

# **LABORATORY 05 - DATABASES AND NETWORK PROTOCOLS - CLOUD**

Laboratory 5 corresponding to the second term

Santiago Amaya Zapata

Julian Camilo Tinjacá Corredor



Escuela Colombiana de Ingeniería Julio Garavito

NETWORK ARCHITECTURE AND SERVICES

GROUP 3

Bogotá DC, Colombia

2025 - 2

# **Lab No.05 - DATABASES AND NETWORK PROTOCOLS - CLOUD**

## **Objectives**

---

Continue learning the installation of basic software and the operation of network protocols.  
Install and configure base software - Web Servers.

## **Tools to be used**

---

- Computers
- Internet access
- Virtualization software
- Wireshark

## **Introduction**

---

In this laboratory, we focused on deepening our understanding of how fundamental IT infrastructure components are installed, configured, and integrated to support real-world business operations. Specifically, the exercise centered on the installation and configuration of the PostgreSQL Database Management System (DBMS) on a Linux Slackware virtual machine. This activity simulated a common enterprise environment, where physical and virtual servers coexist within complex network topologies, including wired and wireless connections managed through switches, routers, and other network devices. Our goal was to replicate the process an organization would follow to deploy a functional and secure database server capable of storing and organizing structured information for future use by various applications and services.

Through this lab, we continued to gain practical experience in managing not only software installation but also user configuration, database creation, and data insertion. We created personalized user accounts, ensuring each had access only to their respective databases, reinforcing the principle of access control and data isolation in multi-user environments. The databases designed in this activity focused on cataloging tourist sites in Colombia, which allowed us to apply database modeling concepts such as relational table design, foreign key constraints, and referential integrity. By following a structured installation and configuration process, we were able to understand how PostgreSQL interacts with the underlying operating system, manages data directories, and provides secure access through authentication mechanisms.

Moreover, this lab emphasized the importance of understanding how network protocols interact with database systems. In a corporate infrastructure, databases rarely operate in isolation; they are constantly communicating with other systems and services over the network. Therefore, learning about DNS, HTTP, and Ethernet protocols using Wireshark allowed us to observe the flow of communication between clients and servers in real time. This understanding bridges the gap between theoretical networking knowledge and practical system administration skills. In essence, the lab helped us strengthen our technical foundation for both database management and network communication—two pillars of modern information technology environments that ensure data accessibility, consistency, and reliability across systems.

## Theoretical Framework

---

A Database Management System (DBMS) is specialized software designed to handle the definition, storage, retrieval, and management of data in a structured and efficient manner. PostgreSQL, in particular, is an advanced open-source relational DBMS that adheres to the relational model proposed by E.F. Codd. It supports the use of SQL (Structured Query Language) to define and manipulate data, ensuring consistency and integrity across large datasets. PostgreSQL's architecture allows multiple users to access the same data simultaneously through robust concurrency control mechanisms, while maintaining data accuracy and preventing conflicts. Its features, including role-based access control, transaction management, referential integrity, and strong data typing, make it a preferred choice for both academic and professional environments.

Within an enterprise infrastructure, the DBMS plays a crucial role in centralizing information and making it accessible to authorized users and applications. It acts as the backbone for various business operations such as customer management, inventory control, and financial reporting. Security and access control are fundamental aspects of database management, as they ensure that sensitive information remains protected against unauthorized access. PostgreSQL allows system administrators to define users, assign privileges, and enforce authentication through password-based or external verification mechanisms.

At the same time, databases are not isolated systems—they function within a larger network ecosystem that relies on communication protocols. Protocols such as DNS (Domain Name System), HTTP (Hypertext Transfer Protocol), and Ethernet govern how data travels between devices, ensuring it reaches the correct destination. DNS resolves human-readable names into IP addresses, HTTP facilitates the exchange of data between clients and web servers, and Ethernet defines the physical and data link layers for network communication. Using tools like Wireshark to analyze these protocols provides valuable insights into how applications and databases interact across the network, revealing message structures, request-response cycles, and latency factors.

Therefore, understanding both database management and network protocols is essential for system administrators and network engineers. The synergy between these two areas allows organizations to deploy reliable, scalable, and secure infrastructures capable of supporting the growing demand for data-driven services. Through this lab, we reinforced this integration, gaining practical experience that reflects the interconnected nature of modern computing environments.

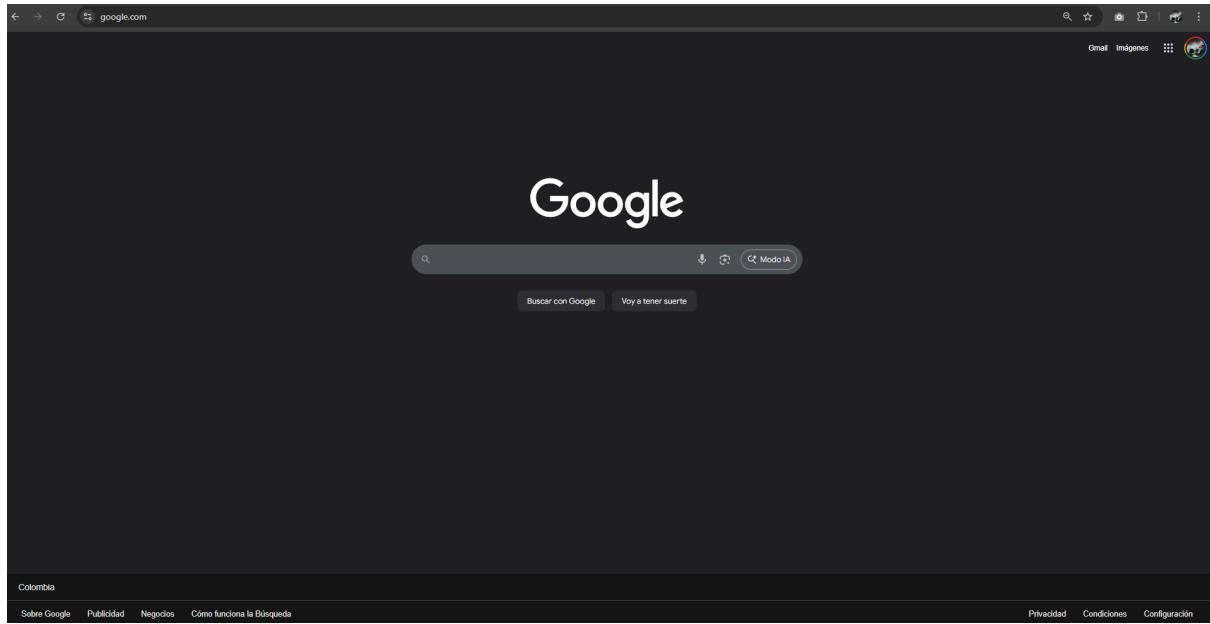
## Review of Network Protocols

---

### 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.

The first thing we need to do is make the query to the webpage <http://www.google.com/>



Then we have to start the packaging capture and after that we apply the dns filter to get all the queries about our request

The screenshot shows two separate Wireshark captures. The top capture, titled "Ethernet 3", displays a list of QUIC protocol packets. The bottom capture, titled "dns", displays a list of DNS protocol packets. Both captures include columns for No., Time, Source, Destination, Protocol, and Length Info.

No.	Time	Source	Destination	Protocol	Length Info
1250	30.716659	192.168.1.4	172.217.30.206	QUIC	120 Handshake, DCID=e9d39d8ed50a1de1
1251	30.716713	192.168.1.4	172.217.30.206	QUIC	73 Protected Payload (KP0), DCID=e9d39d8ed50a1de1
1252	30.716892	192.168.1.4	172.217.30.206	QUIC	488 Protected Payload (KP0), DCID=e9d39d8ed50a1de1
1253	30.722945	172.217.30.206	192.168.1.4	QUIC	162 Protected Payload (KP0)
1254	30.722945	172.217.30.206	192.168.1.4	QUIC	69 Protected Payload (KP0)
1255	30.723051	192.168.1.4	172.217.30.206	QUIC	73 Protected Payload (KP0), DCID=e9d39d8ed50a1de1
1256	30.740099	192.168.1.4	172.217.30.196	UDP	71 63622 → 443 Len=29
1257	30.746885	172.217.30.196	192.168.1.4	UDP	68 443 → 63622 Len=26
1258	30.792594	172.217.30.206	192.168.1.4	QUIC	262 Protected Payload (KP0)
1259	30.792594	172.217.30.206	192.168.1.4	QUIC	329 Protected Payload (KP0)
1260	30.792777	192.168.1.4	172.217.30.206	QUIC	77 Protected Payload (KP0), DCID=e9d39d8ed50a1de1
1261	30.803282	192.168.1.4	172.217.30.174	UDP	71 60799 → 443 Len=29
1262	30.810693	172.217.30.174	192.168.1.4	UDP	67 443 → 60799 Len=25
1263	30.824279	172.217.30.206	192.168.1.4	QUIC	66 Protected Payload (KP0)
1264	30.962212	192.168.1.4	172.217.30.196	UDP	71 63622 → 443 Len=29
1265	30.966723	172.217.30.196	192.168.1.4	UDP	68 443 → 63622 Len=26
1266	31.184412	192.168.1.4	172.217.30.196	UDP	71 63622 → 443 Len=29
1267	31.191088	172.217.30.196	192.168.1.4	UDP	68 443 → 63622 Len=26
1268	31.311900	192.168.1.4	172.217.29.10	UDP	71 56269 → 443 Len=29
1269	31.319360	172.217.29.10	192.168.1.4	UDP	67 443 → 56269 Len=25
1270	31.391588	192.168.1.4	172.217.30.196	UDP	71 63622 → 443 Len=29
1271	31.398277	172.217.30.196	192.168.1.4	UDP	68 443 → 63622 Len=26

No.	Time	Source	Destination	Protocol	Length Info
701	16.886073	fe80::1	fe80::c07a:7127:b7c.. DNS	DNS	124 Standard query response 0x419c a ogads-pa.clients6.google.com A 172.217.162.106
702	16.869513	fe80::1	fe80::c07a:7127:b7c.. DNS	DNS	122 Standard query response 0x4587 A waa-pa.clients6.google.com A 172.217.162.106
703	16.874655	fe80::1	fe80::c07a:7127:b7c.. DNS	DNS	158 Standard query response 0x0e07 HTTPS ogads-pa.clients6.google.com SOA ns1.google.com
704	16.875912	fe80::1	fe80::c07a:7127:b7c.. DNS	DNS	156 Standard query response 0x344f HTTPS waa-pa.clients6.google.com SOA ns1.google.com
712	16.881367	fe80::1	fe80::c07a:7127:b7c.. DNS	DNS	158 Standard query response 0x0e07 HTTPS ogads-pa.clients6.google.com SOA ns1.google.com
713	16.881367	fe80::1	fe80::c07a:7127:b7c.. DNS	DNS	156 Standard query response 0x344f HTTPS waa-pa.clients6.google.com SOA ns1.google.com
783	16.998606	fe80::c07a:7127:b7c.. fe80::1 DNS	DNS	99 Standard query 0xe0e7 HTTPS accounts.google.com	
784	16.998737	fe80::c07a:7127:b7c.. fe80::1 DNS	DNS	99 Standard query 0xe667b A accounts.google.com	

Once we got all the packages displayed we have to click on one the number 784

The screenshot shows the details of the selected DNS packet (Frame 784). The packet is a standard query for 'accounts.google.com' type A. The details view shows the transaction ID (0x6667b), flags (0x0100 Standard query), and various fields such as Response, Message is a query, Opcode, Truncated, Recursion desired, Z: reserved, and Non-authenticated data. The questions section shows the query for 'accounts.google.com' type A. The answers section shows the response from 'bl.nel.goog' with the IP address 172.217.29.13. The authority and additional sections are empty. The bytes view at the bottom shows the raw hex and ASCII data of the packet.

In this part of the capture, we are looking at the details for the **Domain Name System (query)**. This is the packet your computer sent to *ask* for an address.

- **Transaction ID:** 0x667b

**Explanation:** This is a unique identification number where the PC attaches it to the query so it can match the incoming answer to this specific question.

- **Flags:** Details shown when I expanded the option

**Explanation:** The most important field here is Recursion desired: Do query recursively (1). Here is where the pc tells the DNS server, if it doesn't know the answer, find it for him

- **Questions:** 1

**Explanation:** This simply indicates that this packet contains 1 question.

- **Queries:** This section contains the actual question itself

**Name:** [accounts.google.com](http://accounts.google.com)

**Explanation:** The exact domain name the pc wants to resolve.

- **Type:** A (Address)

**Explanation:** This indicates we are asking for an **IPv4 address**.

**Class:** IN (Internet)

**Explanation:** This is a standard field that almost always means "Internet."

Now when we click on the number 786

Frame 786: 115 bytes on wire (920 bits), 115 bytes captured (920 bits) on interface \Device\NPF\_{7455CE38-3D5F-4510-9696-CEF206C4E685}, id 0

Ethernet II, Src: zte\_98:b6:80 (08:f6:06:98:b6:80), Dst: ASRockIncorp\_95:79:08 (70:85:c2:95:79:08)

Internet Protocol Version 6, Src: fe80::1, Dst: fe80::c07a:7127:b7c5:4f22

User Datagram Protocol, Src Port: 53, Dst Port: 61343

Domain Name System (response)

Transaction ID: 0x667b

Flags: 0x8180 Standard query response, No error

- 1. .... .... = Response: Message is a response
- .000 0.... .... = Opcode: Standard query (0)
- .... .0.... .... = Authoritative: Server is not an authority for domain
- .... .0. .... .... = Truncated: Message is not truncated
- .... .0.. .... .... = Recursion desired: Do query recursively
- .... ...1 .... .... = Recursion available: Server can do recursive queries
- .... ...0. .... .... = Z: reserved (0)
- .... ...0. .... .... = Answer authenticated: Answer/authority portion was not authenticated by the server
- .... ...0.. .... .... = Non-authenticated data: Unacceptable
- .... .... ..0000 = Reply code: No error (0)

Questions: 1

Answer RRs: 1

Authority RRs: 0

Additional RRs: 0

Queries

- accounts.google.com: type A, class IN
  - Name: accounts.google.com
  - [Name Length: 19]
  - [Label Count: 3]
  - Type: A (1) (Host Address)
  - Class: IN (0x0001)

Answers

- accounts.google.com: type A, class IN, addr 172.217.204.84
  - Name: accounts.google.com
  - Type: A (1) (Host Address)
  - Class: IN (0x0001)
  - Time to live: 231 (3 minutes, 51 seconds)
  - Data length: 4
  - Address: 172.217.204.84

[Request Id: 784]  
[Time: 0.001606000 seconds]

0000 70 85 c2 95 79 08 08 f6 06 98 b6 80 86 dd 60 00 p y .....  
0010 00 00 00 3d 11 40 fe 80 00 00 00 00 00 00 00 00 00 .. =@ .....  
0020 00 00 00 00 00 01 fe 80 00 00 00 00 00 00 c0 7a ..... z  
0030 71 27 b7 c5 4f 22 00 35 ef 9f 00 3d 55 19 66 7b q "O" 5 ...=U f{  
0040 81 80 00 01 00 01 00 00 00 00 08 61 63 63 6f 75 accou  
0050 6e 74 73 06 67 6f 6f 67 6c 65 03 63 6f 6d 00 00 nts goog le.com  
0060 01 00 01 c0 00 01 00 01 00 00 00 e7 00 04 ac .....  
0070 d9 cc 54 .. T

Now we are looking at the **Domain Name System (response)** packet, this is the answer the server sent back to the computer.

- **Transaction ID:** 0x667b

**Explanation:** We notice this is **identical** to the query's ID because this is how the PC knows this is the answer to the question it asked in packet 784.

- **Flags:**

**Explanation:** we will see new flags, like Message is a response confirming it's an answer and Recursion available, and this is how the server says that he can perform recursive Lookups

- **Questions:** 1

**Explanation:** The server repeats the original question so we know what it is responding to.

- **Answers:**

**Explanation:** Here is the "name resolution" we were looking for it says accounts.google.com: type A..

**Name:** accounts.google.com This is the name that was resolved before

**Type:** A (The type of record being returned).

**Time to live (TTL):** 231 is the time in seconds

**Explanation:** This tells the PC how long it can store this answer. If we need this IP again in the next 180 seconds, it won't have to ask the server again.

**Data length:** 4

**Explanation:** The length of the IP address (an IPv4 address is 4 bytes).

**Address:** 172.217.104.84

**Explanation: This is the answer!** It is the IP address your browser will use to connect to the accounts.google.com server.

## 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.

Here we do almost the same process , but we use http instead of dns filter

No.	Time	Source	Destination	Protocol	Length	Info
41	2.975574	192.168.1.4	45.239.88.86	HTTP	993	GET /~csantiago/ HTTP/1.1
52	2.984594	45.239.88.86	192.168.1.4	HTTP	492	HTTP/1.1 200 OK (text/html)

Frame 41: 993 bytes on wire (7944 bits), 993 bytes captured (7944 bits) on interface \Device\NPF\_{7455CE38-3D5F-4510-9696-CEF206C4E68}  
Ethernet II, Src: ASRockIncorp\_95:79:08 (70:85:c2:95:79:08), Dst: zte\_98:b6:80 (08:f6:06:98:b6:80)  
Internet Protocol Version 4, Src: 192.168.1.4, Dst: 45.239.88.86  
Transmission Control Protocol, Src Port: 54007, Dst Port: 80, Seq: 1, Ack: 1, Len: 939  
Hypertext Transfer Protocol

Then we click on the first package

- #### - Request Method: GET

**Explanation:** This defines the action the client wants the server to perform,

GET method is used to retrieve a resource, such as an HTML file or an image

- Request URI: /~csantiago/

**Explanation:** The Uniform Resource Identifier (URI). This specifies the path to the exact resource the client is requesting on the server.

- **Request Version:** HTTP/1.1

**Explanation:** This indicates the version of the HTTP protocol being used for the communication, then both the client and server must support this version to understand each other.

- **Host:** [profesores.is.escuelaing.edu.co](http://profesores.is.escuelaing.edu.co)

**Explanation:** This header is essential. It specifies the domain name of the server the client wants to talk to so this allows a single server to manage and serve content for many different websites.

- **User-Agent:** Mozilla/5.0 (Windows NT 10.0; Win64; x64) ...

**Explanation:** A string that identifies the client software, it provides details

about the browser and the operating system, then some servers can use this information to send content optimized for that specific software.

- **Accept:** text/html,application/xhtml+xml,application/xml;q=0.9,...

**Explanation:** This field lists the media types the client can understand, often with a quality value to indicate preference. Here the client is saying it prefers text/html above other formats.

- **Accept-Language:** es-ES,es;q=0.9

**Explanation:** This tells the server the client's preferred languages, in order of priority, then the server can use this to send a version of the page in that language, if it is available, so in this case, the preference is "Spanish from Spain" (es-ES) and then any other Spanish (es).

- **Accept-Encoding:** gzip, deflate

**Explanation:** This informs the server about the compression methods the client can handle, if the server supports it, it can "zip" the content before sending it, reducing the file size and speeding up the transfer. The client then decompresses it

- **Cookie:** gcl\_au=...; \_ga=...;

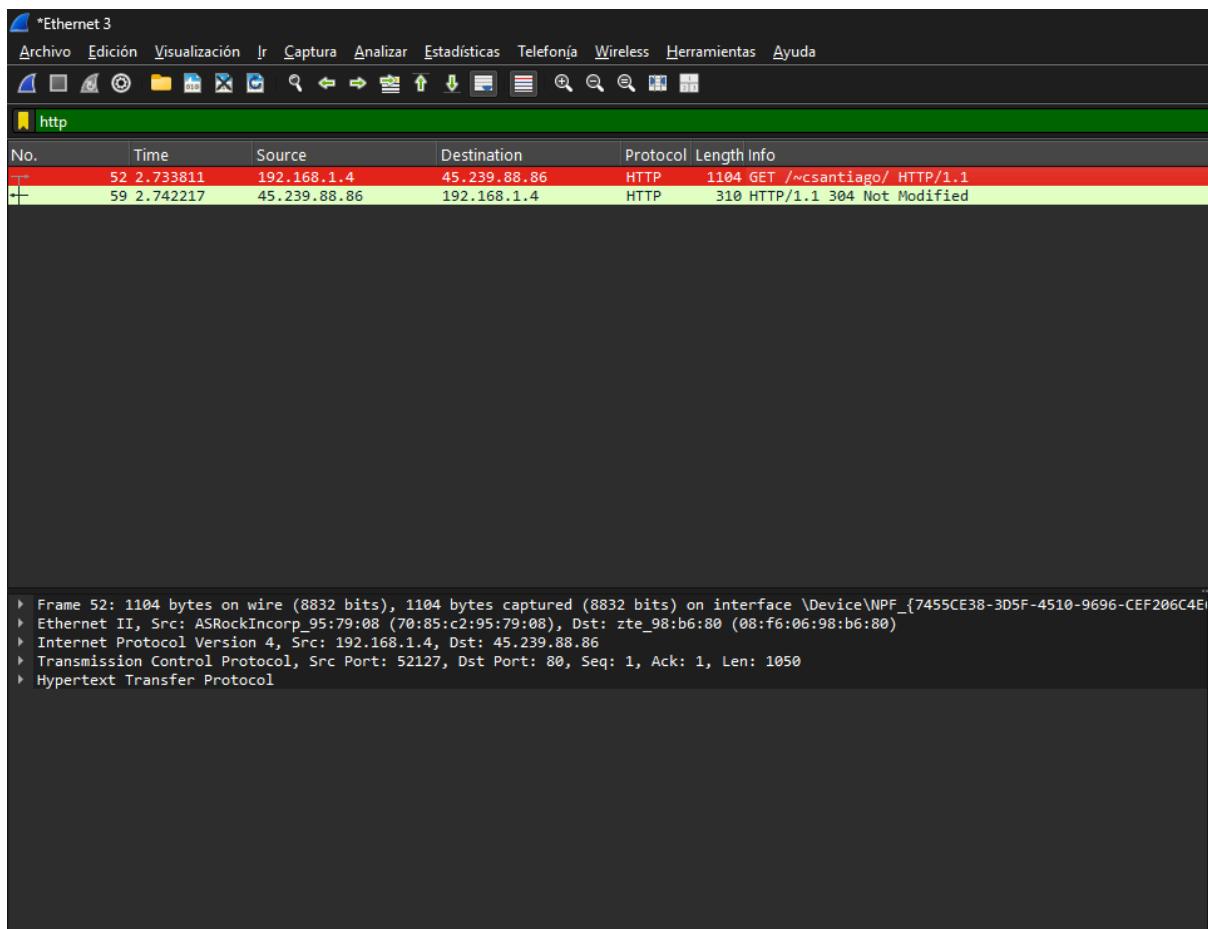
**Explanation:** This header contains data (like session IDs or tracking information) that the server has previously stored on the client. The client sends this data back with each request so the server can "remember" the client's state or identify it.

