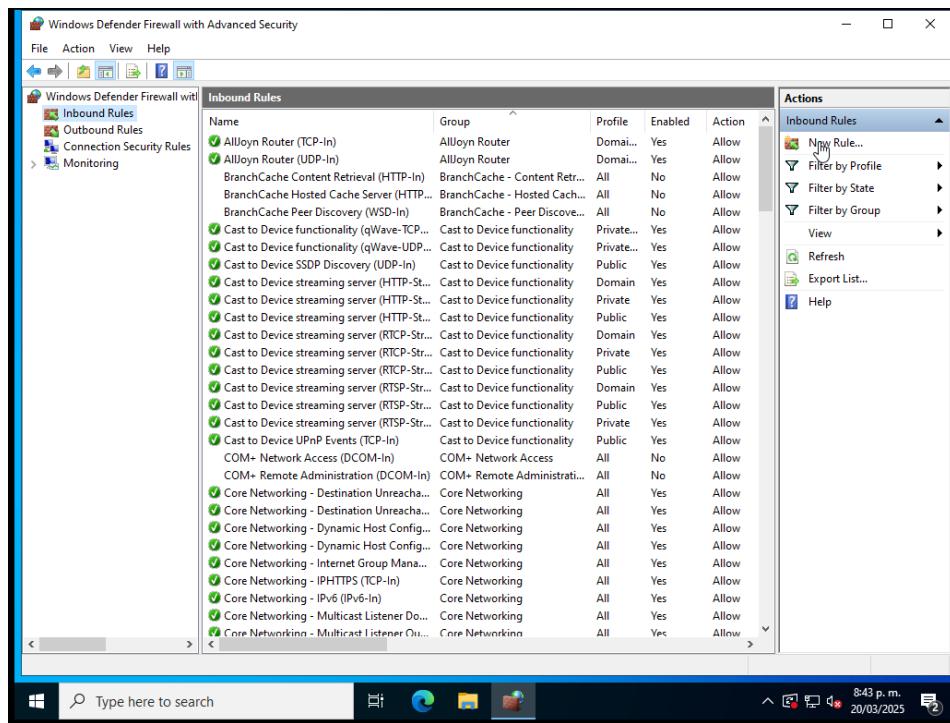
- viii. Reiniciamos el servicio para que queden efectuados los cambios



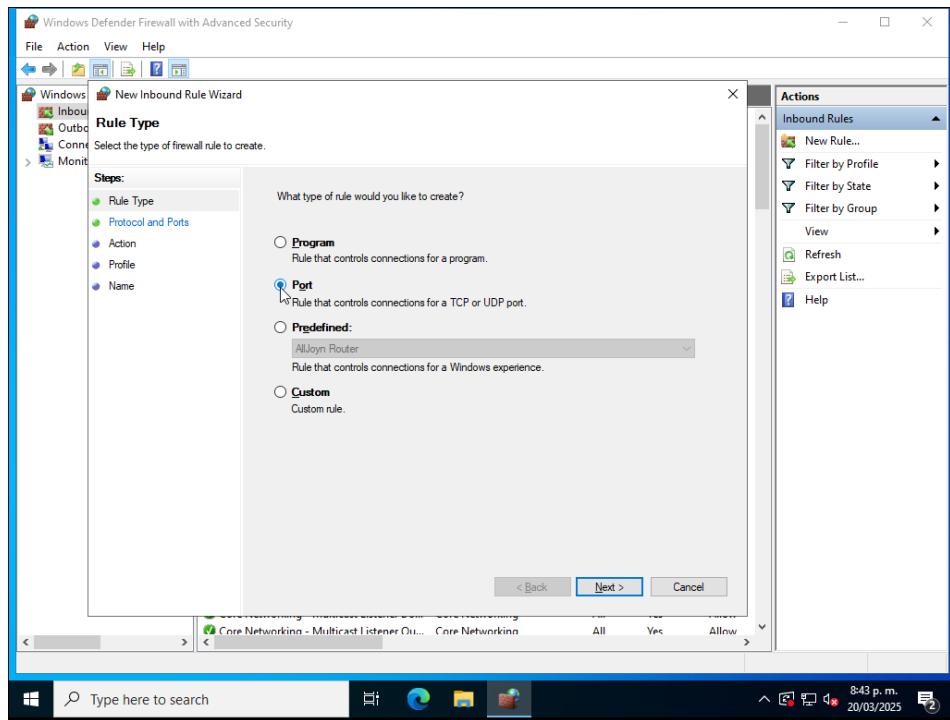
- ix. Ahora vamos a agregar el puerto para SQL server en el firewall. Abrimos el firewall