### 3. Review of Ethernet Frames

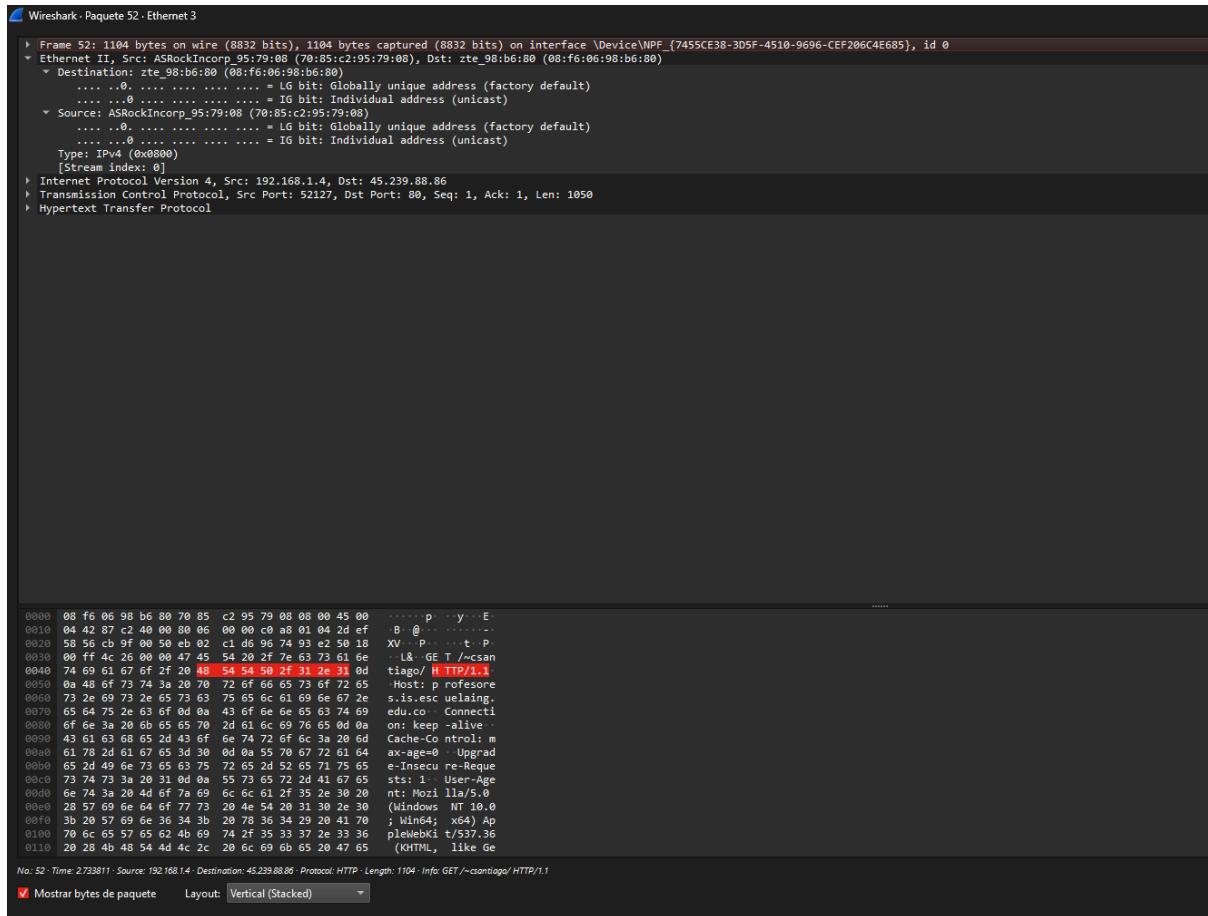
Using Wireshark, make a query to the webpage

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

Same as the previous step we still set the http filter



Then we click on the get,



- **Destination:** zte\_98:b6:80 (00:f6:06:98:b6:80)

**Explanation:** This is the destination MAC address. It is the physical hardware address of the network card that this frame is being sent to on the local network.

**IG bit: Globally unique address:** This indicates the address is a standard, manufacturer-assigned MAC address.

**IG bit: Individual address:** This indicates the frame is being sent to a single, specific device and is not a broadcast.

- **Source:** ASROCKIncorp\_95:79:08 (70:85:c2:95:79:08)

**Explanation:** This is the **source MAC address**. It is the physical hardware address of the network card that this frame is being sent from.

- **Type:** IPv4 (0x0800)

**Explanation:** This field identifies the protocol of the data inside the Ethernet frame. The code 0x0800 signals that the payload is an Internet Protocol

version 4 (IPv4) packet. This tells the receiving device to pass the data up to the network layer for IP processing.

## Base Software Installation

---

Another key component of basic IT infrastructure is database management systems (DBMS). These DBMS can be hosted either in a company's datacenter or on a cloud server. They store structured data for the organization and are used by different applications supporting business operations.

### 1. PostgreSQL - Linux Slackware

- Install the PostgreSQL DBMS on a virtual machine running Linux Slackware.

We changed to the `/tmp` directory to use it as a temporary workspace for the PostgreSQL installation files.

```
root@aysrSlack:~# cd /tmp
root@aysrSlack:/tmp#
```

We downloaded the PostgreSQL SlackBuild script from the official SlackBuilds repository to automate the process of compiling and packaging PostgreSQL for Slackware.

```
root@aysrSlack:~# cd /tmp
root@aysrSlack:/tmp# wget https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz
--2025-10-14 10:29:39--  https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz
Resolving slackbuilds.org (slackbuilds.org)... failed: Name or service not known.
wget: unable to resolve host address "slackbuilds.org"
root@aysrSlack:/tmp# wget https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz
--2025-10-14 10:29:55--  https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz
Resolving slackbuilds.org (slackbuilds.org)... 66.85.79.67, 64:ff9b::4255:4f43
Connecting to slackbuilds.org (slackbuilds.org)|66.85.79.67|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6860 (6.7K) [application/x-gzip]
Saving to: "postgresql.tar.gz"

postgresql.tar.gz      100%[=====]  6.70K --.-KB/s   in 0s

2025-10-14 10:29:56 (3.01 GB/s) - "postgresql.tar.gz" saved [6860/6860]
root@aysrSlack:/tmp#
```

We extracted the downloaded archive to access the SlackBuild script and related files needed for the installation.

```
root@aysrSlack:/tmp# tar -xvf postgresql.tar.gz
postgresql/
postgresql/doinst.sh
postgresql/slack-desc
postgresql/README
postgresql/postgresql.logrotate
postgresql/rc.postgresql.new
postgresql/postgresql.info
postgresql/setup.postgresql
postgresql/README.SBo
postgresql/postgresql.SlackBuild
root@aysrSlack:/tmp# cd postgresql
root@aysrSlack:/tmp/postgresql#
```

Now we entered the **postgresql** directory to prepare for the build process and downloaded the official PostgreSQL 14.19 source code from the PostgreSQL website, which the SlackBuild script requires to compile the program.

```
root@aysrSlack:/tmp/postgresql# wget https://ftp.postgresql.org/pub/source/v14.19/postgresql-14.19.tar.gz
--2025-10-14 10:35:39--  https://ftp.postgresql.org/pub/source/v14.19/postgresql-14.19.tar.gz
Resolving ftp.postgresql.org (ftp.postgresql.org)... 151.101.3.52, 151.101.67.52, 151.101.195.52, ...
Connecting to ftp.postgresql.org (ftp.postgresql.org)|151.101.3.52|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 29472149 (28M) [application/gzip]
Saving to: "postgresql-14.19.tar.gz"

postgresql-14.19.tar.gz 100%[=====] 28.11M 3.69MB/s in 7.2s

2025-10-14 10:35:47 (3.89 MB/s) - "postgresql-14.19.tar.gz" saved [29472149/29472149]

root@aysrSlack:/tmp/postgresql# _
```

We listed the files to verify that the PostgreSQL source file was successfully downloaded.

```
root@aysrSlack:/tmp/postgresql# ls | grep postgresql-14.19.tar.gz
postgresql-14.19.tar.gz
root@aysrSlack:/tmp/postgresql# _
```

We moved one directory level up to organize the workspace before continuing and downloaded the SlackBuild package again to ensure we had all the necessary files.

```
root@aysrSlack:/tmp# wget https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz
--2025-10-14 10:44:44--  https://slackbuilds.org/slackbuilds/15.0/system/postgresql.tar.gz
Resolving slackbuilds.org (slackbuilds.org)... 66.85.79.67, 64:ff9b::4255:4f43
Connecting to slackbuilds.org (slackbuilds.org)|66.85.79.67|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 6860 (6.7K) [application/x-gzip]
Saving to: "postgresql.tar.gz.1"

postgresql.tar.gz.1      100%[=====]  6.70K  --.-KB/s   in 0s

2025-10-14 10:44:44 (2.21 GB/s) - "postgresql.tar.gz.1" saved [6860/6860]

root@aysrSlack:/tmp# _
```

We extracted the SlackBuild archive to access its contents and verify that everything was ready for building.

```
root@aysrSlack:/tmp# tar -xvf postgresql.tar.gz
postgresql/
postgresql/doinst.sh
postgresql/slack-desc
postgresql/README
postgresql/postgresql.logrotate
postgresql/rc.postgresql.new
postgresql/postgresql.info
postgresql/setup.postgresql
postgresql/README.SBo
postgresql/postgresql.SlackBuild
root@aysrSlack:/tmp# _
```

We listed the contents of `/tmp` to confirm that the PostgreSQL source and SlackBuild directories were correctly placed.

```
root@aysrSlack:/tmp# ls /tmp
postgresql/ postgresql.tar.gz  postgresql.tar.gz.1  prueba_cron.log
root@aysrSlack:/tmp# _
```

We moved the PostgreSQL source archive into the `postgresql` directory to maintain an organized structure before running the build script.

```
root@aysrSlack:/tmp# mv /tmp/postgresql.tar.gz /tmp/postgresql/
root@aysrSlack:/tmp# _
```

We created a new group and user called `postgres`, which PostgreSQL uses to run securely. We also created the directory `/var/lib/pgsql` and assigned ownership to the `postgres` user, since it will store the database data there.

```
root@aysrSlack:/tmp/postgresql# groupadd -g 209 postgres
root@aysrSlack:/tmp/postgresql# useradd -u 209 -g postgres -d /var/lib/pgsql -s /bin/bash postgres
root@aysrSlack:/tmp/postgresql# mkdir -p /var/lib/pgsql
root@aysrSlack:/tmp/postgresql# chown -R postgres:postgres /var/lib/pgsql
root@aysrSlack:/tmp/postgresql# _
```

We executed the `postgresql.SlackBuild` script to compile PostgreSQL from the source code and generate a Slackware package.

```
usr/share/postgresql-14/tsearch_data/
usr/share/postgresql-14/tsearch_data/danish.stop
usr/share/postgresql-14/tsearch_data/dutch.stop
usr/share/postgresql-14/tsearch_data/english.stop
usr/share/postgresql-14/tsearch_data/finnish.stop
usr/share/postgresql-14/tsearch_data/french.stop
usr/share/postgresql-14/tsearch_data/german.stop
usr/share/postgresql-14/tsearch_data/hungarian.stop
usr/share/postgresql-14/tsearch_data/hunspell_sample.affix
usr/share/postgresql-14/tsearch_data/hunspell_sample_long.affix
usr/share/postgresql-14/tsearch_data/hunspell_sample_long.dict
usr/share/postgresql-14/tsearch_data/hunspell_sample_num.affix
usr/share/postgresql-14/tsearch_data/hunspell_sample_num.dict
usr/share/postgresql-14/tsearch_data/ispell_sample.affix
usr/share/postgresql-14/tsearch_data/ispell_sample.dict
usr/share/postgresql-14/tsearch_data/italian.stop
usr/share/postgresql-14/tsearch_data/nepali.stop
usr/share/postgresql-14/tsearch_data/norwegian.stop
usr/share/postgresql-14/tsearch_data/portuguese.stop
usr/share/postgresql-14/tsearch_data/russian.stop
usr/share/postgresql-14/tsearch_data/spanish.stop
usr/share/postgresql-14/tsearch_data/swedish.stop
usr/share/postgresql-14/tsearch_data/synonym_sample.syn
usr/share/postgresql-14/tsearch_data/thesaurus_sample.ths
usr/share/postgresql-14/tsearch_data/turkish.stop
var/
var/lib/
var/lib/pgsql/
var/lib/pgsql/14/
var/lib/pgsql/14/data/
var/log/
var/log/setup/
var/log/setup/setup.postgresql

Slackware package /tmp/postgresql-14.19-x86_64-1_SBo.tgz created.

root@aysrSlack:/tmp/postgresql# _
```

We installed the generated PostgreSQL package using **installpkg**, making the database system available for use.

```
root@aysrSlack:/tmp/postgresql# installpkg /tmp/postgresql-14.19-x86_64-1_SBo.tgz
Verifying package postgresql-14.19-x86_64-1_SBo.tgz.
Installing package postgresql-14.19-x86_64-1_SBo.tgz:
PACKAGE DESCRIPTION:
# postgresql (object-relational database management system)
#
# PostgreSQL is an advanced object-relational database management
# system (ORDBMS) based on POSTGRES. With more than 15 years of
# development history, it is quickly becoming the de facto
# database for enterprise level open source solutions.
#
# Homepage: https://www.postgresql.org
#
Executing install script for postgresql-14.19-x86_64-1_SBo.tgz.
Package postgresql-14.19-x86_64-1_SBo.tgz installed.
root@aysrSlack:/tmp/postgresql#
```

We switched to the **postgres** user to perform the database initialization and configuration tasks.

```
root@aysrSlack:/tmp/postgresql# su - postgres
Workers of the world, arise! You have nothing to lose but your chairs.

postgres@aysrSlack:~$
```

We initialized the PostgreSQL data directory using **initdb**, creating the internal file structure necessary for the database to operate.

```
postgres@aysrSlack:~$ 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

postgres@aysrSlack:~$
```

We ran the initialization command again with additional parameters to specify the data path, locale, authentication method, and password for the postgres user. We used `su - postgres -c "initdb -D /var/lib/pgsql/14/data --locale=en_US.UTF-8 -A md5 -W"`

```
Starting PostgreSQL
You should initialize the PostgreSQL database at location /var/lib/pgsql/14/data
e.g. su postgres -c "initdb -D /var/lib/pgsql/14/data --locale=en_US.UTF-8 -A md5 -W"
root@aysrSlack:/tmp/postgresql# su - postgres -c "initdb -D /var/lib/pgsql/14/data --locale=en_US.UTF-8 -A md5 -W"
initdb: unrecognized option '--locale=en_US.UTF-8'
Try "initdb --help" for more information.
root@aysrSlack:/tmp/postgresql# su - postgres -c "initdb -D /var/lib/pgsql/14/data --locale=en_US.UTF-8 -A md5 -W"
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 locale "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.

Enter new superuser password:
Enter it again:

fixing permissions on existing directory /var/lib/pgsql/14/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

Success. You can now start the database server using:

    pg_ctl -D /var/lib/pgsql/14/data -l logfile start
root@aysrSlack:/tmp/postgresql# _
```

We started the PostgreSQL service using `/etc/rc.d/rc.postgresql start`, launching the database server and completing the installation process successfully.

```
root@aysrSlack:/tmp/postgresql# /etc/rc.d/rc.postgresql start
Starting PostgreSQL
waiting for server to start.... done
server started
root@aysrSlack:/tmp/postgresql# _
```

b. Create a user for each group member. Use the students' names as the username.

We switched to the postgres user and accessed the PostgreSQL terminal to start managing roles and databases directly as the database administrator.(passwd: admin123)

```

root@aysrSlack:/tmp/postgresql# su - postgres
Zisla's Law:
    If you're asked to join a parade, don't march behind the elephants.

postgres@aysrSlack:~$ psql
Password for user postgres:
psql (14.19)
Type "help" for help.

postgres=# _

```

We created two roles, one for each student, giving them login credentials so they can access and manage their own databases independently.

```

postgres=# CREATE ROLE santiago WITH LOGIN PASSWORD '1234';
CREATE ROLE
postgres=# CREATE ROLE julian WITH LOGIN PASSWORD '1234';
CREATE ROLE
postgres=#

```

We listed all the roles in the system to confirm that both users were successfully created and have login permissions.

Role name	Attributes	List of roles	
			Member of
julian		Ø	
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	Ø	
santiago		Ø	

```

postgres-# \du

```

c. 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.

We created a separate database for each user and assigned ownership to their respective roles, ensuring that each student has exclusive control over their own database.

```

postgres=# CREATE DATABASE turismo_santiago OWNER santiago;
CREATE DATABASE
postgres=# CREATE DATABASE turismo_julian OWNER julian;
CREATE DATABASE
postgres=#

```

We exited the PostgreSQL session and reconnected using the user “santiago” to start working inside his own database with his permissions.

```

postgres=# \q
postgres@aysrSlack:~$ psql -U santiago -d turismo_santiago -W
Password:
psql (14.19)
Type "help" for help.

turismo_santiago=>

```

We created three tables — departamentos, ciudades, and sitios\_turisticos — establishing relationships between them using foreign keys to organize the data hierarchically and maintain referential integrity.

```
turismo_santiago=> CREATE TABLE departamentos (id SERIAL PRIMARY KEY, nombre VARCHAR(100) NOT NULL);
ERROR: syntax error at or near "PRIMERY"
LINE 1: CREATE TABLE departamentos (id SERIAL PRIMERY KEY, nombre VA...
^
turismo_santiago=> CREATE TABLE departamentos (id SERIAL PRIMARY KEY, nombre VARCHAR(100) NOT NULL);
CREATE TABLE
turismo_santiago=> CREATE TABLE ciudades (id SERIAL PRIMARY KEY, nombre VARCHAR(100) NOT NULL, id_departamento INT REFERENCES departamentos(id));
CREATE TABLE
turismo_santiago=> CREATE TABLE sitios_turisticos (id SERIAL PRIMARY KEY, nombre VARCHAR(150) NOT NULL, descripcion TEXT, id_ciudad INT REFERENCES ciudades(id));
CREATE TABLE
turismo_santiago=> _
```

#### d. Insert data into the databases.

We inserted sample data into each table to simulate real information about departments, cities, and tourist sites, allowing us to test and validate the structure of the database.

```
turismo_santiago=> INSERT INTO departamentos (nombre) VALUES ('Antioquia'), ('Bolivar'), ('Cundinamarca');
INSERT 0 3
turismo_santiago=> INSERT INTO ciudades (nombre, id_departamento) VALUES ('Medellin',1), ('Cartagena',2), ('Bogota',3);
ERROR: relation "ciudades" does not exist
LINE 1: INSERT INTO ciudades (nombre, id_departamento) VALUES ('Mede...
^
turismo_santiago=> INSERT INTO ciudades (nombre, id_departamento) VALUES ('Medellin',1), ('Cartagena',2), ('Bogota',3);
INSERT 0 3
turismo_santiago=> INSERT INTO sitios_turisticos (nombre, descripcion, id_ciudad) VALUES ('Parque Arvi', 'Reserva natural con senderos ecológicos en Medellin', 1);
INSERT 0 1
turismo_santiago=> INSERT INTO sitios_turisticos (nombre, descripcion, id_ciudad) VALUES ('Castillo de San Felipe', 'Fuerte histórico en Cartagena.', 2), ('Monserrat', 'Cerro con santuario y vista panorámica de Bogota.', 3);
INSERT 0 2
turismo_santiago=> _
```

We selected all records from each table to verify that the data was inserted correctly and that the relationships between tables are functioning as expected.

turismo_santiago=> SELECT * FROM sitios_turisticos;			
id	nombre	descripcion	id_ciudad
1	Parque Arvi	Reserva natural con senderos ecológicos en Medellin	1
2	Castillo de San Felipe	Fuerte histórico en Cartagena.	2
3	Monserrat	Cerro con santuario y vista panorámica de Bogota.	3

turismo_santiago=> SELECT * FROM ciudades;		
id	nombre	id_departamento
1	Medellin	1
2	Cartagena	2
3	Bogota	3

turismo_santiago=> SELECT * FROM departamentos;	
id	nombre
1	Antioquia
2	Bolivar
3	Cundinamarca

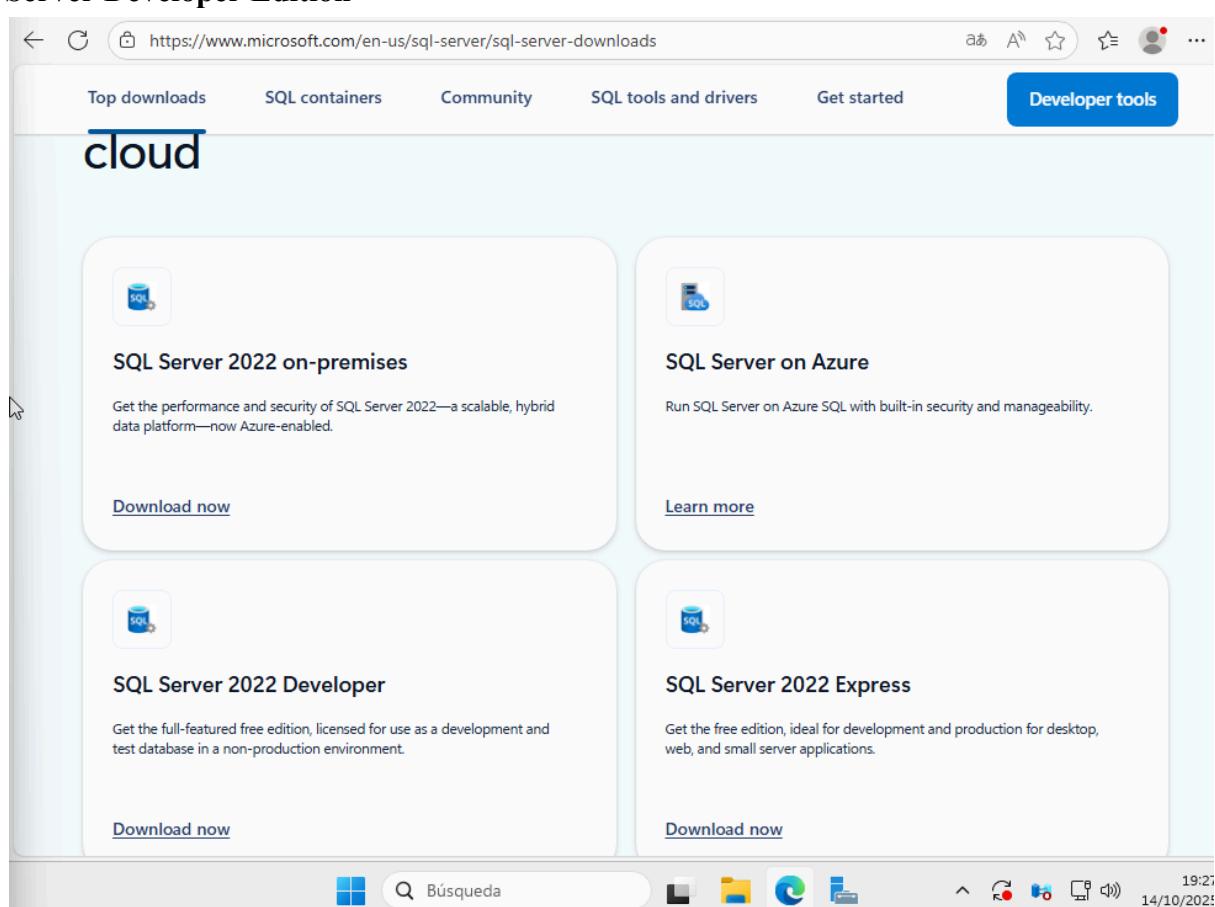
turismo\_santiago=> S\_

## 2. SQL Server - Windows Server

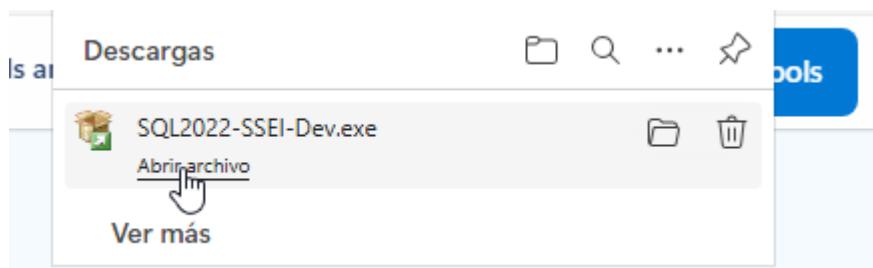
a. Install the SQL Server DBMS on a virtual machine running Windows Server.

First, we go to the following link:

<https://www.microsoft.com/en-us/sql-server/sql-server-downloads>, and we download **SQL Server Developer Edition**



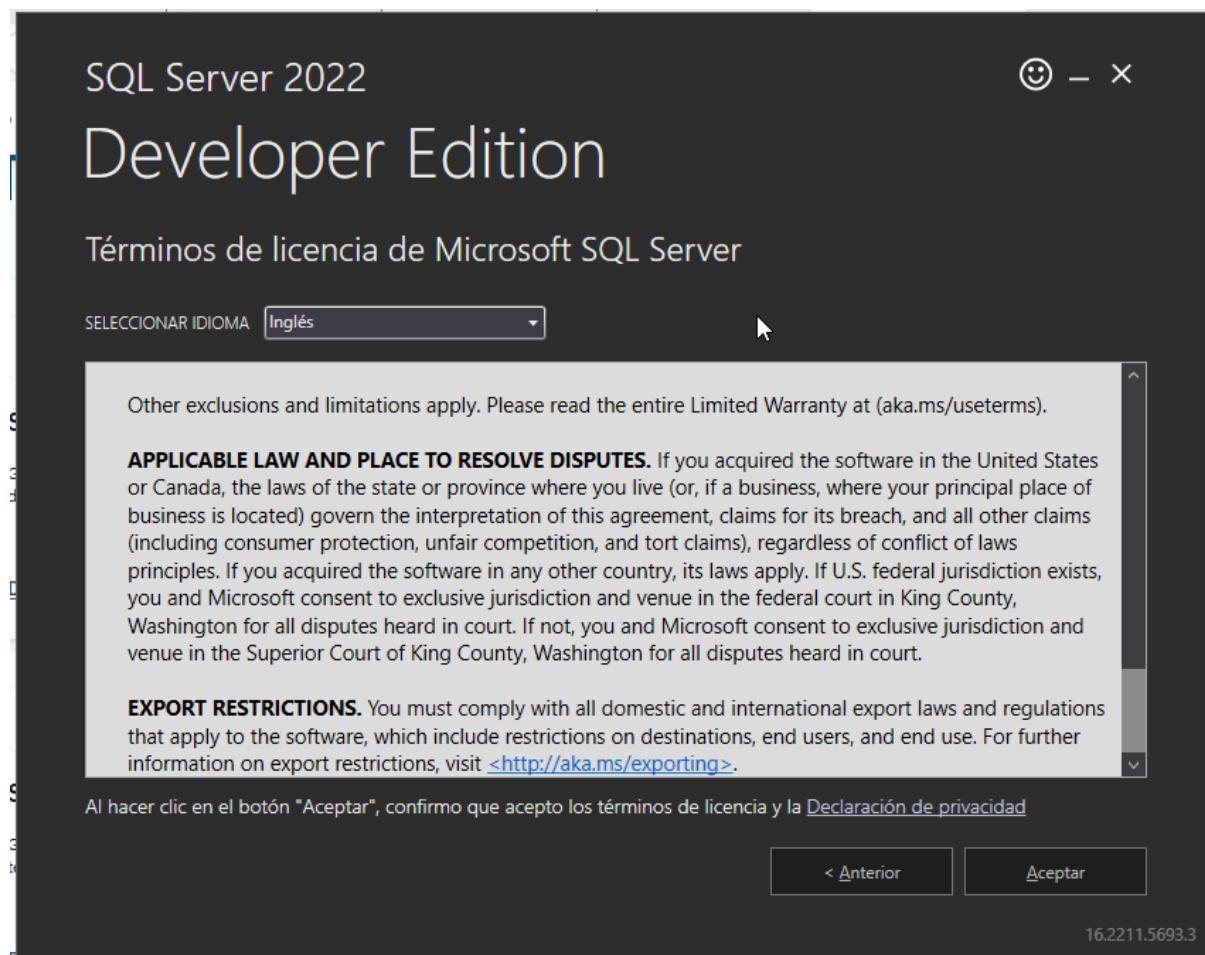
We run the .exe to start the installer



Select Basic or Custom for a more detailed installation.



We accept the terms, select the directory, and wait for the installation to complete.



SQL Server 2022



-

X

# Developer Edition

La descarga se ha completado correctamente.

Instalando...

Extrayendo archivos de instalación...

Forum Support

If you have questions on any of the above, you can post your question on Microsoft Q&A: Microsoft Q&A (<https://docs.microsoft.com/en-us/answers/topics/sql-server-general.html>) or you could post your question on Twitter using the #sqlhelp hashtag and tagging @SQLServer (<http://twitter.com/SQLServer>). Microsoft Q&A has more targeted forums for specific areas like Transact-SQL (<https://docs.microsoft.com/en-us/answers/topics/sql-server-transact-sql.html>), which can help with your questions on writing Transact-SQL code for querying the database or SQL Server Documentation (<https://docs.microsoft.com/en-us/sql/>).

You can post issues or suggestions on UserVoice: SQL Server (<http://aka.ms/sqlfeedback>).

Pausar

Cancelar

16.2211.5693.3

[Download now](#)

[Download now](#)

Finally, we download and install SSMS to be able to manage the server graphically.

SQL Server 2022



# Developer Edition

La instalación se ha completado correctamente.

NOMBRE DE INSTANCIA

MSSQLSERVER

CADENA DE CONEXIÓN

Server=localhost;Database=master;Trusted\_Connection=True;



ADMINISTRADORES DE SQL

WIN-IJDT4H4DQP8\Administrador

CARPETA DEL REGISTRO DE INSTALACIÓN DE SQL SERVER

C:\Program Files\Microsoft SQL Server\160\Setup Bootstrap\Log\2025101



CARACTERÍSTICAS INSTALADAS

SQLENGINE

CARPETA DE MEDIOS DE INSTALACIÓN

C:\SQL2022\Developer\_ENU



VERSIÓN

16.0.1000.6, RTM

CARPETA DE RECURSOS DE INSTALACIÓN

C:\Program Files\Microsoft SQL Server\160\SSEI\Resources



← Conectarse ahora

Personalizar

Instalar SSMS

Cerrar

Instalar SSMS

Un entorno integrado para configurar, administrar, gestionar y desarrollar todos los componentes de SQL Server, así como para acc

<https://learn.microsoft.com/es-es/ssms/install/install?redirectedfrom=MSDN>

## Paso 2: Determinación de la versión de SQL Server Management Studio que se va a instalar

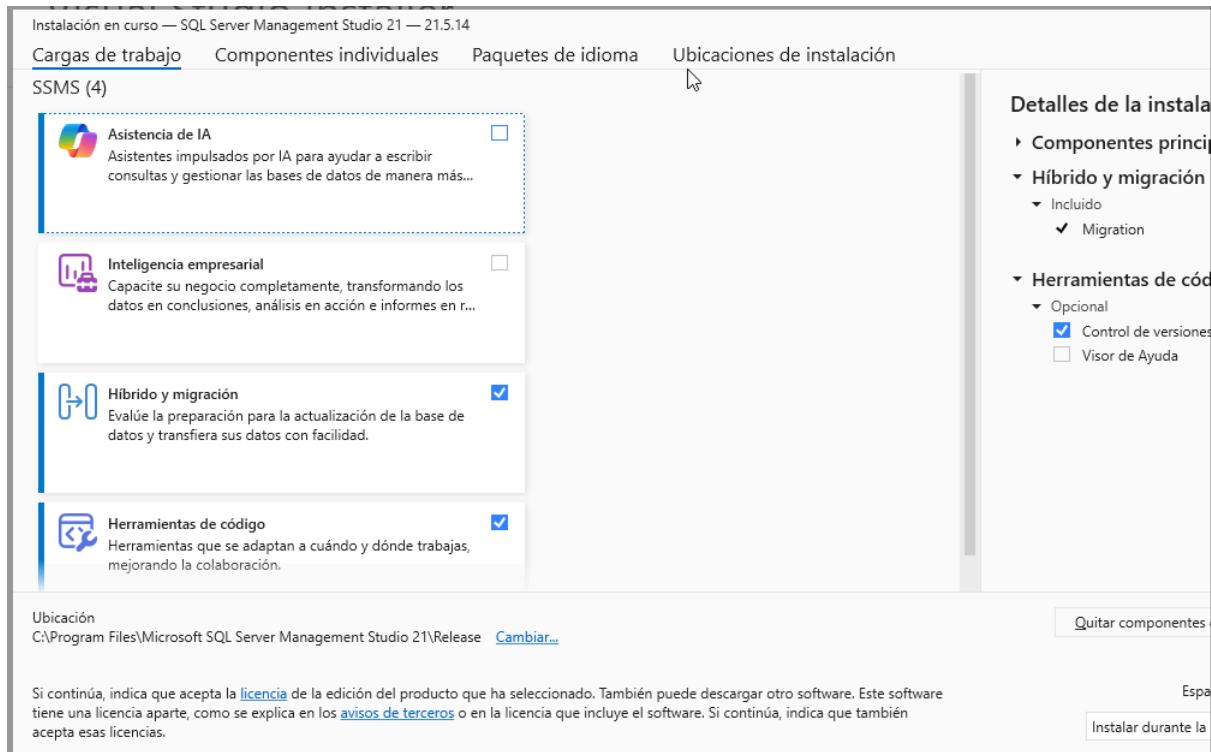
Decida qué versión de SSMS va a instalar. Las opciones más comunes son:

- La versión más reciente de SQL Server Management Studio 21 hospedada en servidores de Microsoft. Para instalar esta versión, seleccione el vínculo siguiente, que descarga un instalador reducido o *programa de arranque* en su carpeta *Descargas*.

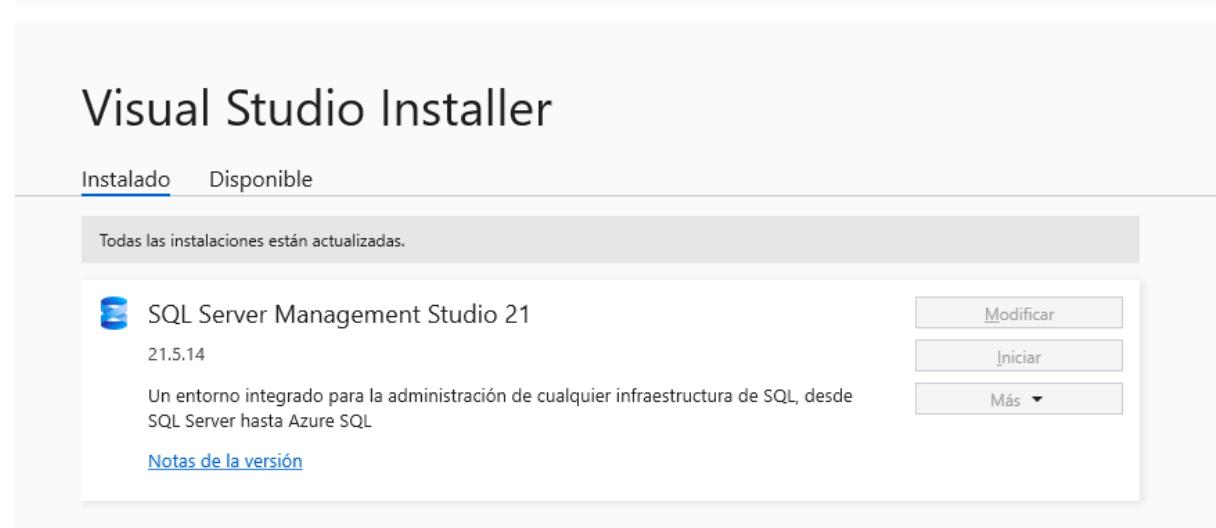
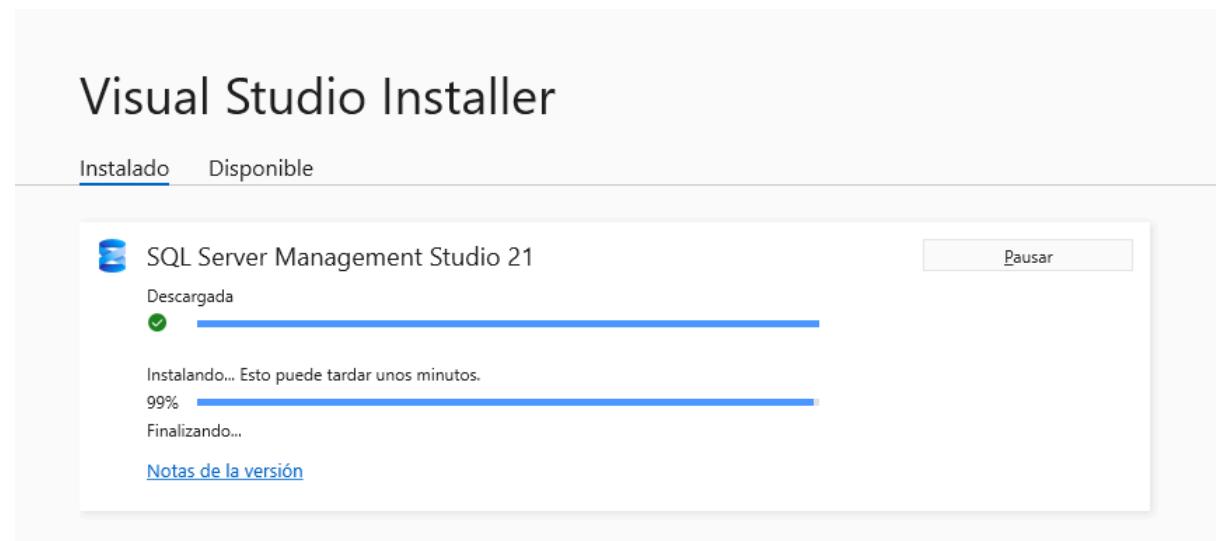
[Descarga de SSMS 21](#)

- Si ya tiene instalado SQL Server Management Studio 21, puede [instalar otra versión junto a esta](#).

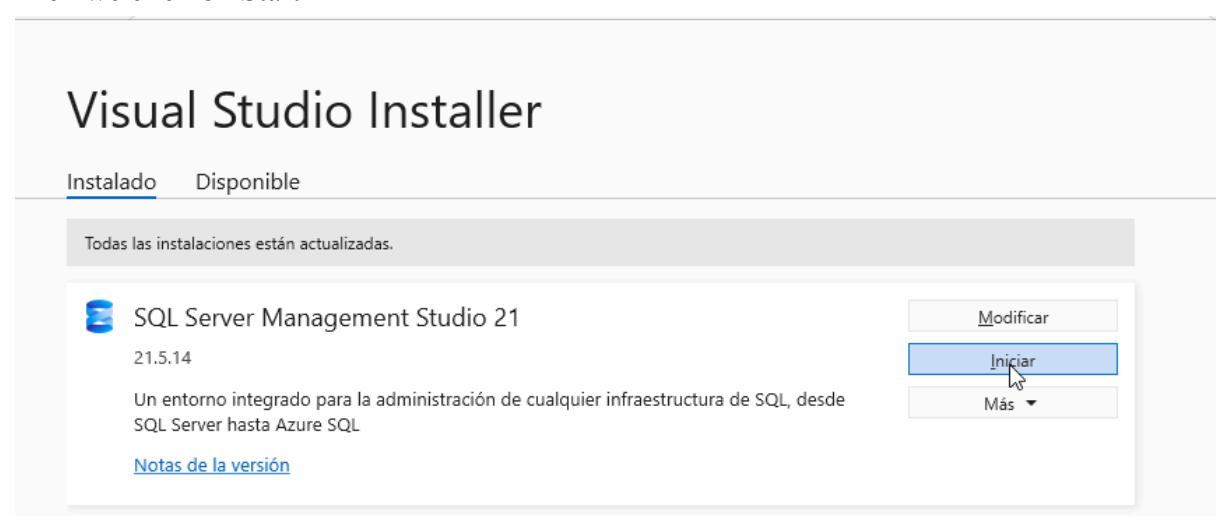
Puede descargar un bootstrapper o instalador para una versión específica desde la página [Historial de versiones](#) y utilizarlo para instalar otra versión de SSMS.