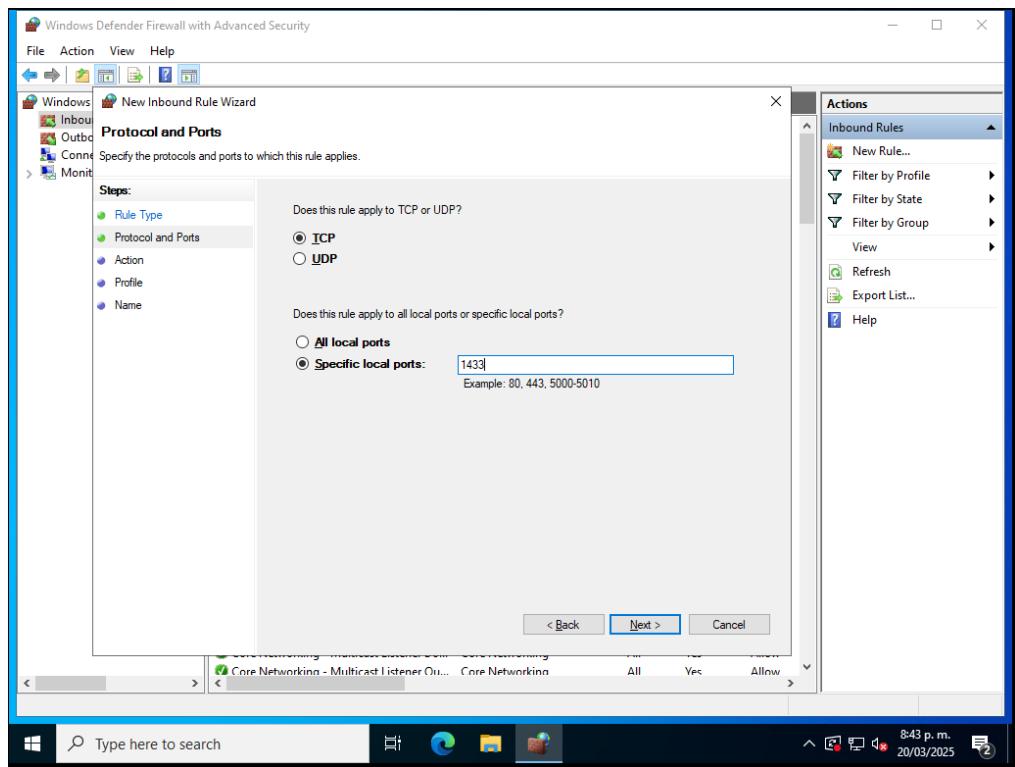
- x. Seleccionamos Inbound Rules y creamos una nueva regla en New Rule