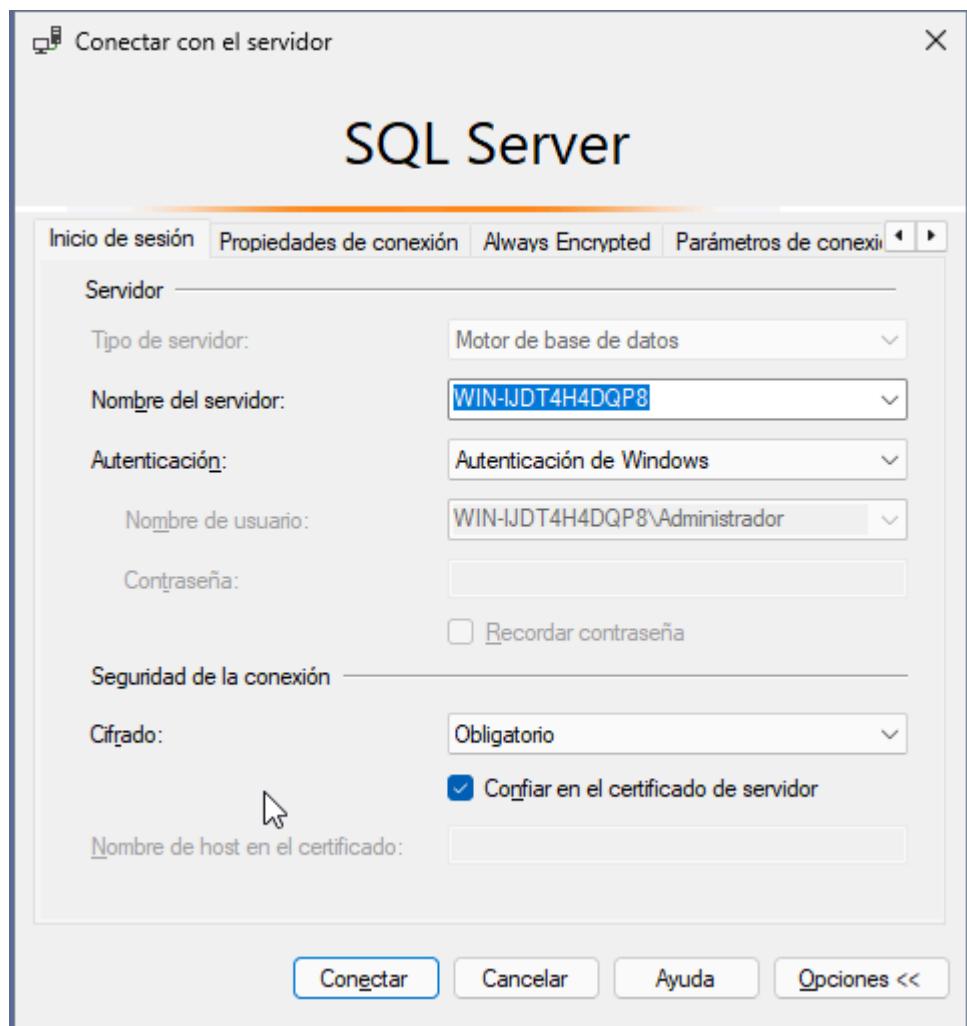
We wait for it to finish installing



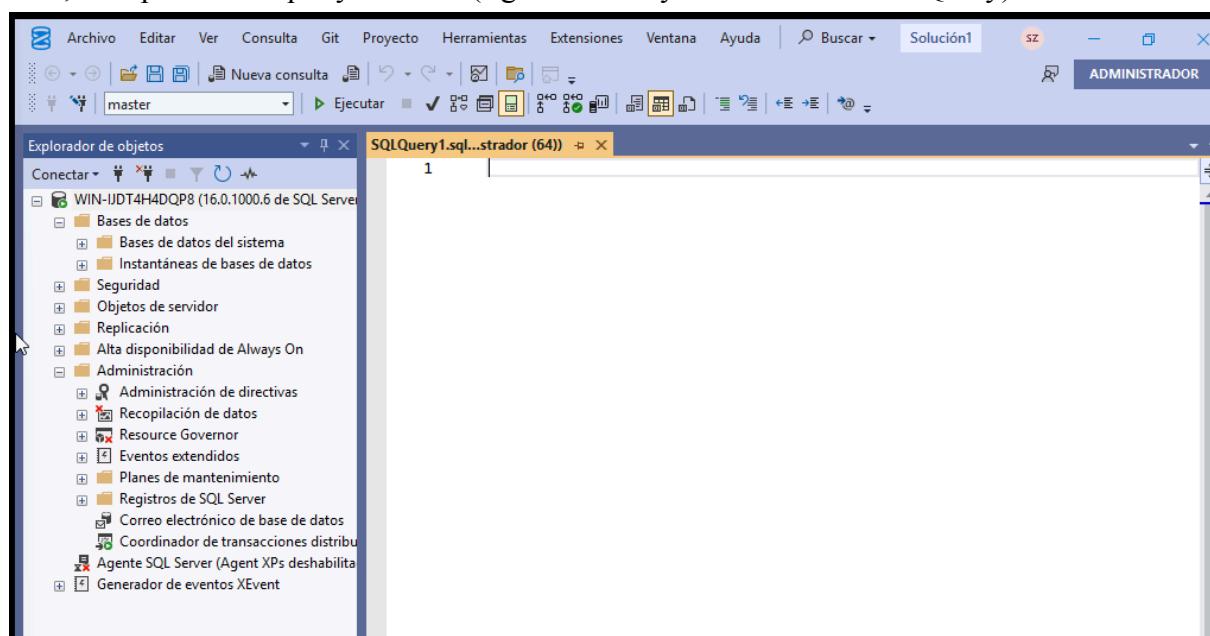
Then we click on Start



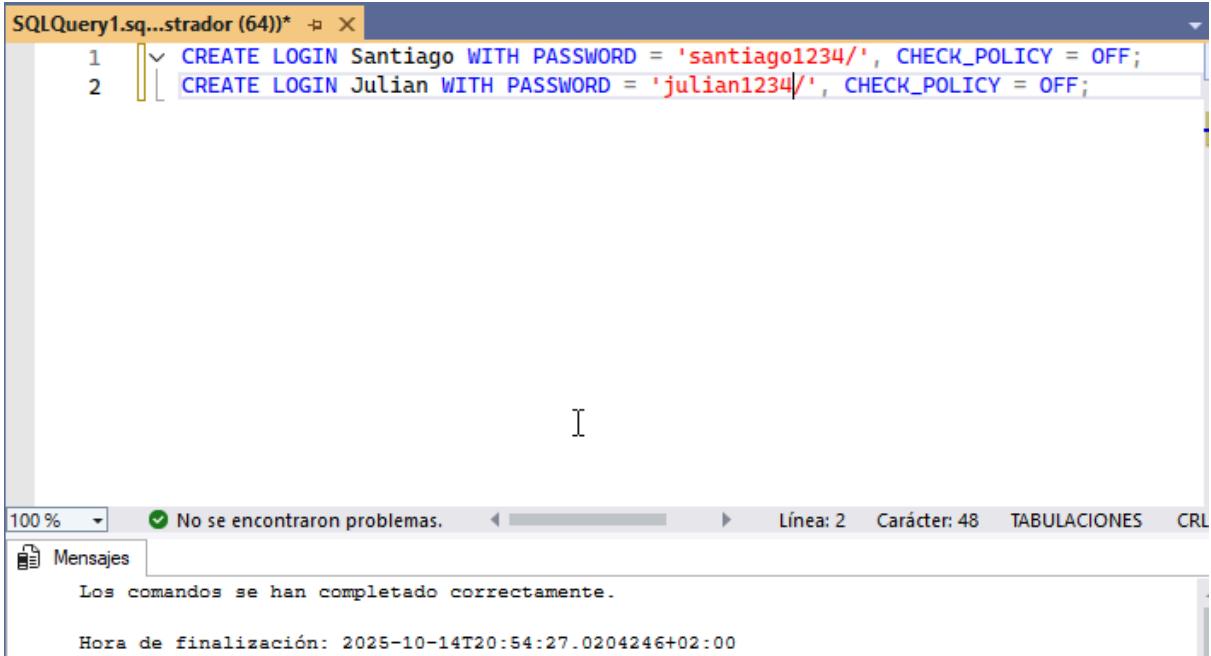
We log in with our school Microsoft account to start the connection. We mark on “Confiar en el certificado de servidor”



b. Create a user for each group member. Use the students' names as the username.  
 First, we open a new query window (right click on your server → New Query).



Create a user for each group member

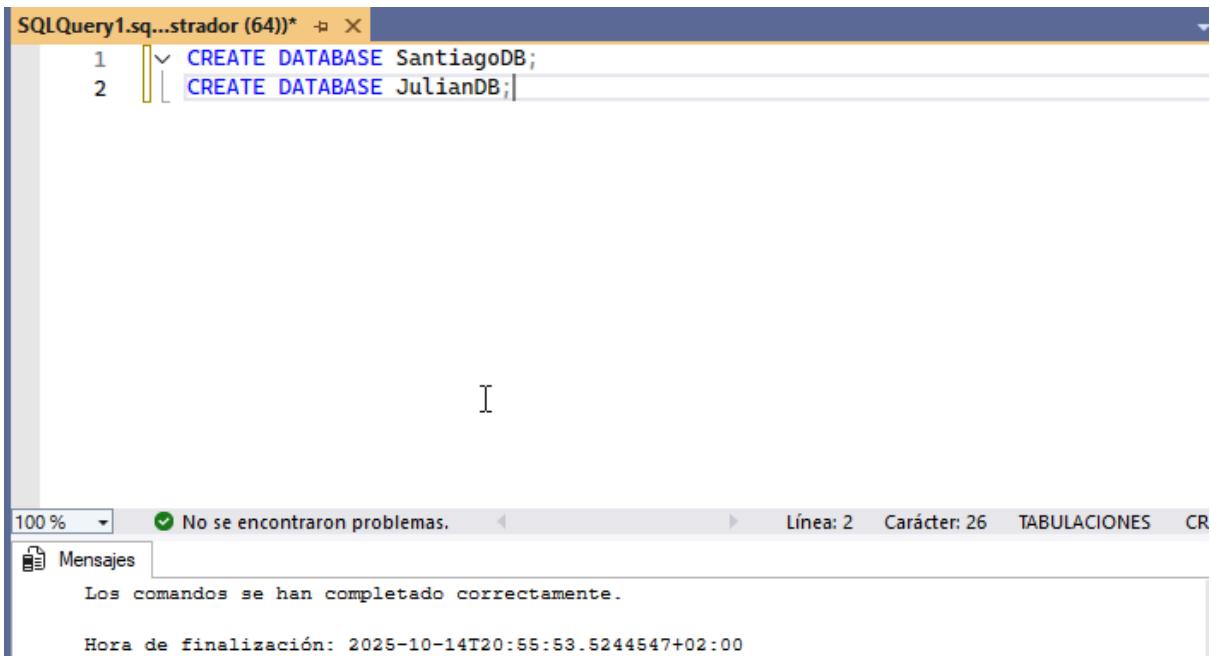


```
SQLQuery1.sq...strador (64)* 1 CREATE LOGIN Santiago WITH PASSWORD = 'santiago1234/' , CHECK_POLICY = OFF; 2 CREATE LOGIN Julian WITH PASSWORD = 'julian1234/' , CHECK_POLICY = OFF;
```

100 % ✓ No se encontraron problemas. Línea: 2 Carácter: 48 TABULACIONES CRL  
Mensajes  
Los comandos se han completado correctamente.  
Hora de finalización: 2025-10-14T20:54:27.0204246+02:00

c. 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.

We created a database for each student



```
SQLQuery1.sq...strador (64)* 1 CREATE DATABASE SantiagoDB; 2 CREATE DATABASE JulianDB;
```

100 % ✓ No se encontraron problemas. Línea: 2 Carácter: 26 TABULACIONES CR  
Mensajes  
Los comandos se han completado correctamente.  
Hora de finalización: 2025-10-14T20:55:53.5244547+02:00

Assign access permissions only to your own database

The screenshot shows a SQL query window titled "SQLQuery1.sql...strador (64)\*". The code creates two users, "Santiago" and "Julian", and adds them to the "db\_owner" role in their respective databases, "SantiagoDB" and "JulianDB". The execution message indicates the commands were completed successfully at 2025-10-14T20:59:41.7094924+02:00.

```
1 USE SantiagoDB;
2 CREATE USER Santiago FOR LOGIN Santiago;
3 ALTER ROLE db_owner ADD MEMBER Santiago;
4
5 USE JulianDB;
6 CREATE USER Julian FOR LOGIN Julian;
7 ALTER ROLE db_owner ADD MEMBER Julian;
```

Mensajes

Los comandos se han completado correctamente.

Hora de finalización: 2025-10-14T20:59:41.7094924+02:00

Create the 3 tables to organize the monthly schedule

The screenshot shows a SQL query window titled "SQLQuery1.sql...strador (64)\*". It contains T-SQL code to create three tables: "Category", "Activity", and "Schedule". The "Category" table has an identity primary key and a non-null category name. The "Activity" table has an identity primary key, a non-null title, a description, and a foreign key linking to the "Category" table. The "Schedule" table has an identity primary key, activity ID, activity date, start time, end time, and a foreign key linking to the "Activity" table.

```
1 USE SantiagoDB;
2
3 CREATE TABLE Category (
4     CategoryID INT IDENTITY(1,1) PRIMARY KEY,
5     CategoryName VARCHAR(50) NOT NULL
6 );
7
8 CREATE TABLE Activity (
9     ActivityID INT IDENTITY(1,1) PRIMARY KEY,
10    Title VARCHAR(100) NOT NULL,
11    Description VARCHAR(255),
12    CategoryID INT,
13    FOREIGN KEY (CategoryID) REFERENCES Category(CategoryID)
14 );
15
16 CREATE TABLE Schedule (
17     ScheduleID INT IDENTITY(1,1) PRIMARY KEY,
18     ActivityID INT,
19     ActivityDate DATE,
20     StartTime TIME,
21     EndTime TIME,
22     FOREIGN KEY (ActivityID) REFERENCES Activity(ActivityID)
23 );
```

d. Insert data into the databases.

```
SQLQuery1.sq...strador (64)* USE SantiagoDB;
1  USE SantiagoDB;
2
3  INSERT INTO Category (CategoryName)
4  VALUES ('Estudio'), ('Ejercicio'), ('Ocio');
5
6  INSERT INTO Activity (Title, Description, CategoryID)
7  VALUES
8  ('Leer libro de redes', 'Repasar capítulo 4', 1),
9  ('Correr en el parque', '30 minutos', 2),
10 ('Jugar videojuegos', 'Tiempo libre', 3);
11
12 INSERT INTO Schedule (ActivityID, ActivityDate, StartTime, EndTime)
13 VALUES
14  (1, '2025-10-15', '08:00', '09:00'),
15  (2, '2025-10-15', '18:00', '18:30');

100 % 11 0 ↑ ↓ Línea: 16 Carácter: 37 TABULACIONES
Mensajes

(3 filas afectadas)

(3 filas afectadas)

(3 filas afectadas)

Hora de finalización: 2025-10-14T21:07:22.823Z+02:00
```

### 3. 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.

We signed up with my institutional email

The screenshot shows the Microsoft Azure portal's login interface. At the top, there is a search bar, a Copilot button, and a sign-in button for 'julian.tinjaca-c@mail.es... ESCUELA COLOMBIANA DE INGE...'. Below the sign-in button, there are links for 'Ayúdame a entender cómo estimar los costos' and 'Guíame a través de la creación de un nuevo recurso en Azure'. A message at the bottom left says '¡ejercicios para temas como la navegación, la arquitectura de su primera solución de Azure'. On the right, the user's profile information is displayed: 'julian.tinjaca-c@mail.es...', 'julian.tinjaca-c@mail.es...', 'Ver cuenta', 'Cambiar directorio', and a '...' button. Below the profile, there is a link 'Iniciar sesión con una cuenta distinta'. At the bottom of the page, there are navigation links for 'Precios y administración' and 'Completar'.

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?

Cloud computing is the delivery of IT services over the internet on a pay-as-you-go basis. Advantages of Azure (Cloud) versus on-premise infrastructure include cost, scalability, agility, and maintenance. The cloud uses an OpEx model instead of the large upfront CapEx of on-premise. Cloud is elastic, meaning you can add or remove resources in minutes, whereas on-premise scaling is slow and expensive. It also provides agility to deploy applications globally in minutes and offloads hardware maintenance to Microsoft.

- What types of services does Microsoft Azure offer (IaaS, PaaS, SaaS), and how do they differ from one another?

These are the three main service models, defining what you manage versus what the provider manages.

IaaS (Infrastructure as a Service) provides the basic building blocks. The provider manages the physical hardware, while we manage the operating system, data, and applications. An example is Azure Virtual Machines.

PaaS (Platform as a Service) provides a managed platform for development. The provider manages the servers and the operating system. Some examples include Azure SQL Database and Azure App Service.

SaaS (Software as a Service) is a complete, ready to use application where the provider manages everything. We manage nothing, just use the software. Examples include Microsoft 365 and Gmail.

- What is the importance of regions and availability zones in Azure, and how do they affect service availability?

Their importance is about speed and reliability.

Regions let us place our services close to our users, which reduces latency (lag). They also help us comply with data laws that require data to stay in a specific country.

Availability Zones provide high availability. They are separate data centers in one region, so if one fails, our apps keep running from another, preventing downtime.

- What is the difference between vertical scaling and horizontal scaling in Azure, and when would you choose one over the other?

Vertical scaling means making a single server stronger, this could be more CPU or RAM, which we would choose for monolithic applications like a traditional database.

Horizontal scaling means adding more servers to share the load, which is better for modern web apps that need high availability and massive scale.

- 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?

The main difference is security because Azure SQL Database enforces TLS encryption by default, so our connection is always secure, otherwise a local VM database often has TLS disabled on an internal network, meaning data might be sent unencrypted, which is a security risk

- 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?

A local VM database has a direct TCP connection to the VM's specific IP and port.

Azure SQL Database uses a gateway. We connect to the gateway first, which then redirects our TCP connection to the correct database server.

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

First we have to access the interface and then create the server and the database

Nombre	Servidor	Tipo de réplica	Plan de tarifa	Ubicación	Suscripción
LibraryDB (srv-my-biblioteca-2025/LibraryDB)	srv-my-biblioteca-2025	--	General Purpose: Gen5, 2 vCores	West US 2	Azure subscription 1

Once it is created we have to click on the query editor, there is where we create the tables and start to insert data

it asks us to log in with the user and password we created or with the institutional email, we log in with the user.

El editor de consultas (versión preliminar) es una herramienta para ejecutar consultas SQL en Azure SQL Database en Azure Portal. Está diseñado para realizar consultas ligeras y explorar objetos en la base de datos. Para obtener más información y solucionar problemas, [Más información](#)



Le damos la bienvenida al editor de consultas de SQL Database

Here's the space to start manipulating the database

We start creating 3 tables

Consulta 1 ×

Ejecutar Cancelar consulta Guardar consulta Exportar datos como Mostrar solo editor

```
1 CREATE TABLE Authors (
2     AuthorID INT PRIMARY KEY IDENTITY(1,1),
3     FirstName NVARCHAR(100) NOT NULL,
4     LastName NVARCHAR(100) NOT NULL,
5     Email NVARCHAR(255) UNIQUE
6 );
7
8 CREATE TABLE Resources (
9     ResourceID INT PRIMARY KEY IDENTITY(1,1),
10    Title NVARCHAR(500) NOT NULL,
11    ResourceType NVARCHAR(50) NOT NULL CHECK (ResourceType IN ('Book', 'Scientific Article')),
12    PublicationDate DATE,
13    ISSN_ISBN NVARCHAR(20) UNIQUE,
14    Abstract NVARCHAR(MAX)
15
16 CREATE TABLE Resource_Authors (
17     ResourceID INT NOT NULL,
18     AuthorID INT NOT NULL,
19     PRIMARY KEY (ResourceID, AuthorID),
20
21     FOREIGN KEY (ResourceID) REFERENCES Resources(ResourceID) ON DELETE CASCADE,
22     FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID) ON DELETE CASCADE
23 );
24 ):
```

d. Insert data into the database.

Then we insert some data on the tables

```
1 INSERT INTO Authors (FirstName, LastName, Email) VALUES
2 ('Gabriel', 'García Márquez', 'ggm@example.com'),
3 ('Marie', 'Curie', 'mcurie@science.org'),
4 ('Alan', 'Turing', 'turing@cs.com');
5
6 INSERT INTO Resources (Title, ResourceType, PublicationDate, ISSN_ISBN, Abstract) VALUES
7 ('One Hundred Years of Solitude', 'Book', '1967-05-30', '978-0307474728', 'A foundational novel of magical realism.'),
8 ('Radioactivity and New Transformations', 'Scientific Article', '1903-01-01', 'ISSN-1903-001', 'A key article on the discovery of radioactivity and new transformations in science'),
9 ('On Computable Numbers', 'Scientific Article', '1936-05-28', 'ISSN-1936-002', 'The theoretical basis of modern computer science and mathematics')
10
11 INSERT INTO Resource_Authors (ResourceID, AuthorID) VALUES
12 (1, 1),
13 (2, 2),
14 (3, 3);
```

We verify that everything we added to the database is working

Consulta 2 ×

Ejecutar Cancelar consulta Guardar consulta Exportar datos como Mostrar solo editor

```
1 SELECT
2     R.Title,
3     R.ResourceType,
4     A.FirstName + ' ' + A.LastName AS Author
5 FROM
6     Resources AS R
7 INNER JOIN
8     Resource_Authors AS RA ON R.ResourceID = RA.ResourceID
9 INNER JOIN
10    Authors AS A ON RA.AuthorID = A.AuthorID;
```

Resultados Mensajes

Buscar en elementos de filtro...

Title	ResourceType	Author
One Hundred Years of Solitude	Book	Gabriel García Márquez
Radioactivity and New Transformations	Scientific Article	Marie Curie
On Computable Numbers	Scientific Article	Alan Turing

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

<https://youtu.be/HVzpy2ttcU0>

- f. Delete all resources created for the database (including the database itself) to avoid additional costs and depletion of available credits.

It's deleted now.

## 5. 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.

On Linux Slackware(PostgreSQL)

First we check that the startup script exists

```
root@aysrSlack:~# ls -l /etc/rc.d/rc.postgresql
-rwxr-xr-x 1 root root 4983 Oct 14 11:03 /etc/rc.d/rc.postgresql*
root@aysrSlack:~#
```

We manually check that rc.postgresql works

```
root@aysrSlack:~# /etc/rc.d/rc.postgresql start
Starting PostgreSQL
waiting for server to start.... done
server started
root@aysrSlack:~#
```

You also need to verify that PostgreSQL responds.

```
root@aysrSlack:~# ps aux | grep postgres
postgres 1518 0.0 1.2 163612 19228 ? Ss 09:22 0:00 /usr/lib64/postgresql/14/bin/postgres -D /var/lib/pgsql/14/data -p 5432
postgres 1520 0.0 0.2 163612 3516 ? Ss 09:22 0:00 postgres: checkpointer
postgres 1521 0.0 0.3 163612 5180 ? Ss 09:22 0:00 postgres: background writer
postgres 1522 0.0 0.6 163612 8948 ? Ss 09:22 0:00 postgres: walwriter
postgres 1523 0.0 0.5 164152 7592 ? Ss 09:22 0:00 postgres: autovacuum launcher
postgres 1524 0.0 0.2 18348 3964 ? Ss 09:22 0:00 postgres: stats collector
postgres 1525 0.0 0.3 164048 5900 ? Ss 09:22 0:00 postgres: logical replication launcher
root 1539 0.0 0.1 3988 2100 ttys1 S+ 09:24 0:00 grep postgres
root@aysrSlack:~#
```

We edit /etc/rc.d/rc.local (as root) and add the line before the final line exit 0 (or if there is no exit 0, at the end of the file)

```
root@aysrSlack:~# nano /etc/rc.d/rc.local
```

```
GNU nano 6.0                               /etc/rc.d/rc.local                         Modified
#!/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.
if [ -x /etc/rc.d/rc.postgresql ]; then
    /etc/rc.d/rc.postgresql start
fi
```

We save. Then, we must also verify that rc.local is executable.

```
root@aysrSlack:~# chmod +x /etc/rc.d/rc.local
root@aysrSlack:~# _
```