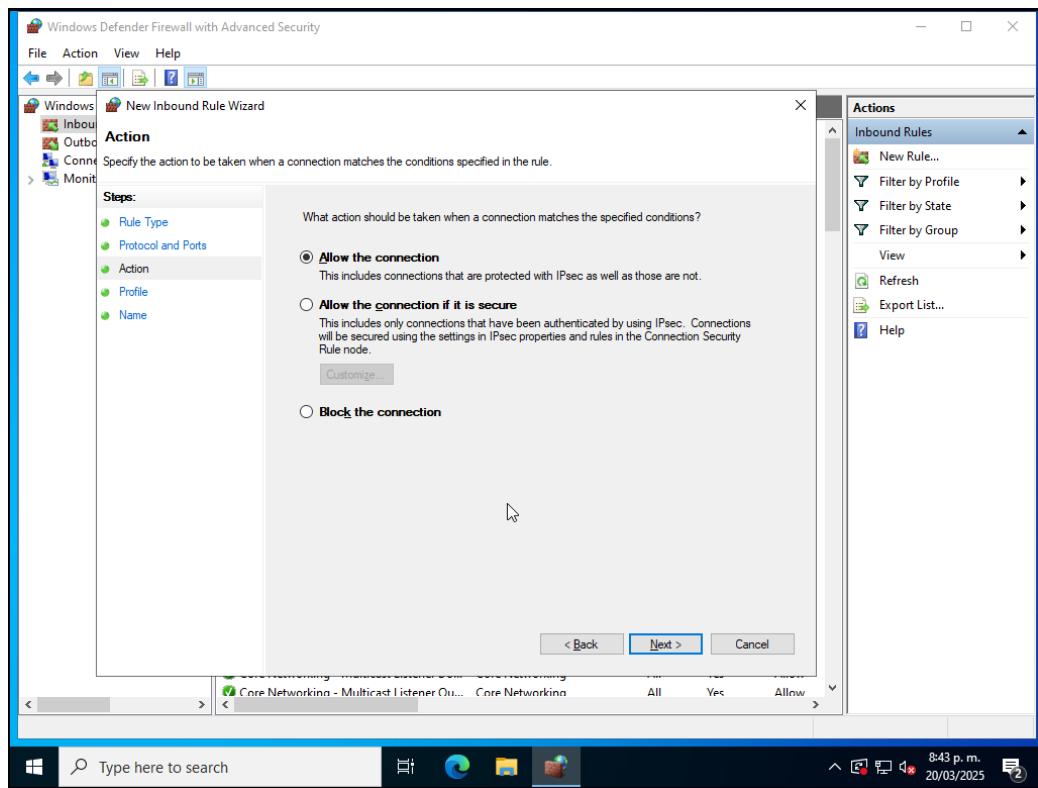
xi. Seleccionamos Port



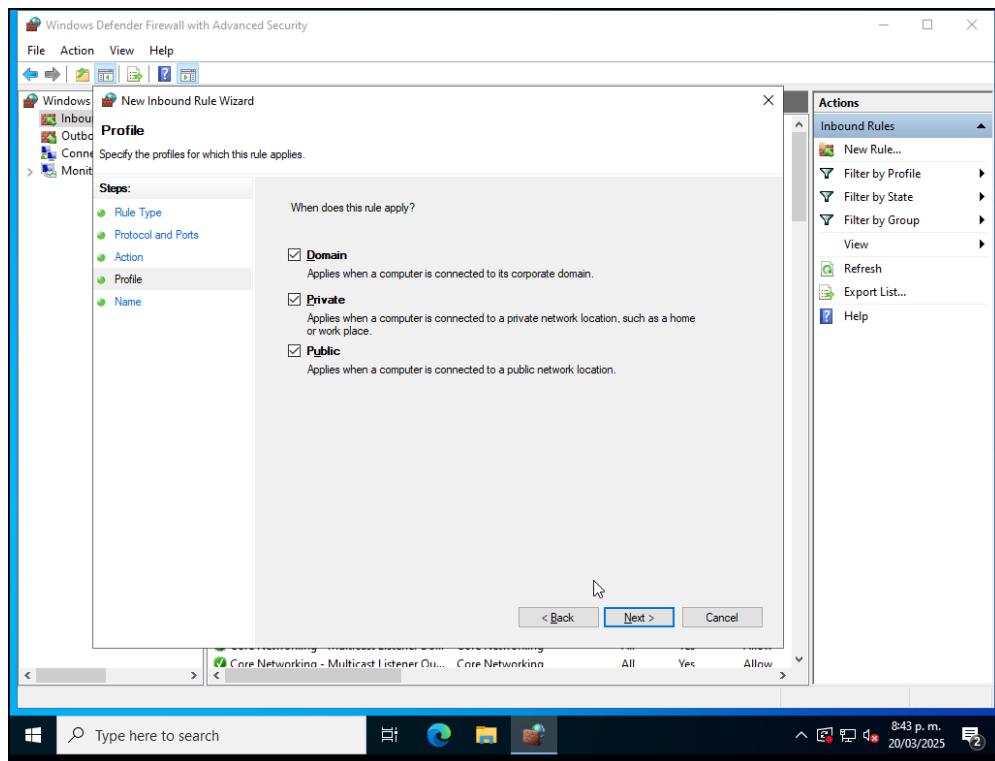
xii. Colocamos el puerto 1433



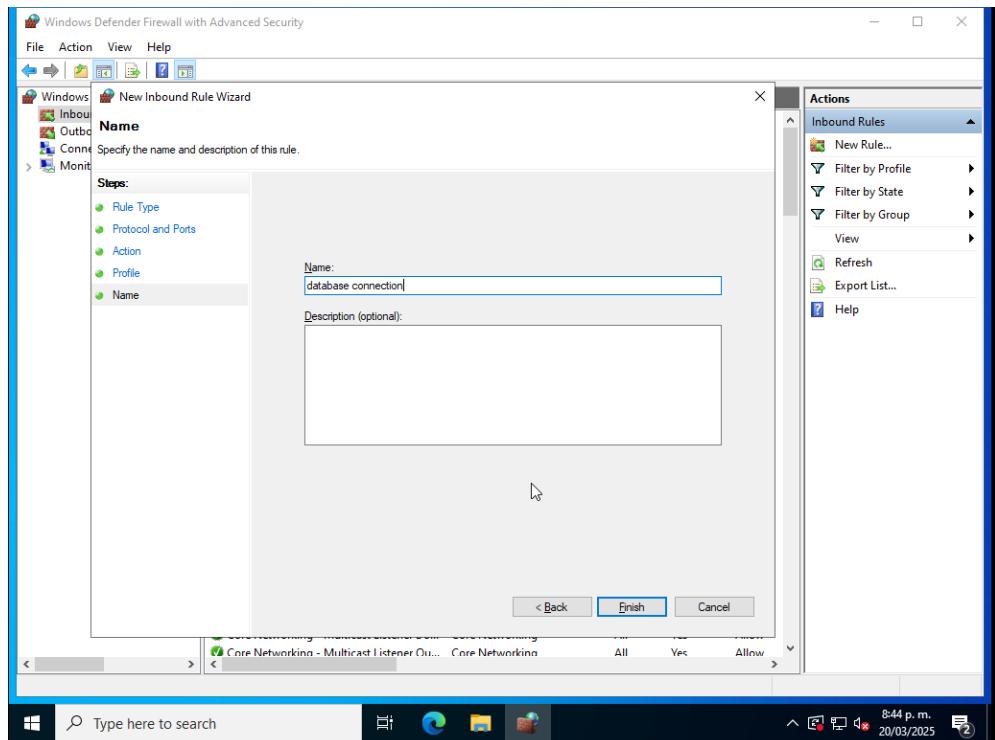
xiii. Seleccionamos Allow the connection



xiv. Seleccionamos todas las opciones



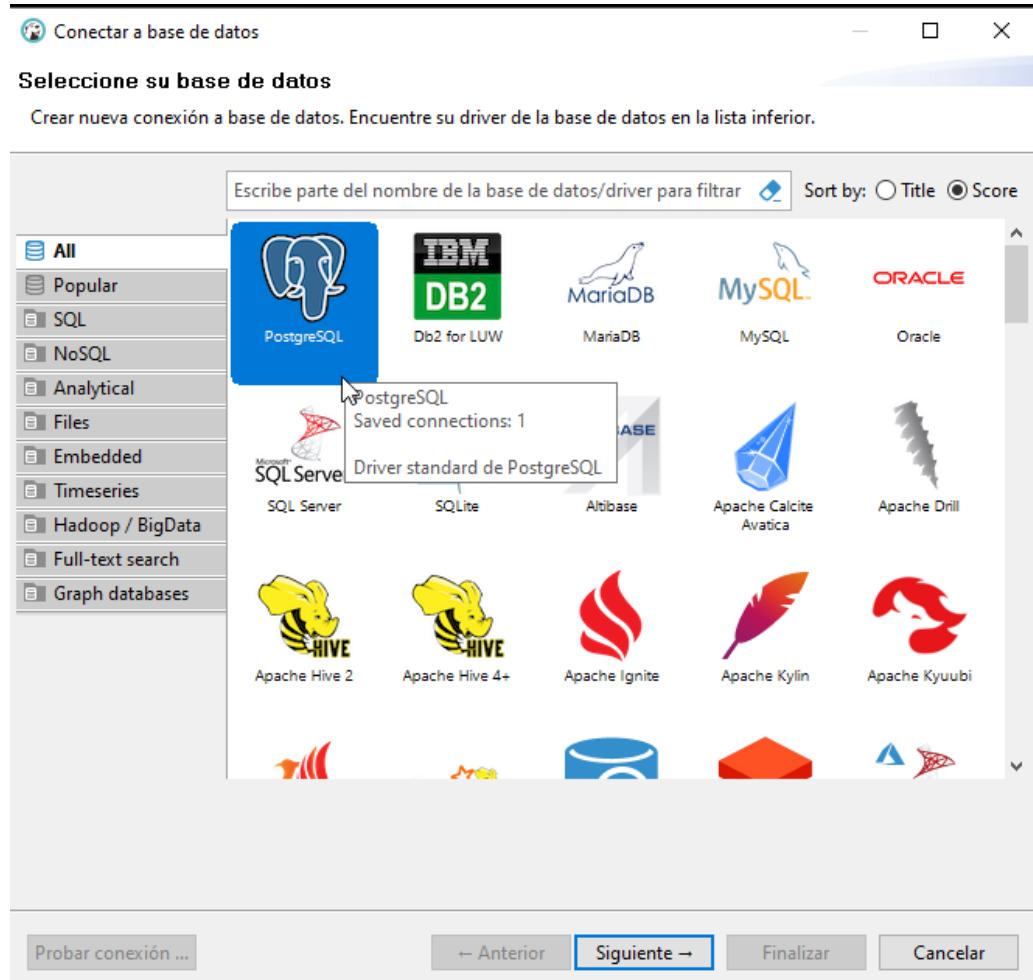
- xv. Finalmente le damos el nombre a la regla, en este caso le pondremos database connection