Upon restart we can see that the service started correctly

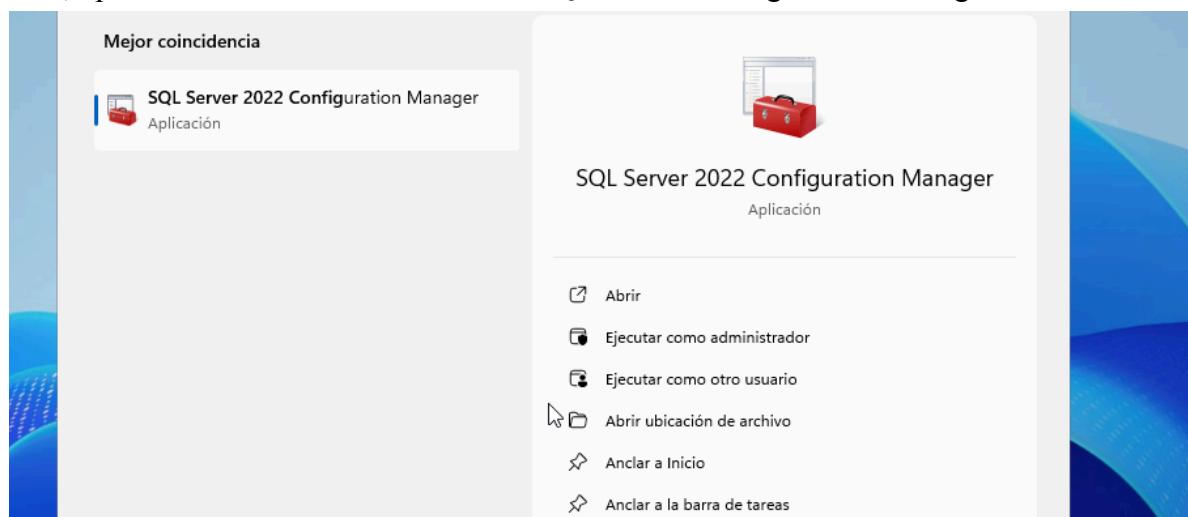
```
Starting gpm: /usr/sbin/gpm -m /dev/mouse -t imps2
Starting PostgreSQL
pg_ctl: another server might be running; trying to start server anyway
waiting for server to start.... done
server started

Welcome to Linux 5.15.19 x86_64 (tty1)

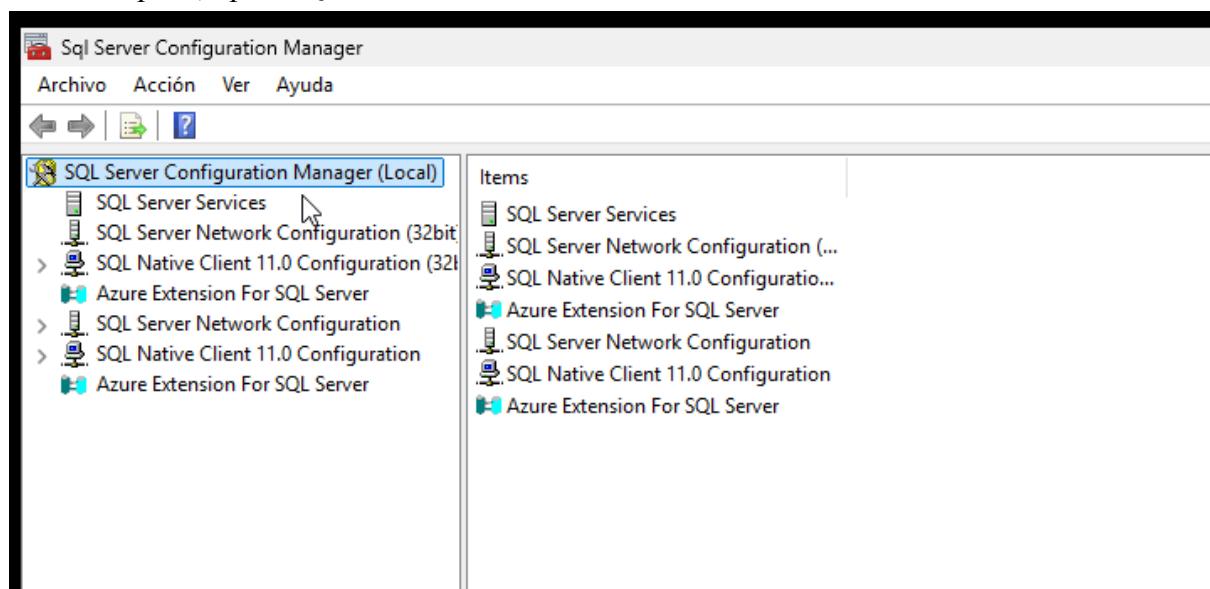
aysrSlack login: _
```

On Windows Server(SQL Server)

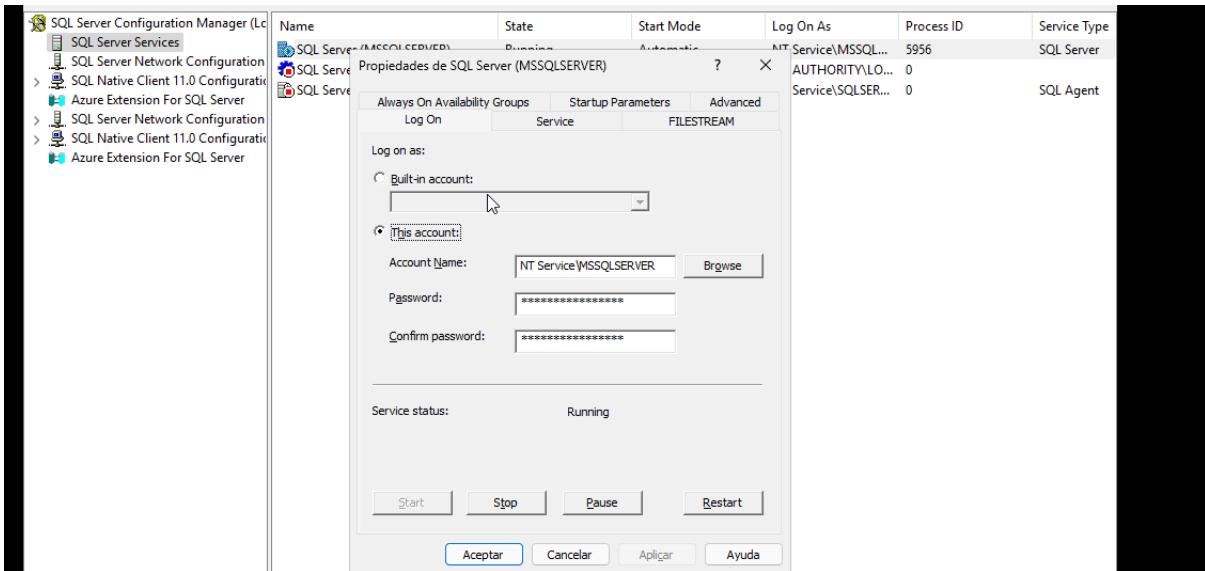
First, open the Start menu and search for SQL Server Configuration Manager.



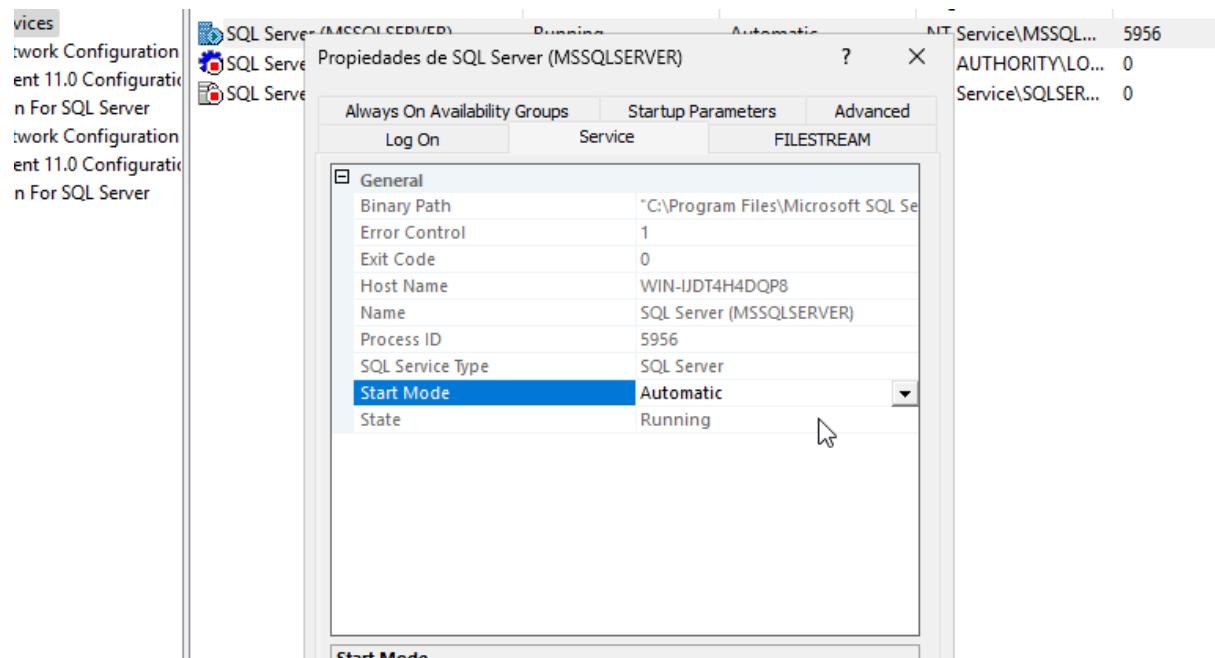
In the left pane, open SQL Server Services



We right click on SQL Server (MSSQLSERVER) → Properties.



In the Service tab, we look for Start Mode and change it to Automatic.



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

On Linux Slackware:

First, Edit postgresql.conf to allow remote listening

```
root@aysrSlack:~# cd /var/lib/pgsql/14/data
root@aysrSlack:/var/lib/pgsql/14/data# cp postgresql.conf.bak
cp: missing destination file operand after 'postgresql.conf.bak'
Try 'cp --help' for more information.
root@aysrSlack:/var/lib/pgsql/14/data# cp postgresql.conf postgresql.conf.bak
root@aysrSlack:/var/lib/pgsql/14/data#
```

We look for the line `#listen_addresses = 'localhost'` and change it to: `listen_addresses = '*'`

```
GNU nano 6.0                               postgresql.conf                         Modified

# 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                                 # (change requires restart)
max_connections = 100                        # (change requires restart)
#superuser_reserved_connections = 3          # (change requires restart)
#unix_socket_directories = '/tmp'            # comma-separated list of directories
                                              # (change requires restart)
#unix_socket_group = ''                      # (change requires restart)
#unix_socket_permissions = 0777              # begin with 0 to use octal notation
                                              # (change requires restart)
# - SSL Settings -
#ssl = off                                     # (change requires restart)
#ssl_cert_file = ''                           # (change requires restart)
#ssl_key_file = ''                            # (change requires restart)
#ssl_ciphers = ''                             # (change requires restart)
```

We make a backup and edit pg\_hba.conf

```
root@aysrSlack:/var/lib/pgsql/14/data# cp pg_hba.conf pg_hba.conf.bak
root@aysrSlack:/var/lib/pgsql/14/data# nano pg_hba.conf_
```

At the end we add a line to authenticate with MD5 password:

```
# replication privilege.
local  replication  all                                md5
host   replication  all      127.0.0.1/32           md5
host   replication  all      ::1/128                md5
# Allow all to connect all DB with md5
host   all          all      0.0.0.0/0             md5
```

Then, restart PostgreSQL

```
root@aysrSlack:/var/lib/pgsql/14/data# /etc/rc.d/rc.postgresql restart
Restarting PostgreSQL...
waiting for server to shut down.... done
server stopped
waiting for server to start.... done
server started
root@aysrSlack:/var/lib/pgsql/14/data# _
```

Allow access from 192.168.56.1 (CLIENT IP)

```
root@aysrSlack:/var/lib/pgsql/14/data# iptables -I INPUT -p tcp -s 192.168.56.1 --dport 5432 -j ACCEPT
root@aysrSlack:/var/lib/pgsql/14/data# _
```

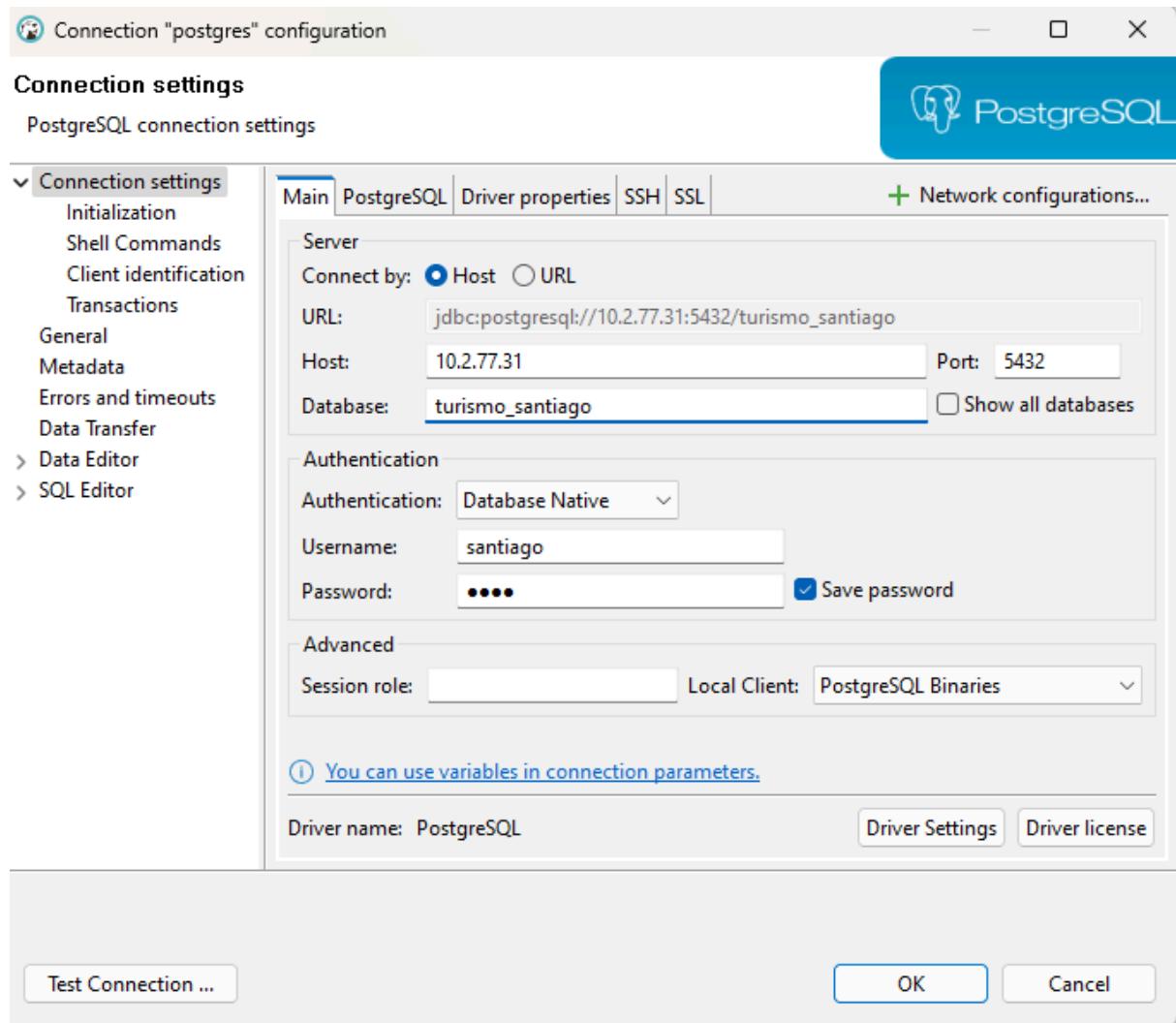
We keep the rules

```
root@aysrSlack:/var/lib/pgsql/14/data# iptables-save > /etc/iptables.rules
root@aysrSlack:/var/lib/pgsql/14/data# _
```

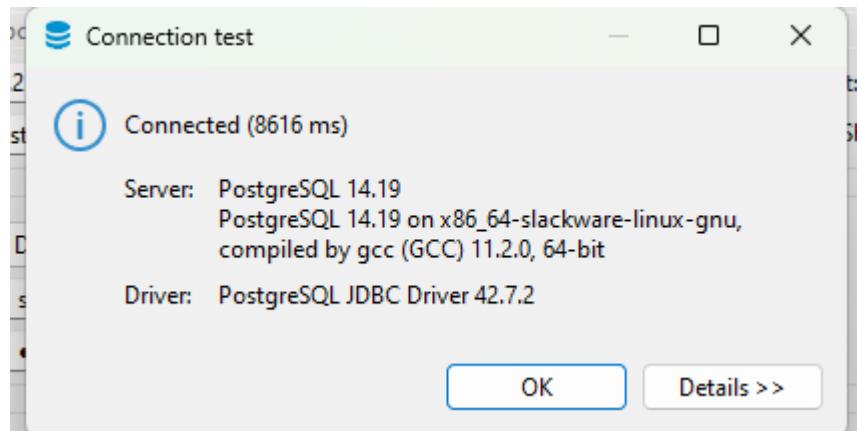
We verify that PostgreSQL is listening on port 5432

```
root@aysrSlack:/var/lib/pgsql/14/data# ss -ltnp | grep 5432
LISTEN 0      244                           0.0.0.0:5432        0.0.0.0:*      users:(("postgres",pid=1538,fd=5))
LISTEN 0      244                           [::]:5432          [::]:*       users:(("postgres",pid=1538,fd=6))
root@aysrSlack:/var/lib/pgsql/14/data# _
```

In DBeaver, choose PostgreSQL and fill out the form as follows:



Press Test Connection and verify



Then, we verify that it is working

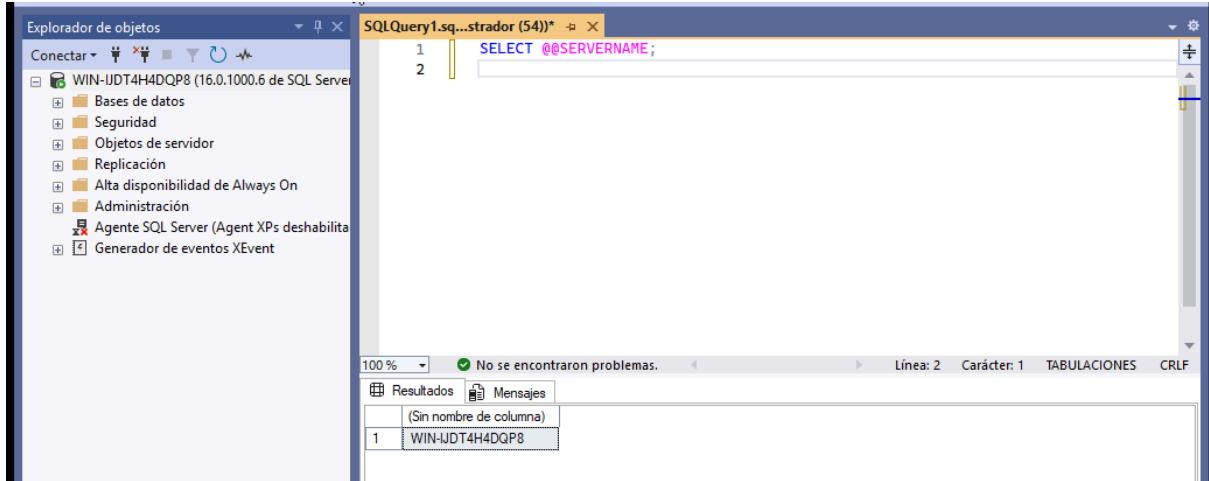
The screenshot shows the Database Navigator interface. On the left, the 'Database Navigator' sidebar lists databases: 'postgres - 10.2.77.31:5432' (selected), 'SantiagoDB - 10.2.77.33:1433', and 'SantiagoDB - 10.2.77.33:1433'. The main area shows a script tab with the SQL query 'select \* from ciudades;'. Below it, a results grid titled 'ciudades 1' displays the following data:

id	nombre	id_departamento
1	Medellin	1
2	Cartagena	2
3	Bogota	3

On Windows Server:

First we check which instance of SQL Server is being used, open SSMS and run:

SELECT @@SERVERNAME; . We save that value, since it is the name of the server we are going to use in DBeaver.

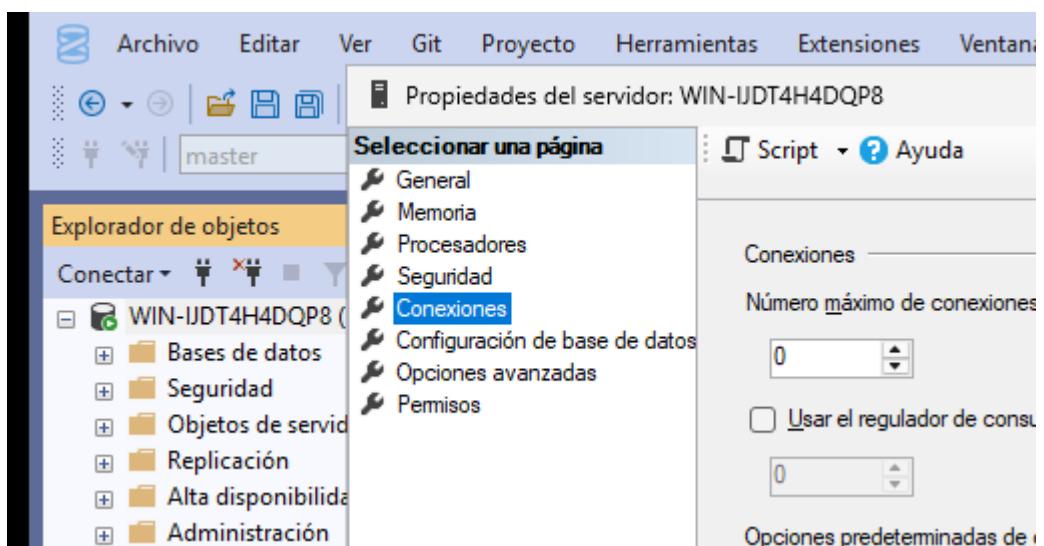


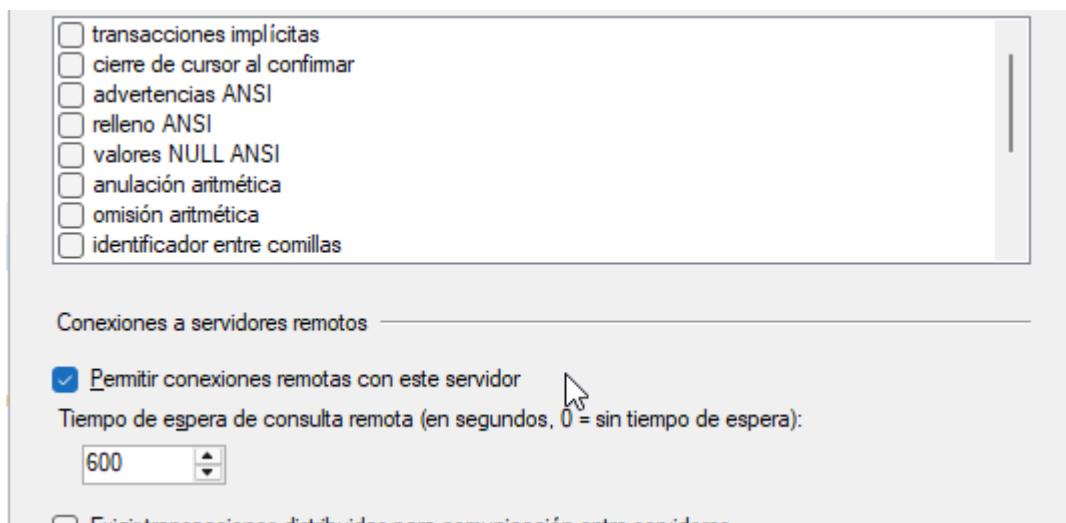
The screenshot shows the SSMS interface. On the left is the Object Explorer pane, which is expanded to show the connection details for 'WIN-IJDT4H4DQP8 (16.0.1000.6 de SQL Server)'. The 'Administración' folder is selected. In the center is the 'SQLQuery1.sq...rador (54)\*' query window containing the following code:

```
1  SELECT @@SERVERNAME;
2
```

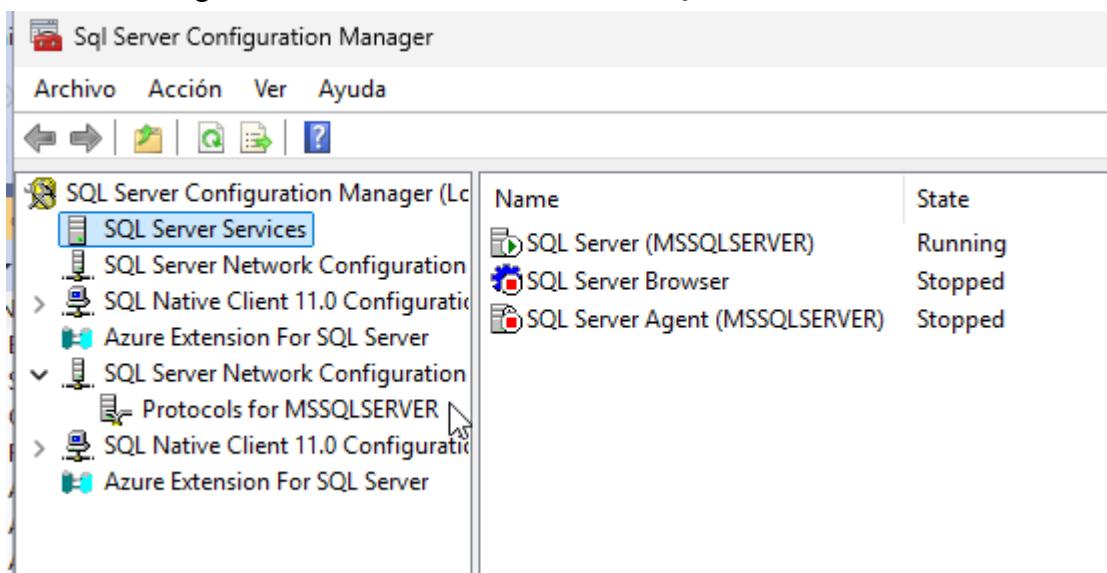
Below the query window is the results grid, which displays one row with the value 'WIN-IJDT4H4DQP8' in the first column. The status bar at the bottom indicates 'No se encontraron problemas.' (No errors found).

Now we enable remote connections in SQL Server. In SSMS, we connect as an administrator. Right-click the server name and click Properties. Then go to the Connections tab and select the checkbox: Allow remote connections to this server and click OK.





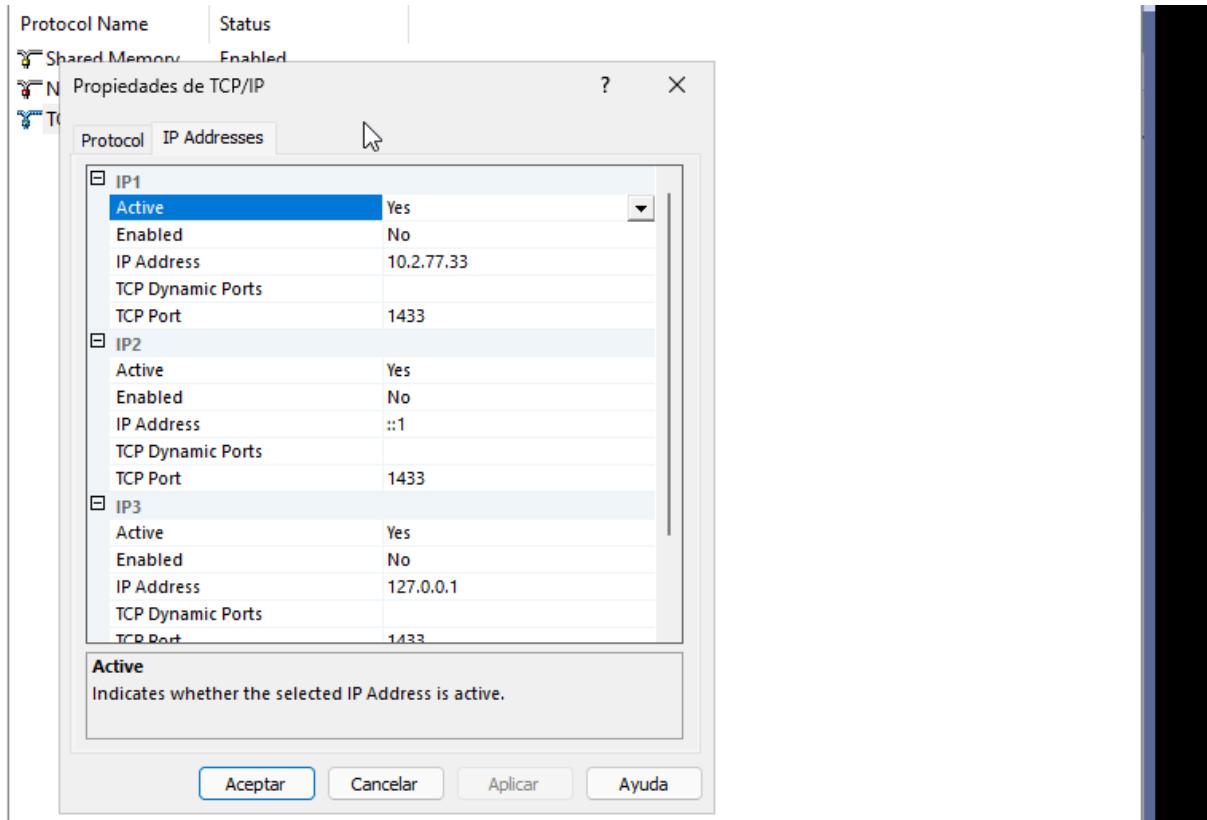
Let's go to SQL Server Configuration Manager, in the left panel, we go to: SQL Server Network Configuration and then Protocols for MSSQLSERVER



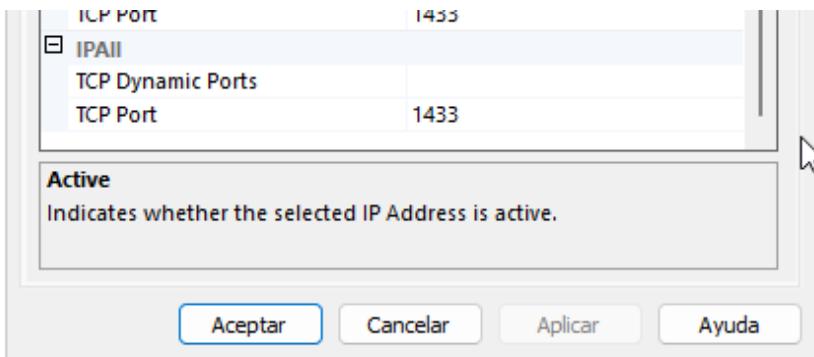
In the right panel, right-click on TCP/IP and click Enable.

Protocol Name	Status
Shared Memory	Enabled
Named Pipes	Disabled
TCP/IP	Enabled

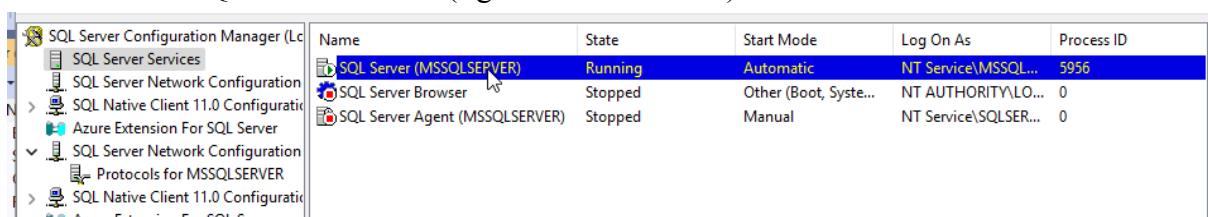
Then, we right-click again and go to Properties and go to the IP Addresses tab.



We download and make sure we have: TCP Port = 1433

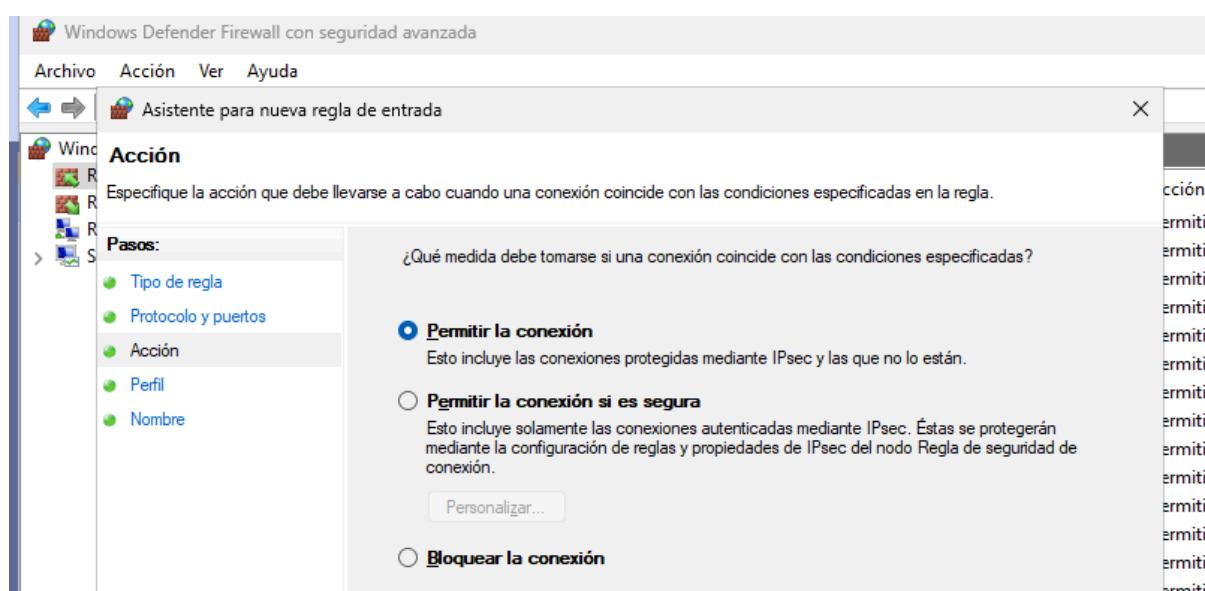
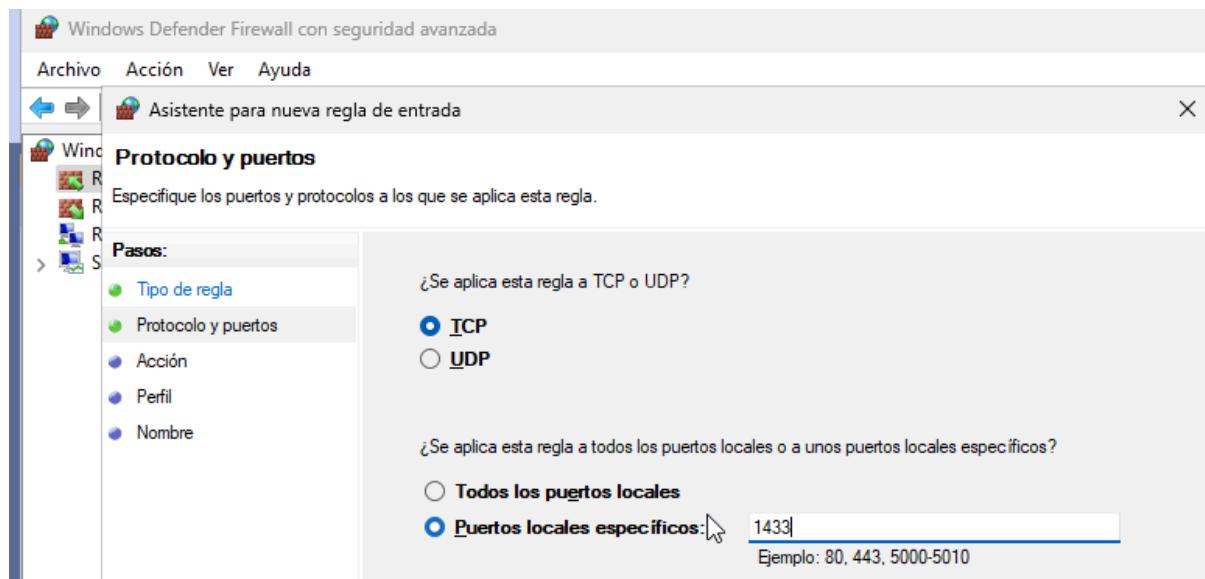
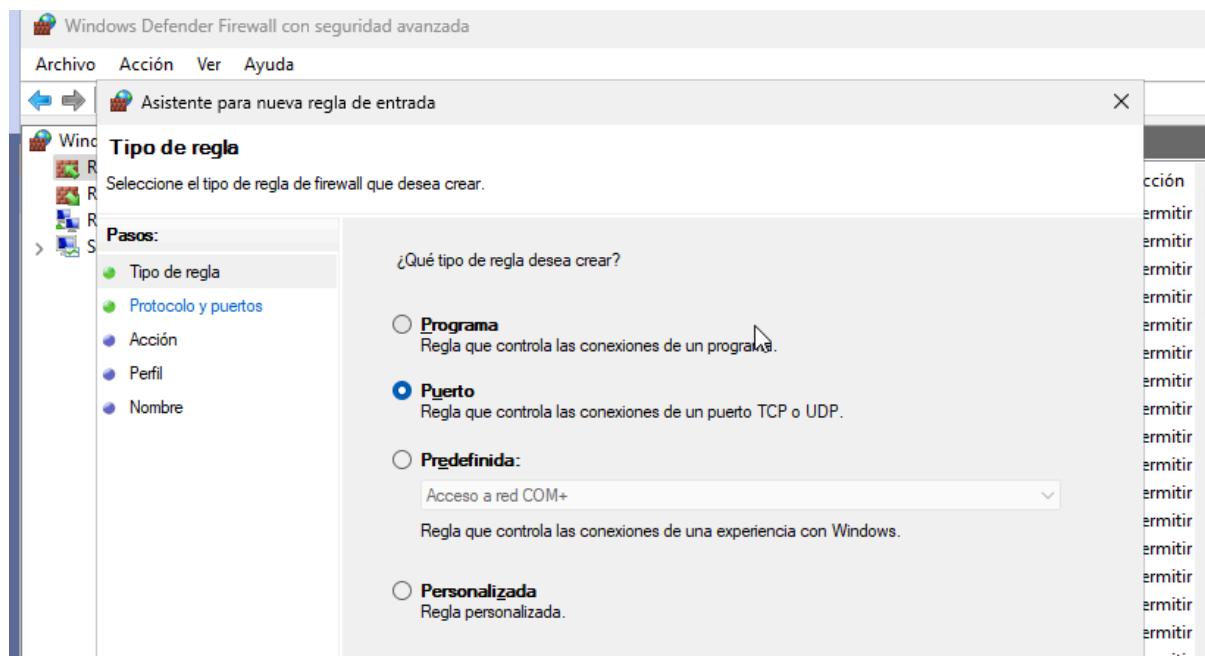


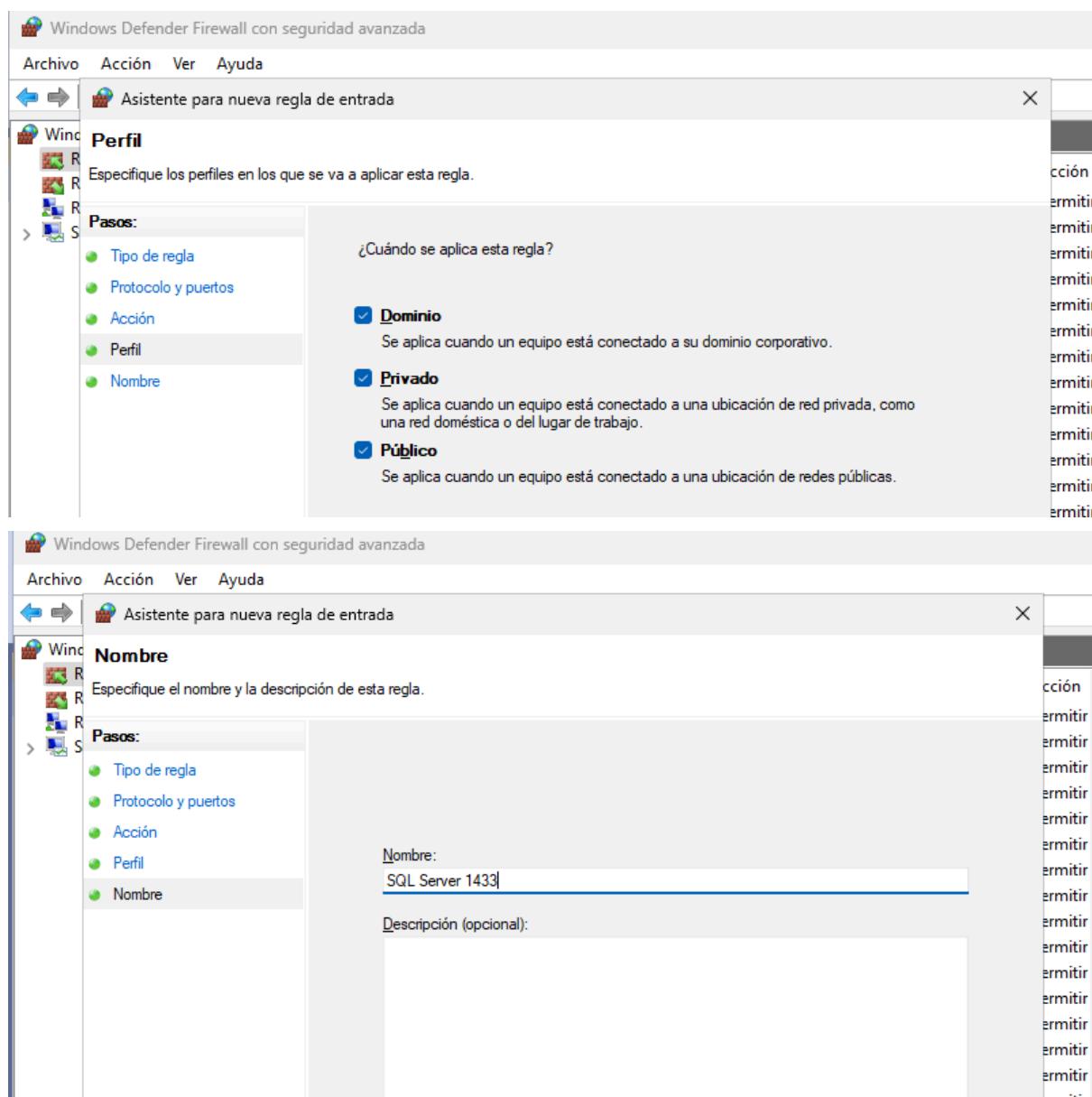
We restart the SQL Server service (right click → Restart).



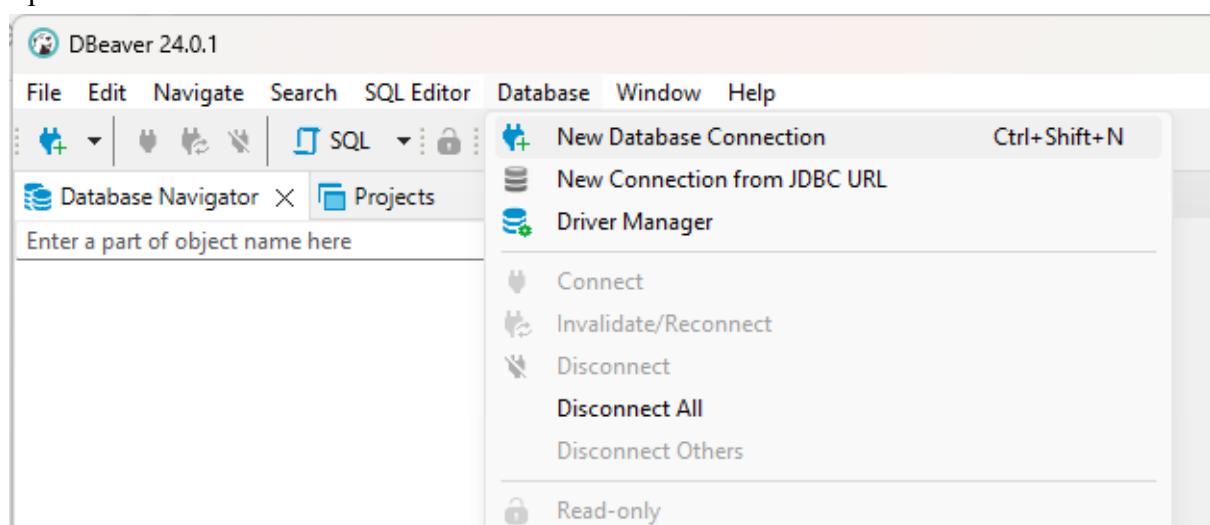
Now we allow port 1433 in the Windows firewall (Control Panel → System and Security → Windows Defender Firewall → Advanced settings.)