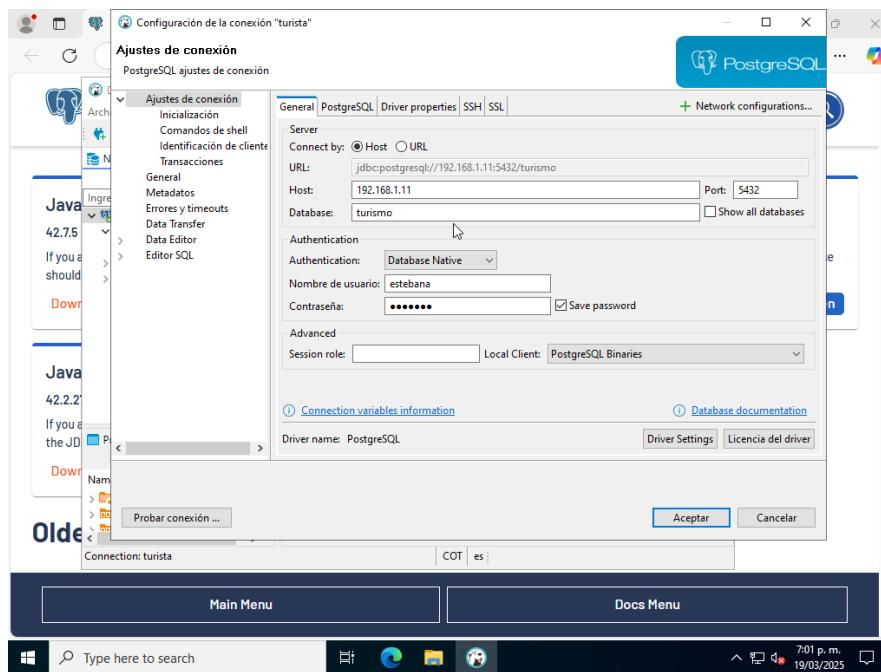
- xvi. Con esto ya quedo configurado el firewall para la base de datos

c. Pruebas

- i. Abrimos la aplicación DBeaver
- ii. Seleccionamos la base de datos a la cual nos queremos conectar, en este caso postgresql



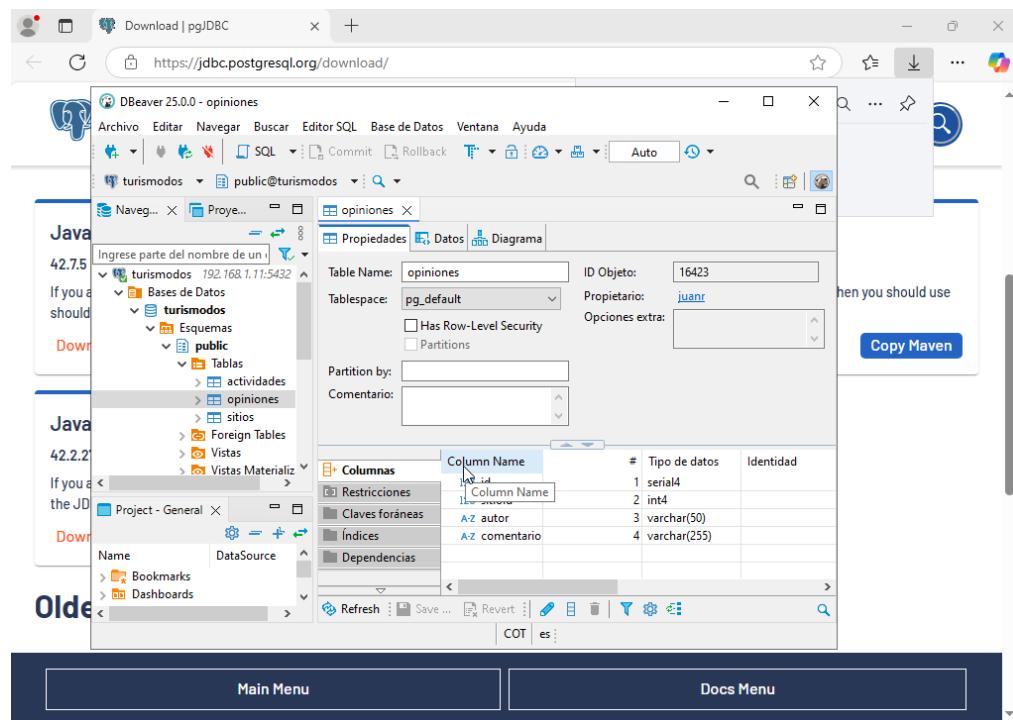
- iii. Ingresamos los datos de la base de datos, en este caso turismo es de esteban en la base de datos de Slackware la cual fue configurada anteriormente



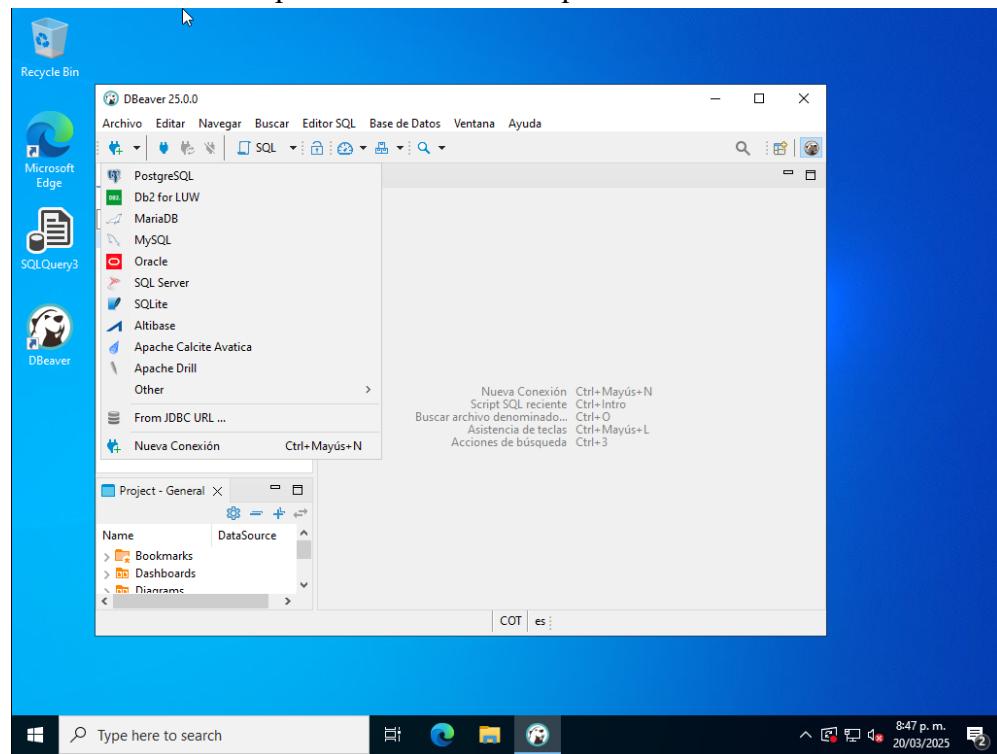
- iv. Damos click en aceptar y ya podemos ver las 3 tablas creadas en slackware y sus respectivos datos