In the left bar, select Inbound Rules → New Rule. Fill in the fields as follows:

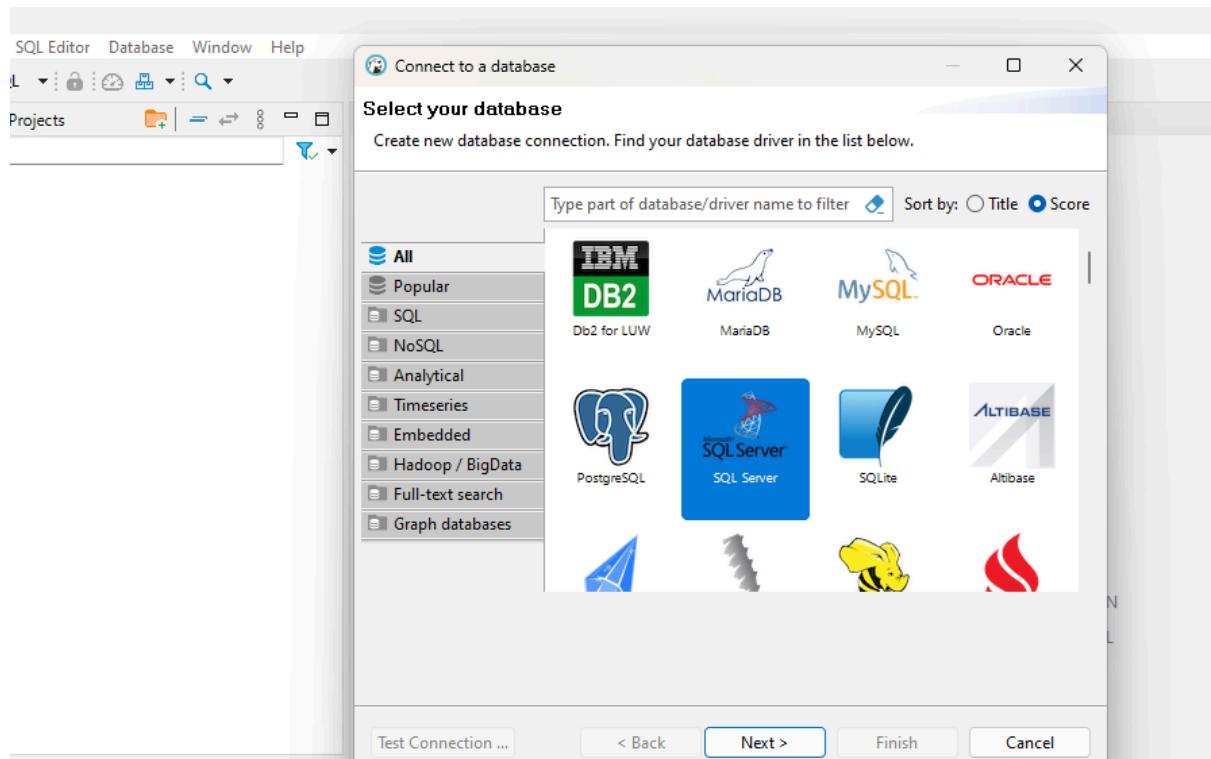




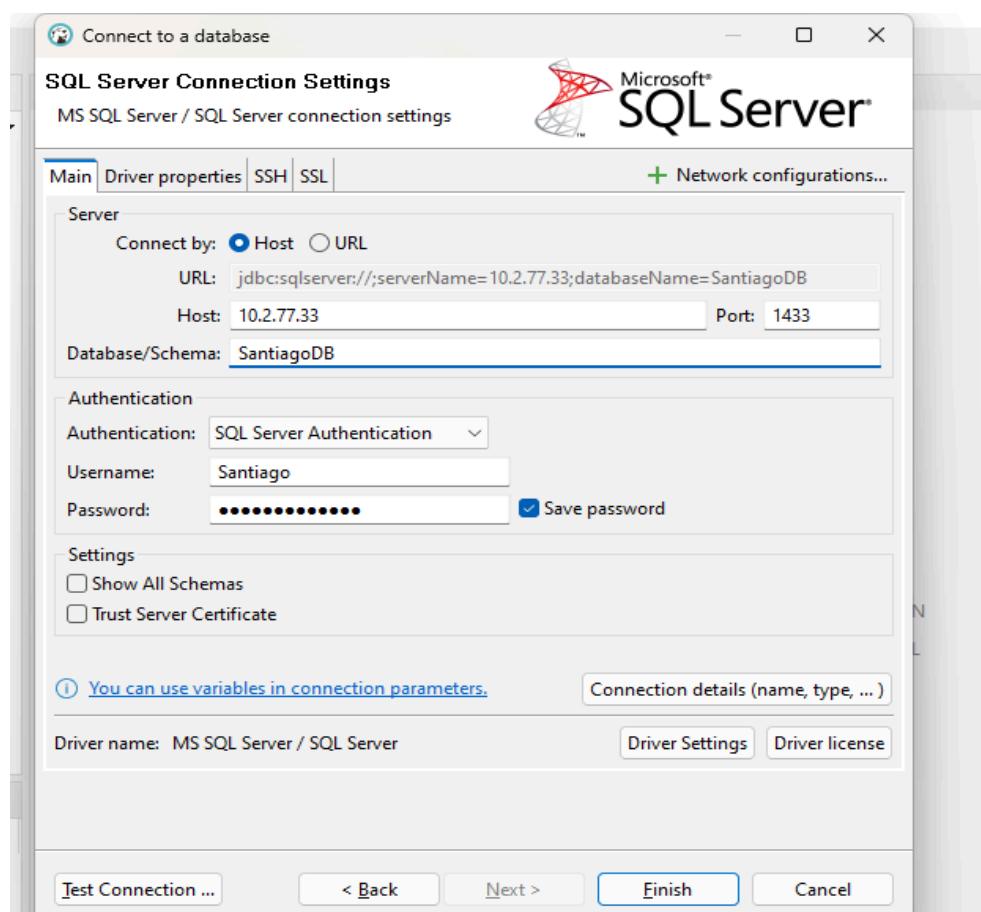
Open DBeaver. Click on New Database Connection.



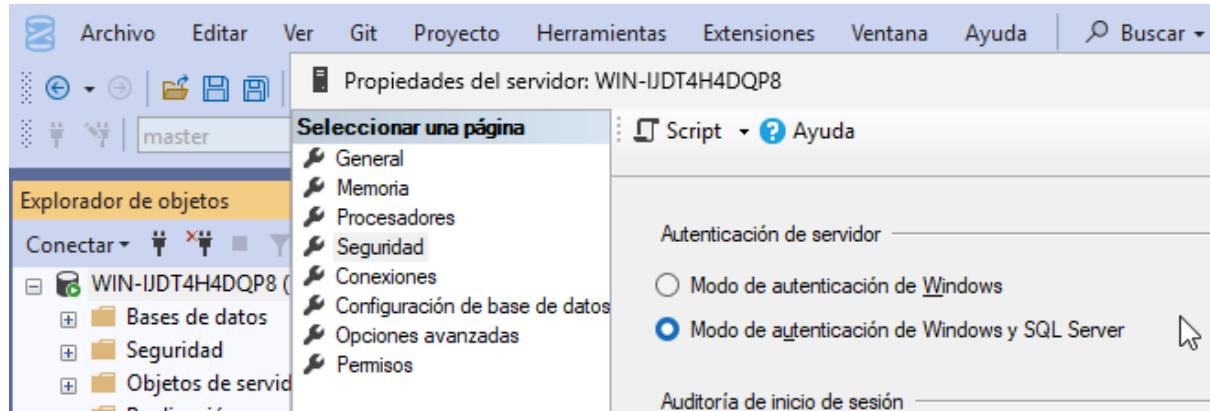
We choose SQL Server



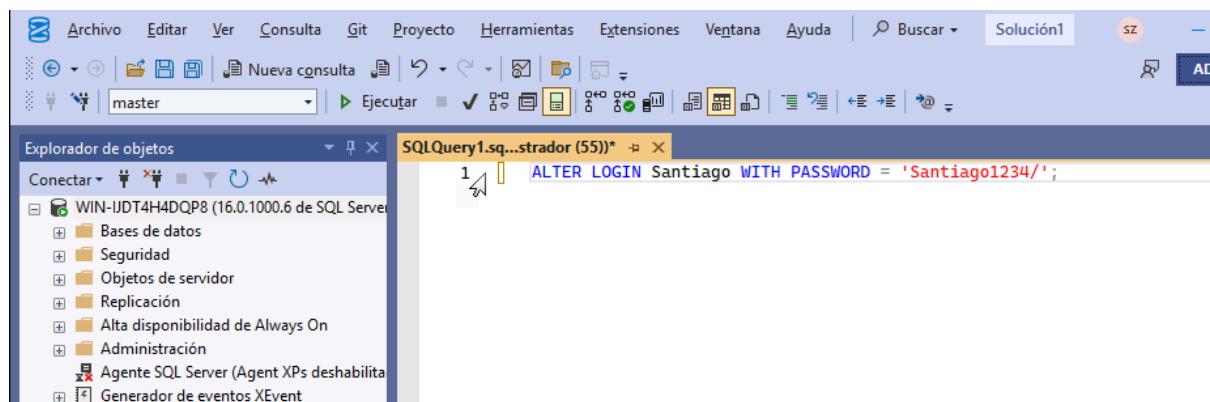
We fill out the form as follows and click on Test Connection.



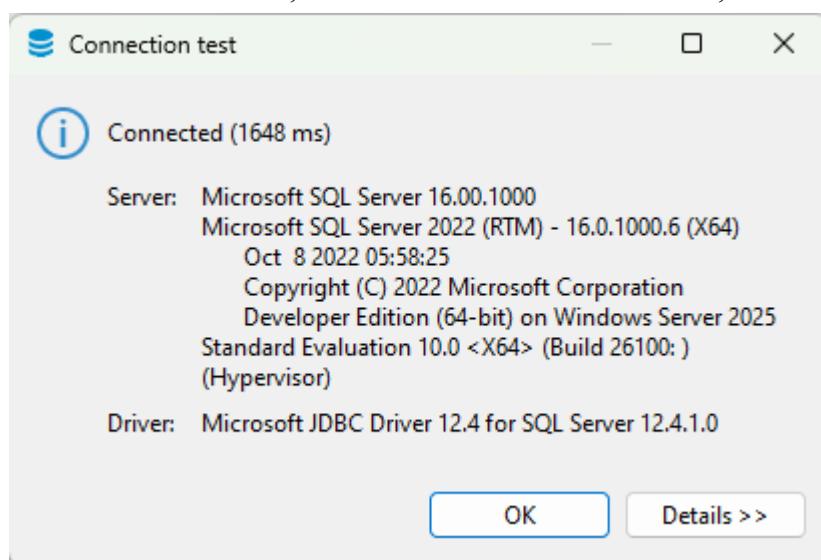
First, open SQL Server Management Studio (SSMS) and log in as Administrator (Windows Authentication). In Object Explorer, right-click your server → Properties. Go to the Security tab and select SQL Server and Windows Authentication mode. Then press OK and restart the server



Execute this before test



Press Test Connection, if the connection was successful, the following will appear:



Then, we verify that the remote connection worked by making a query

DBeaver 24.0.1 - <SantiagoDB> Script

File Edit Navigate Search SQL Editor Database Window Help

Database Navigator Projects SantiagoDB dbo@SantiagoDB Auto SantiagoDB <SantiagoDB> Script

Enter a part of object name here

SantiagoDB - 10.2.77.33:1433

Databases JulianDB SantiagoDB

Schemas dbo

Tables Activity Category Schedule External Tables Views Indexes Procedures Sequences Synonyms Triggers Data Types Database triggers

Security Administrator

Project - General X Name Data Source Bookmarks

SQL \*<SantiagoDB> Script X

SELECT \* FROM Category;

Results 1 X

SELECT \* FROM Category Enter a SQL expression to filter results (use Ctrl+Space)

CategoryID	CategoryName
1	Estudio
2	Ejercicio
3	Ocio

Value X

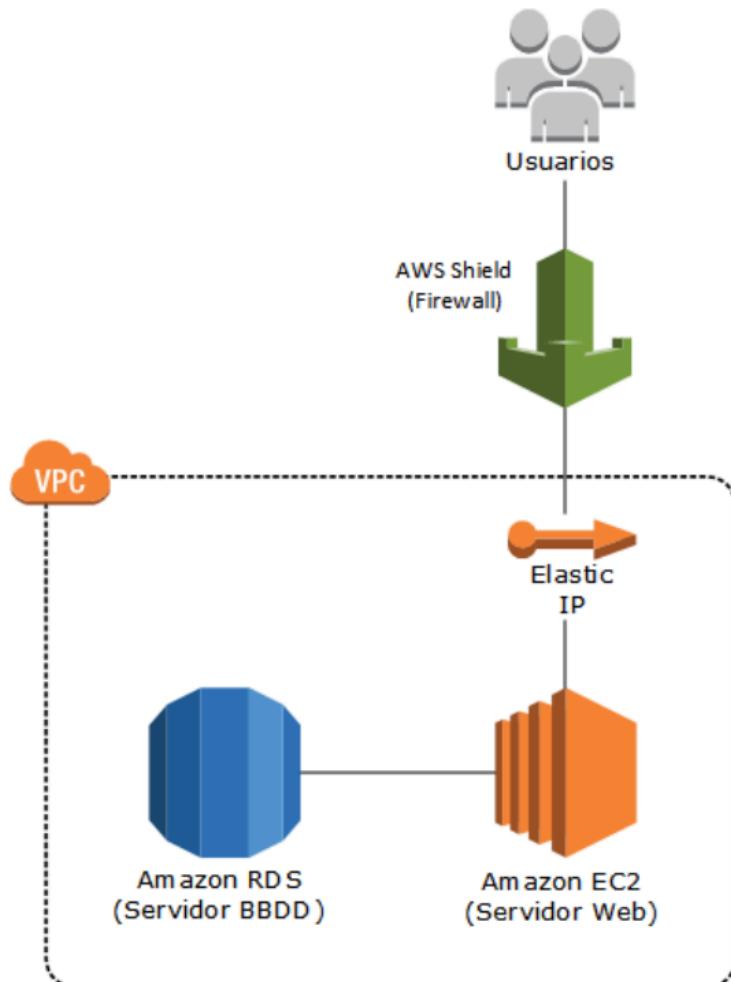
1

# Cloud - AWS - Base Software Installation

Part of the core platform of an organization's computational infrastructure involves web services, which can be hosted either within the company's data center or on a cloud server. These services store the organization's web pages and are accessed by various clients. In this lab, we will implement this service.

## 1. Cloud Web Service Installation

Amazon Elastic Compute Cloud (Amazon EC2) provides the most extensive and versatile computing platform, offering more than 500 instance types and the flexibility to choose the latest processors, storage options, networking, operating systems, and purchasing models, so you can tailor it to perfectly meet your workload requirements. For this lab, we will gain hands-on experience with Amazon Cloud by deploying an EC2 instance and installing a web server.



## 2. Research

- What is an EC2 instance and what is it used for?

An Amazon EC2 (Elastic Compute Cloud) instance is a virtual server in the AWS cloud that provides resizable computing capacity. It is used to host applications, run workloads, perform data processing, or simulate environments without needing physical hardware.

b. What is a VPC, how should it be configured, and what best practices should be considered?

A Virtual Private Cloud (VPC) is a private network within AWS that allows you to isolate and control your resources. It should be configured with subnets, route tables, and security groups to ensure secure communication. Best practices include using private subnets for sensitive resources, enabling flow logs, and applying the principle of least privilege in access control.

c. How can I run multiple systems within an Amazon EC2 environment?

You can run multiple systems in an EC2 environment by launching multiple instances, each configured with its own operating system and applications. You can also use containerization tools like Docker or orchestration platforms like Kubernetes to manage multiple systems efficiently within the same infrastructure.

d. How quickly can I scale the capacity (both up and down) of an EC2 instance?

EC2 instances can scale up or down within minutes. Vertical scaling (changing instance type) can be done by stopping and modifying the instance, while horizontal scaling (adding or removing instances) can be automated using Auto Scaling Groups to adjust capacity dynamically based on demand.

e. How does this service differ from standard hosting services?

Unlike traditional hosting, EC2 offers on-demand, scalable, and flexible infrastructure. You pay only for what you use, have full control over your operating system and configurations, and can automate scaling and deployment—something standard hosting services typically don't provide.

f. What is Amazon RDS?

Amazon RDS (Relational Database Service) is a managed service that simplifies the setup, operation, and scaling of relational databases in the cloud. It supports engines like MySQL, PostgreSQL, and Oracle, and handles tasks such as backups, patching, and replication automatically.

### **3. Configuration**

a. Log in to the AWS Management Console at [awsacademy.instructure.com](https://awsacademy.instructure.com) and locate Lab 5.

The screenshot shows the AWS Academy Dashboard. On the left, there is a sidebar with icons for Account, Dashboard, Courses, Calendar, Inbox, and History. The main area is titled "Dashboard" and contains a large dark box with three dots in the top right corner. Below this box, the text "AWS Academy Learner Lab [1339...]" and "ALLv2ES-LA-LTI13-133966" is visible, along with a small icon.

- b. Navigate to the Modules section, click on "Learner Lab", accept the terms and conditions, and then click on "Start Lab."

The screenshot shows the "Learner Lab" environment. At the top, there is a navigation bar with the URL "ALLv2ES-LA-LTI13-133966 > Módulos > Laboratorio de aprendizaje de AWS Academy > Iniciar el Laboratorio de aprendizaje de AWS Academy". The main interface includes a sidebar with links like "Página de Inicio", "Módulos", "Foros de discusión", "Calificaciones", and "Lucid (pizarra)". The central area has a terminal window showing "eee\_W\_4957100@runweb191100:~\$". To the right, there is a "Learner Lab" panel with a list of topics: Environment Overview, Environment Navigation, Access the AWS Management Console, Region restriction, Service usage and other restrictions, Using the terminal in the browser, Running AWS CLI commands, Using the AWS SDK for Python, Preserving your budget, Accessing EC2 Instances, SSH Access to EC2 Instances, SSH Access from Windows, and SSH Access from a Mac. At the bottom, there are "Anterior" and "Siguiente" buttons.

- c. Once the lab loads, click the "AWS" button in the upper left corner to be redirected to the AWS Console.

The screenshot shows the AWS Management Console homepage. At the top, there's a navigation bar with the AWS logo, a search bar, and account information. Below the navigation bar, the main content area has several sections:

- Visitados recientemente**: Shows a single item: EC2.
- Aplicaciones**: Shows 0 applications. It includes a search bar for "Buscar aplicaciones" and a button to "Crear aplicación".
- Le damos la bienvenida a AWS**: A welcome message.
- AWS Health**: Shows 0 open problems.
- Costo y uso**: Shows current month costs and currency.

At the bottom of the page, there are links for CloudShell, Comentarios, and a footer with copyright information and links to Privacy, Términos, and Preferencias de cookies.

d. Click on EC2.

## Página de inicio de la Consola [Información](#)

### Visitados recientemente [Información](#)



[EC2](#)

e. Select the EC2 Dashboard and then choose the “Launch Instance” option.

f. For this exercise, select the Amazon Linux 2 AMI.

g. Choose the instance type t2.micro.

We can't use t2.micro for this Linux 2 AMI, because this type is not supported by the selected AMI, instead we use t3.micro

What are the different instance types available in Amazon EC2? Why do you think we chose t2.micro?

Amazon EC2 offers several instance types designed for different use cases, such as General Purpose, Compute Optimized, Memory Optimized, Storage Optimized, and Accelerated Computing. Each type provides a specific balance of CPU, memory, storage, and networking capacity. The t2.micro instance is often chosen because it is part of the free tier, low-cost, and suitable for small workloads or testing environments. It provides a good balance of performance and affordability for beginners or lightweight applications.

h. Consult the following documentation related to creating a VPC and create the VPC using the addresses provided by your instructor.

Open the Amazon VPC Console

The screenshot shows the AWS VPC console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and account information. Below the navigation bar is a sidebar titled "Panel de VPC" containing a "Nube virtual privada" section with links for Sus VPCs, Subredes, Tablas de enrutamiento, Puertas de enlace de Internet, Puerta de enlace de Internet de solo salida, Gateways de operador, Conjuntos de opciones de DHCP, Direcciones IP elásticas, Listas de prefijos administradas, Gateways NAT, Interconexiones, and Servidores de ruta. The main content area is titled "Recursos por región" and lists various VPC components with their counts in the Northern Virginia region: VPC (1), Subredes (6), Tablas de enrutamiento (1), Gateways de Internet (1), Gateways de Internet de solo salida (0), Gateways NAT (0), Gateways de cliente (0), Gateways NAT (0), Gateways de Internet (1), Interconexiones de VPC (0), ACL de red (1), Grupos de seguridad (2), and Gateways de cliente (0). On the right side, there are three boxes: "Estado del servicio" (with a link to "Ver los detalles completos del estado del servicio"), "Configuración" (with links to "Bloquear el acceso público", "Zonas", and "Experimentos de la consola"), and "Información adicional" (with links to "Documentación de la VPC", "Todos los recursos de VPC", "Foros", and "Informar de un problema"). A "AWS Network Manager" section is also present at the bottom right.

choose Create VPC

The screenshot shows the AWS VPC Dashboard. At the top, there's a navigation bar with the AWS logo, a search bar, and account information. Below the navigation bar is a sidebar titled "Virtual private cloud" with links for Your VPCs, Subnets, Routing tables, and Internet gateways. The main content area is titled "Resources by region" and shows a summary for the Northern Virginia region: "VPC" (1), "See all regions". There are also "Create VPC" and "Launch EC2 instances" buttons at the top of this section. A note states: "Note: Your instances will be launched in the United States region." The overall layout is clean and modern, typical of AWS interfaces.

For resources to create under VPC setting, choose VPC and more

## Create VPC [Information](#)

A VPC is an isolated part of the AWS Cloud that contains AWS objects, such as

### VPC Configuration

#### Resources to be created [Information](#)

Create only the VPC resource or the VPC and other network resources.

Only the VPC

VPC and more

For the VPC settings, set as follows:

#### Automatic generation of name tags [Information](#)

Enter a value for the Name tag. This value will be used to automatically generate Name tags for all resources in the VPC.

Generate automatically

Lab05\_VPC

#### IPv4 CIDR Block [Information](#)

Determine the initial IP and size of the VPC using CIDR notation.

10.2.77.31/16

65,536 IPs

The CIDR block size must be between /16 and /28.