| | id | sitioid | nombre | tipo |
|---|----|---------|----------|--------|
| 1 | 1 | 1 | nombre 1 | tipo 1 |
| 2 | 2 | 2 | nombre 2 | tipo 2 |
| 3 | 3 | 3 | nombre 3 | tipo 3 |

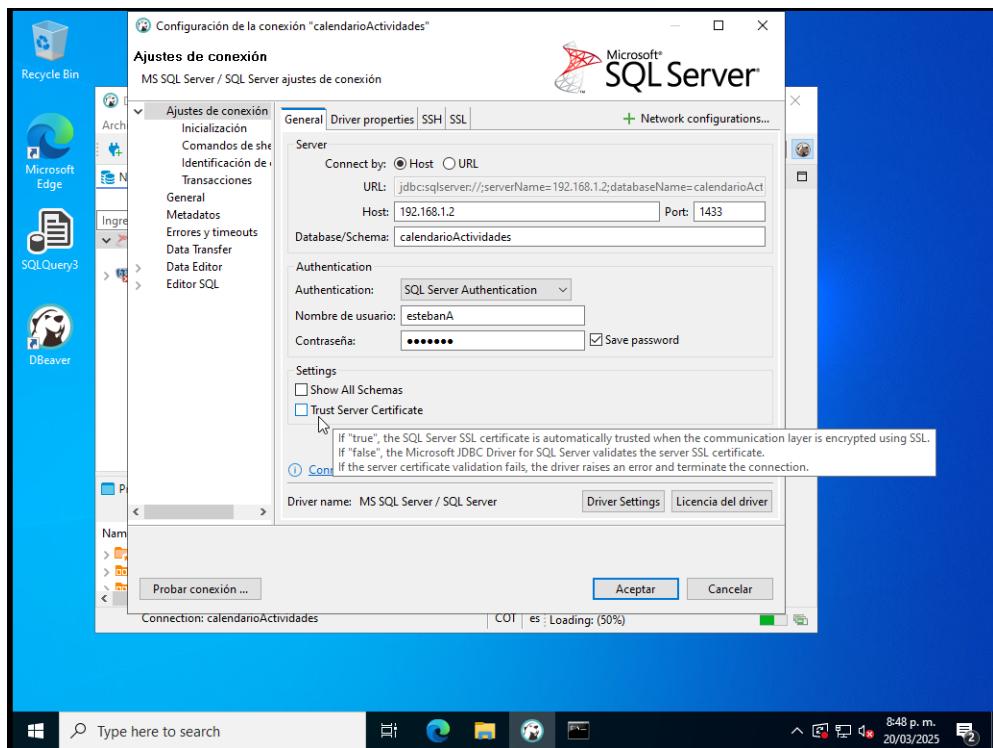
- v. Se repite el proceso con Juan y también se pueden ver las 3 tablas con los datos



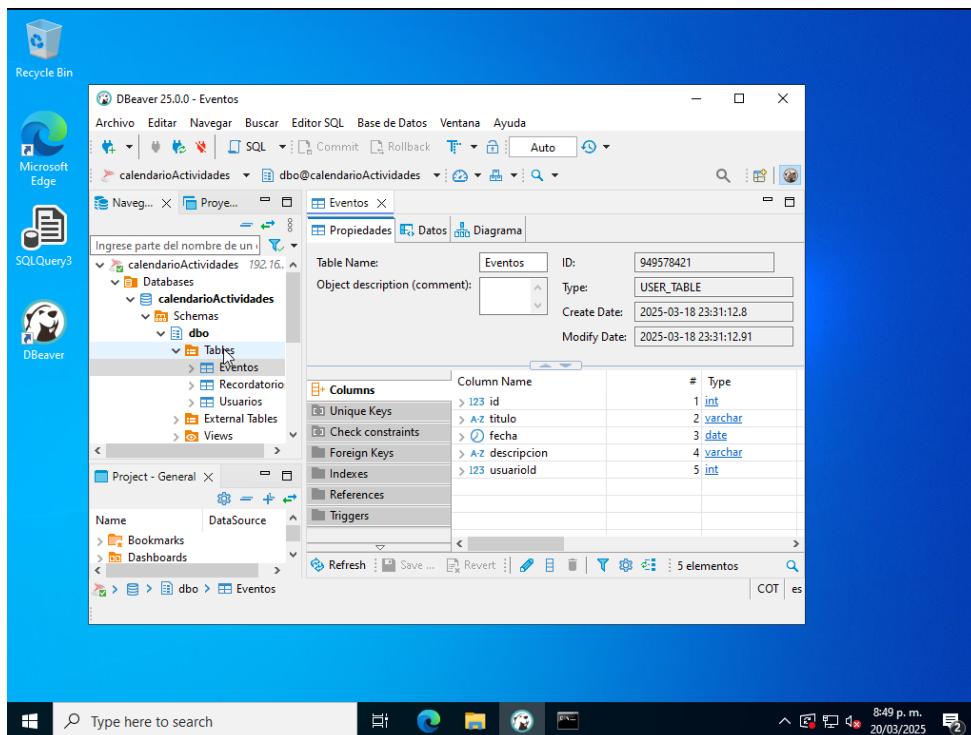
- vi. Ya probadas las de slackware ahora procedemos con Windows. Agregamos una nueva conexión pero en este caso es Sql Server



- vii. Ingresamos los datos de la base de datos, en este caso calendarioActividades (base de datos estebanA)



- viii. Damos click en aceptar y ahora podemos ver la estructura de la base de datos de EstebanA



- ix. Repetimos el mismo proceso para JuanR y miramos la estructura de la base de datos para verificar que conecto correctamente

The image displays two windows side-by-side on a Windows desktop. The top window is titled "Configuración de la conexión 'calendarioActividadesdos'" and shows the "Ajustes de conexión" (Connection Settings) dialog for "MS SQL Server / SQL Server ajustes de conexión". It includes tabs for General, Driver properties, SSH, and SSL. The General tab shows the server URL as "jdbc:sqlserver://serverName=192.168.1.2;databaseName=calendarioActividadesdos", host as "192.168.1.2", port as "1433", authentication as "SQL Server Authentication", user as "juanR", and password as "*****". The bottom window is titled "DBeaver 25.0.0 - Recordatorios" and shows the "Propiedades" (Properties) tab for the "Recordatorios" table. The table has an ID of 1301579675, type USER_TABLE, and was created on 2025-03-19 16:41:25.72. It contains four columns: id (int), evento_id (int), fechaHora (datetime), and mensaje (varchar). The DBeaver interface also shows a navigation tree on the left and various toolbars at the top.

CONCLUSIONS

- Mediante el uso de Wireshark, analizamos en detalle mensajes DNS, HTTP y Ethernet, lo que nos permitió comprender mejor cómo se comunican los dispositivos en una red y cómo se lleva a cabo la resolución de nombres de dominio.
- Identificamos elementos clave en los encabezados de estos protocolos, como direcciones IP, métodos de solicitud y tipos de contenido, lo que nos ayudó a entender el flujo de datos en una comunicación de red.
- La instalación y configuración de sistemas de gestión de bases de datos, tanto en entornos locales como en la nube, nos permitió conocer en profundidad el proceso de administración de bases de datos.
- Implementamos PostgreSQL y SQL Server en máquinas virtuales, creando usuarios y bases de datos específicas, lo que reforzó nuestra comprensión sobre el manejo de datos estructurados y el control de acceso en entornos colaborativos.
- Al trabajar con Microsoft Azure, exploramos las ventajas de la computación en la nube en comparación con las infraestructuras tradicionales y analizamos la importancia de la escalabilidad en la gestión eficiente de recursos.
- Este laboratorio fortaleció el conocimiento teórico sobre redes y bases de datos. Además nos proporcionó experiencia práctica en la configuración y administración de infraestructura TI. Aprendimos a instalar, gestionar y monitorear servicios en distintos entornos, entendiendo la importancia de una configuración adecuada para garantizar el rendimiento y la seguridad de los sistemas.