#### IPv6 CIDR Block [Information](#)

No IPv6 CIDR block

IPv6 CIDR block provided by Amazon

#### Tenure [Information](#)

Predetermined

#### Number of Availability Zones (AZ) [Information](#)

Choose the number of Availability Zones you want to provision subnets in. We recommend having at least two to increase availability.

1 | **2** | 3

#### Preview

##### VPC [Show details](#)

Your AWS virtual network

Lab05\_VPC-vpc

## Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances. Mouse over a resource to highlight the related resources.

### VPC settings

#### Resources to create Info

Create only the VPC resource or the VPC and other networking resources.

 VPC only VPC and more

#### Name tag auto-generation Info

Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.

 Auto-generate

Lab05AYSR\_VPC

#### IPv4 CIDR block Info

Determine the starting IP and the size of your VPC using CIDR notation.

10.2.77.0/24

256 IPs

CIDR block size must be between /16 and /28.

#### IPv6 CIDR block Info

 No IPv6 CIDR block Amazon-provided IPv6 CIDR block

#### Tenancy Info

Default

### Preview

#### VPC Show details

Your AWS virtual network

Lab05AYSR\_VPC-vpc

### Number of public subnets Info

The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.

0 | 2

### Number of private subnets Info

The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.

0 | 2 | 4

### ▼ Customize subnets CIDR blocks

#### Public subnet CIDR block in us-east-1a

10.2.77.0/26

64 IPs

#### Public subnet CIDR block in us-east-1b

10.2.77.64/26

64 IPs

#### Private subnet CIDR block in us-east-1a

10.2.77.128/26

64 IPs

#### Private subnet CIDR block in us-east-1b

10.2.77.192/26

64 IPs

### Preview

#### VPC Show details

Your AWS virtual network

Lab05AYSR\_VPC-vpc

### NAT gateways (\$) Info

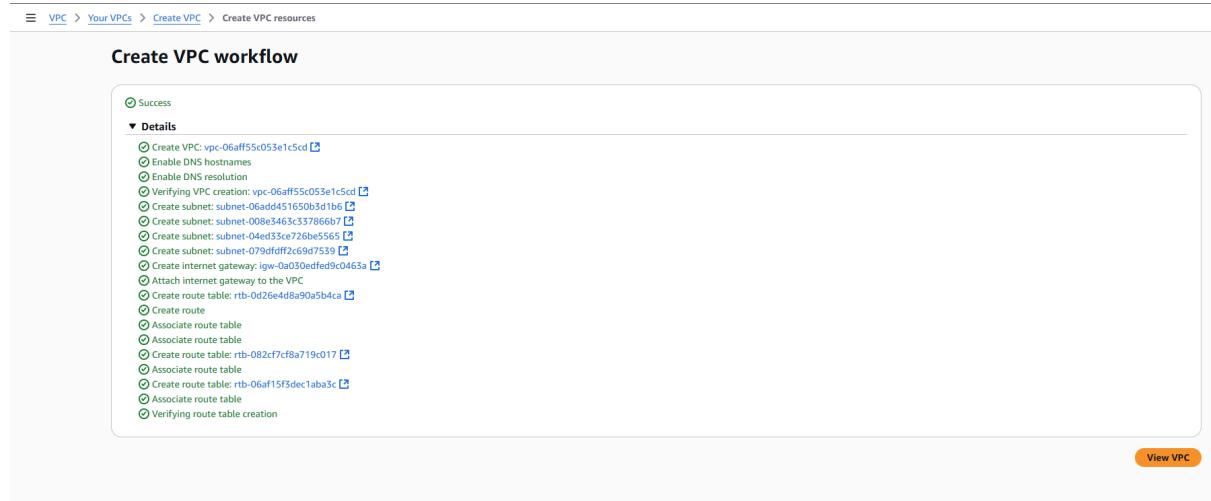
Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway.

 None | In 1 AZ | 1 per AZ

### VPC endpoints Info

Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.

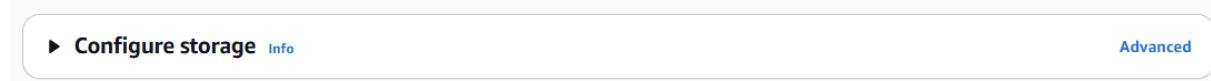
 None | S3 Gateway



- i. On the "Configure Instance Details" page, set the values as specified in the previous step.
- j. Under "Auto-assign Public IP," select Enable.

The screenshot shows the 'Configure Instance Details' page. It includes sections for 'Network settings', 'Auto-assign public IP', 'Firewall (security groups)', and 'Description - required'. In the 'Network settings' section, the VPC is set to 'vpc-06aff55c053e1c5cd (Lab05AYSR\_VPC-vpc)'. Under 'Auto-assign public IP', 'Enable' is selected. In the 'Firewall (security groups)' section, a new security group is being created, with the 'Create security group' option selected. The 'Description - required' field contains the text 'Amazon Linux 2 AMI (HVM), SSD Volume Type (64-bit x86) Operating System-2.0.20250929.2-Autogen'. The 'Inbound Security Group Rules' section shows one rule: 'Security group rule 1 (TCP, 22, 0.0.0.0/0)'. This rule has 'Type' set to 'ssh', 'Protocol' set to 'TCP', 'Port range' set to '22', 'Source type' set to 'Anywhere', and 'Source' set to 'Add CIDR, prefix list or security group'. There is also an optional 'Description' field with the placeholder 'e.g. SSH for admin desktop'.

- k. Click on "Next: Add Storage."



**Configure storage** [Info](#)

1x  GiB  [▼](#) Root volume, Not encrypted

[Add new volume](#)

[Edit](#)

Click refresh to view backup information  
The tags that you assign determine whether the instance will be backed up by any Data Lifecycle Manager policies.

0 x File systems

- l. On the "Add Storage" page, leave the default values and click "Next: Add Tags."
- m. On the "Add Tags" page, click "Add Tag," then enter Name for the Key and tutorial-web-server for the Value.

**Name and tags** [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

**Key** [Info](#) **Value** [Info](#) **Resource types** [Info](#)

[X](#)  [X](#)  [▼](#) [Remove](#)

[Instances](#) [X](#)

[Add new tag](#)

You can add up to 49 more tags.

- n. On the "Configure Security Group" page, choose "Select an existing security group."

**Network settings** [Info](#)

**VPC - required** [Info](#)

vpc-06aff55c053e1c5cd (Lab05AYSR\_VPC-vpc)  
10.2.77.0/24

[C](#)

**Subnet** [Info](#)

subnet-06add451650b3d1b6 Lab05AYSR\_VPC-subnet-public1-us-east-1a  
VPC: vpc-06aff55c053e1c5cd Owner: 975050211860 Availability Zone: us-east-1a (use1-az2)  
Zone type: Availability Zone IP addresses available: 59 CIDR: 10.2.77.0/26

[C](#) [Create new subnet](#)

**Auto-assign public IP** [Info](#)

Enable

**Firewall (security groups)** [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group  Select existing security group

**Common security groups** [Info](#)

Select security groups

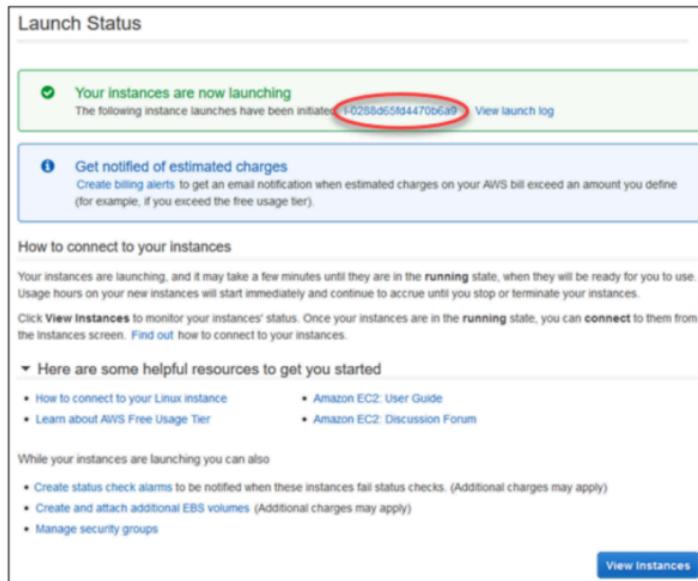
[C](#) [Compare security group rules](#)

**Advanced network configuration**

What are security groups, and what should be considered when creating a security group for a public web server?

**Security groups** in AWS act as **virtual firewalls** that control the **inbound and outbound traffic** to your EC2 instances. They define which types of network connections are allowed to reach your instance (inbound) and which can leave it (outbound).

- o. On the "Review Instance Launch" page, verify your configuration and click "Launch." Note: To launch an EC2 instance, click "Launch Instances." On the "Launch Status" page, note the identifier of the new EC2 instance (e.g., i-0288d65fd4470b6a9).



Result:

**▼ Summary**

**Number of instances** | **Info**

1

**Software Image (AMI)**  
Amazon Linux 2 AMI (HVM), SSD Volume Type (64-bit x86) Operating System  
ami-091d7d61336a4c68f

**Virtual server type (instance type)**  
t3.micro

**Firewall (security group)**  
-

**Storage (volumes)**  
1 volume(s) - 8 GiB

**Cancel** **Launch instance** **Preview code**

p. To locate the created instance, click on “View Instances.”

The screenshot shows the AWS EC2 Instances page. On the left, a sidebar navigation includes: Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, and Network Interfaces. The main content area displays the "Instance summary for i-0045f149e41dc3656 (tutorial-web-server)" with the following details:

- Public IPv4 address:** 44.211.195.123 | [open address](#)
- Instance state:** Running
- Private IP/DNS name (IPv4 only):** ip-10-2-77-15.ec2.internal
- Instance type:** t3.micro
- VPC ID:** vpc-06aff55c053e1c5cd (Lab0SAYSR\_VPC-vpc)
- Subnet ID:** subnet-06add451650b3d1b6 (Lab0SAYSR\_VPC-subnet-public1-us-east-1a)
- Instance ARN:** arn:aws:ec2:us-east-1:975050211860:instance/i-0045f149e41dc3656
- IMDSv2:** Optional (EC2 recommends setting IMDSv2 to required) | [Learn more](#)
- IAM Role:** -
- Operator:** -
- Details Tab:** Shows AMI ID (ami-091d7d61336a4c68f), Monitoring (disabled), and AMI name.
- Status and alarms Tab:** Not visible in the screenshot.
- Monitoring Tab:** Not visible in the screenshot.
- Security Tab:** Not visible in the screenshot.
- Networking Tab:** Not visible in the screenshot.
- Storage Tab:** Not visible in the screenshot.
- Tags Tab:** Not visible in the screenshot.
- Platform details:** Linux/UNIX
- Termination protection:** Managed (false)

q. Install a web server on the newly created instance. Note: Refer to the instructions on how to connect to the instance in order to install the web service.

First, go to the EC2 Console:

The screenshot shows the AWS EC2 Instance Details page for the instance i-0045f149e41dc3656. The terminal session output is as follows:

```

Amazon Linux 2
AL2 End of Life is 2026-06-30.
A newer version of Amazon Linux is available!
Amazon Linux 2023, GA and supported until 2028-03-15.
https://aws.amazon.com/linux/amazon-linux-2023/
4 package(s) needed for security, out of 4 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-10-2-77-15 ~]$ 
```

At the bottom, it shows the instance ID (i-0045f149e41dc3656 (tutorial-web-server)), Public IPs (34.201.66.70), and Private IPs (10.2.77.15).

Once in the console, we execute the command `sudo yum update -y`, to update system packages.



We install Apache httpd

```
[ec2-user@ip-10-2-77-15 ~]$ sudo yum install -y httpd
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
| 3.6 kB 00:00:00

Resolving Dependencies
--> Running transaction check
----> Package httpd.x86_64 0:2.4.65-1.amzn2.0.2 will be installed
--> Processing Dependency: httpd-filesystem = 2.4.65-1.amzn2.0.2 for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Processing Dependency: httpd-tools = 2.4.65-1.amzn2.0.2 for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Processing Dependency: /etc/httpd/me.types for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Processing Dependency: httpd-filesystem for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Processing Dependency: mod_http2 for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Processing Dependency: system-logos-httplib for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Processing Dependency: libapr-1.so.0() (64bit) for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Processing Dependency: libaprutil-1.so.0() (64bit) for package: httpd-2.4.65-1.amzn2.0.2.x86_64
--> Running transaction check
--> Package apr.x86_64 0:1.7.2-1.amzn2.0.1 will be installed
--> Package apr-util.x86_64 0:1.6.3-1.amzn2.0.1 will be installed
--> Processing Dependency: apr-util-bdb(x86-64) for package: apr-util-1.6.3-1.amzn2.0.1.x86_64
--> Package generic-logos-httplib.noarch 0:18.0-0.4.amzn2 will be installed
--> Package httpd-filesystem.noarch 0:2.4.65-1.amzn2.0.2 will be installed
--> Package httpd-tools.x86_64 0:2.4.65-1.amzn2.0.2 will be installed
--> Package mailcap.noarch 0:2.1.41-2.amzn2 will be installed
--> Package mod_ssl.x86_64 0:1.15.19-1.amzn2.0.2 will be installed
--> Running transaction check
----> Package apr-util-bdb.x86_64 0:1.6.3-1.amzn2.0.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
== Package Arch Version Repository

```



```

9  Installing : httpd-2.4.65-1.amzn2.0.2.x86_64          9/
9  Verifying  : apr-1.7.2-1.amzn2.0.1.x86_64           1/
9  Verifying  : apr-util-bdb-1.6.3-1.amzn2.0.1.x86_64      2/
9  Verifying  : httpd-filesystem-2.4.65-1.amzn2.0.2.noarch   3/
9  Verifying  : httpd-tools-2.4.65-1.amzn2.0.2.x86_64       4/
9  Verifying  : mod_http2-1.15.19-1.amzn2.0.2.x86_64        5/
9  Verifying  : apr-util-1.6.3-1.amzn2.0.1.x86_64         6/
9  Verifying  : mailcap-2.1.41-2.amzn2.noarch            7/
9  Verifying  : generic-logos-httpd-18.0.0-4.amzn2.noarch    8/
9  Verifying  : httpd-2.4.65-1.amzn2.0.2.x86_64          9/
9

Installed:
  httpd.x86_64 0:2.4.65-1.amzn2.0.2

Dependency Installed:
  apr.x86_64 0:1.7.2-1.amzn2.0.1           apr-util.x86_64 0:1.6.3-1.amzn2.0.1           apr-util-bdb.x86_64 0:1.6.3-1.amzn2.0.1
  generic-logos-httpd.noarch 0:18.0.0-4.amzn2          httpd-filesystem.noarch 0:2.4.65-1.amzn2.0.2           httpd-tools.x86_64 0:2.4.65-1.amzn2.0.2
  mailcap.noarch 0:2.1.41-2.amzn2                mod_http2.x86_64 0:1.15.19-1.amzn2.0.2

Complete!
[ec2-user@ip-10-2-77-15 ~]$
```

## We activate and start the web service

```
[ec2-user@ip-10-2-77-15 ~]$ sudo systemctl enable httpd
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-10-2-77-15 ~]$ sudo systemctl start httpd
[ec2-user@ip-10-2-77-15 ~]$
```

## Next, we check that the service is running

```
[ec2-user@ip-10-2-77-15 ~]$ sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2025-10-15 19:52:22 UTC; 1min 21s ago
     Docs: man:httpd.service(8)
 Main PID: 5835 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Bytes served/sec: 0 B/sec"
   CGroup: /system.slice/httpd.service
           ├─5835 /usr/sbin/httpd -DFOREGROUND
           ├─5836 /usr/sbin/httpd -DFOREGROUND
           ├─5837 /usr/sbin/httpd -DFOREGROUND
           ├─5855 /usr/sbin/httpd -DFOREGROUND
           ├─5860 /usr/sbin/httpd -DFOREGROUND
           └─5866 /usr/sbin/httpd -DFOREGROUND

Oct 15 19:52:22 ip-10-2-77-15.ec2.internal systemd[1]: Starting The Apache HTTP Server...
Oct 15 19:52:22 ip-10-2-77-15.ec2.internal systemd[1]: Started The Apache HTTP Server.
[ec2-user@ip-10-2-77-15 ~]$
```

Finally search <http://34.201.66.70> and see the result that the web server is running correctly



This page is used to test the proper operation of the Apache HTTP server after it has been installed. If you can read this page, it means that the Apache HTTP server installed at this site is working properly.

### If you are a member of the general public:

The fact that you are seeing this page indicates that the website you just visited is either experiencing problems, or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

### If you are the website administrator:

You may now add content to the directory `/var/www/html/`. Note that until you do so, people visiting your website will see this page, and not your content. To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

You are free to use the image below on web sites powered by the Apache HTTP Server:



r. Finally, add content to the web server that connects to your Amazon database instance.

First we enter the site directory

```
[ec2-user@ip-10-2-77-15 ~]$ cd /var/www/html  
[ec2-user@ip-10-2-77-15 html]$
```

we create an index.html file

```
GNU nano 2.9.8                                         index.html  
  
^G Get Help    ^C Write Out   ^W Where Is   ^K Cut Text   ^J Justify   [ New File ]  
^X Exit        ^R Read File   ^V Replace    ^U Uncut Text  ^T To Spell   ^C Cur Pos    M-U Undo  
M-A Mark Text  M-D Copy Text M-W WhereIs Next M-V Previous  
^A Go To Line  M-E Redo     M-W WhereIs Prev M-N Next  
  
^G Get Help    ^C Write Out   ^W Where Is   ^K Cut Text   ^J Justify   [ New File ]  
^X Exit        ^R Read File   ^V Replace    ^U Uncut Text  ^T To Spell   ^C Cur Pos    M-U Undo  
M-A Mark Text  M-D Copy Text M-W WhereIs Next M-V Previous  
^A Go To Line  M-E Redo     M-W WhereIs Prev M-N Next
```

we add content to the file and save

```
GNU nano 2.9.8                                         M index.html  
modified  
<html>  
    <head><title>My AWS Web Server</title></head>  
    <body>  
        <h1>Welcome to my Amazon EC2 web server!</h1>  
        <p>This web server is running on an EC2 instance in AWS.</p>  
    </body>  
</html>
```

We refresh the page to see the result



## Conclusions

---

This laboratory provided a comprehensive and practical understanding of how to install, configure, and manage one of the most essential components of any IT infrastructure: a database management system. Through the installation and setup of PostgreSQL on a Linux Slackware virtual machine, we not only learned the technical steps involved but also understood the underlying principles of system administration, user management, and data organization. Each phase of the process—from downloading the PostgreSQL package and configuring the system user “postgres,” to initializing the data directory and starting the service—reinforced our understanding of how Linux interacts with database processes and how administrative control is established within multi-user environments.

Creating individual users for each group member allowed us to explore PostgreSQL’s security and access control mechanisms, an essential aspect of database management in both academic and professional settings. By assigning ownership of databases to specific users, we ensured that each had isolated access to their respective data, reflecting best practices in real-world scenarios where data confidentiality and role-based permissions are critical. Furthermore, designing and populating a relational database containing tourist sites in Colombia gave us the opportunity to apply database modeling concepts, including primary and foreign keys, normalization, and relationships between entities. These concepts are fundamental to maintaining data consistency and preventing redundancy in structured datasets.

The insertion of real data into the created tables also gave us a hands-on perspective of how queries and transactions operate within a relational model. Executing SQL commands such as CREATE TABLE, INSERT and SELECT demonstrated the core operations of data definition, manipulation, and retrieval that form the foundation of modern database systems. Moreover, the use of PostgreSQL’s command-line interface (psql) allowed us to interact directly with the DBMS, reinforcing the importance of command-line proficiency in server environments where graphical tools may not be available. This practical exposure not only improved our technical fluency but also enhanced our problem-solving skills when dealing with potential errors or configuration issues.

In addition, this exercise strengthened our understanding of how database systems integrate into broader networked infrastructures. The PostgreSQL server, like any networked service, relies on communication protocols and proper configuration to ensure connectivity, security, and reliability. The combination of database setup with prior Wireshark-based protocol analysis provided a holistic view of how data travels through layers of the network—from physical transmission to application-level interpretation. This reinforced the interdependence between networking and database management, highlighting how both must work seamlessly for systems to function efficiently in enterprise and cloud environments.

Ultimately, this lab bridged theoretical knowledge with practical experience, aligning concepts from operating systems, networking, and database theory into a cohesive and realistic application. By successfully completing the installation and configuration of PostgreSQL, we demonstrated the ability to deploy a fully functional database system from scratch, apply access control policies, model relational data structures, and insert and retrieve meaningful information. These competencies form the foundation for more advanced studies in system administration, networked applications, and cloud computing. The experience also emphasized the importance of precision, attention to detail, and documentation—key habits that define effective engineers and IT professionals in real-world environments.

## Bibliography

---

PostgreSQL Global Development Group. (2024). *PostgreSQL 14 Documentation*. Retrieved from <https://www.postgresql.org/docs/14/>

SlackBuilds.org. (2024). *PostgreSQL SlackBuild Script for Slackware 15.0*. Retrieved from <https://slackbuilds.org/repository/15.0/system/postgresql/>

The Linux Documentation Project. (2024). *Introduction to Linux System Administration*. Retrieved from <https://tldp.org/LDP/intro-linux/html/>

The PostgreSQL Wiki. (2024). *PostgreSQL Installation on Linux*. Retrieved from [https://wiki.postgresql.org/wiki/Detailed\\_installation\\_guides](https://wiki.postgresql.org/wiki/Detailed_installation_guides)

Slackware Linux Project. (2024). *Official Slackware Linux Documentation*. Retrieved from <https://docs.slackware.com/>

DigitalOcean Community Tutorials. (2024). *How To Install and Use PostgreSQL on Linux*. Retrieved from <https://www.digitalocean.com/community/tags/postgresql>

Wireshark Foundation. (2024). *Wireshark User's Guide*. Retrieved from <https://www.wireshark.org/docs/>

GeeksforGeeks. (2024). *Introduction to Database Management Systems (DBMS)*. Retrieved from <https://www.geeksforgeeks.org/introduction-of-dbms-database-management-system-set-1/>

TutorialsPoint. (2024). *PostgreSQL – Create Database, Users and Tables*. Retrieved from <https://www.tutorialspoint.com/postgresql/>

IBM Knowledge Center. (2024). *Understanding Database Architecture and Security Concepts*. Retrieved from <https://www.ibm.com/docs/en/>

