ESCUELA COLOMBIANA DE INGENIERÍA JULIO GARAVITO
REDES DE COMPUTADORES
LABORATORIO N°. 3

Integrantes:

- **Ana Gabriela Silva Briceño.**
- **Juan Sebastián Mina Echavarría.**

MARCO TEÓRICO.

PostgreSQL, o simplemente Postgres para darle un nombre más pintoresco, es un sistema de código abierto de administración de bases de datos del tipo relacional, aunque también es posible ejecutar consultas que sean no relaciones. En este sistema, las consultas relacionales se basan en SQL, mientras que las no relacionales hacen uso de JSON.

Como decíamos, se trata de un sistema de código abierto y además gratuito, y su desarrollo es llevado adelante por una gran comunidad de colaboradores de todo el mundo que día a día ponen su granito de arena para hacer de este sistema una de las opciones más sólidas a nivel de bases de datos.

MariaDB es un sistema gestor de bases de datos (SGBD), es decir, un conjunto de programas que permiten modificar, almacenar, y extraer información de una base de datos. Disponiendo de otro tipo de funcionalidades como la administración de usuarios, y recuperación de la información si el sistema se corrompe, entre otras.

MariaDB surge a raíz de la compra, de la compañía desarrolladora de otro (SGBD) llamado MySQL, por la empresa Sun Microsystems. El desarrollador original, decide tomar el código fuente original de MySQL y genera un derivado con mejoras y cambios a los que llama MariaDB. Perimitiendo así la existencia de una versión de este producto con licencia GPL (General Public License).

A lo largo de los años las comparaciones de PostgreSQL y MySQL no han parado de crecer, y es que se trata de los sistemas de bases de datos relacionales más usados a nivel mundial, aunque claro tienen algunas diferencias entre sí que hacen que los usuarios se inclinen por uno o por otro.

PostgreSQL como ya decíamos sigue siendo un sistema de código libre y totalmente gratuito que está en manos de una comunidad. Con MySQL la historia fue similar durante un buen tiempo, hasta que quedó en manos de Oracle. Actualmente sigue existiendo una versión gratuita de MySQL, aunque eso solo será hasta que Oracle así lo quiera. Afortunadamente existen alternativas MySQL extremadamente similares como MariaDB, por si algún día necesitas un reemplazo.

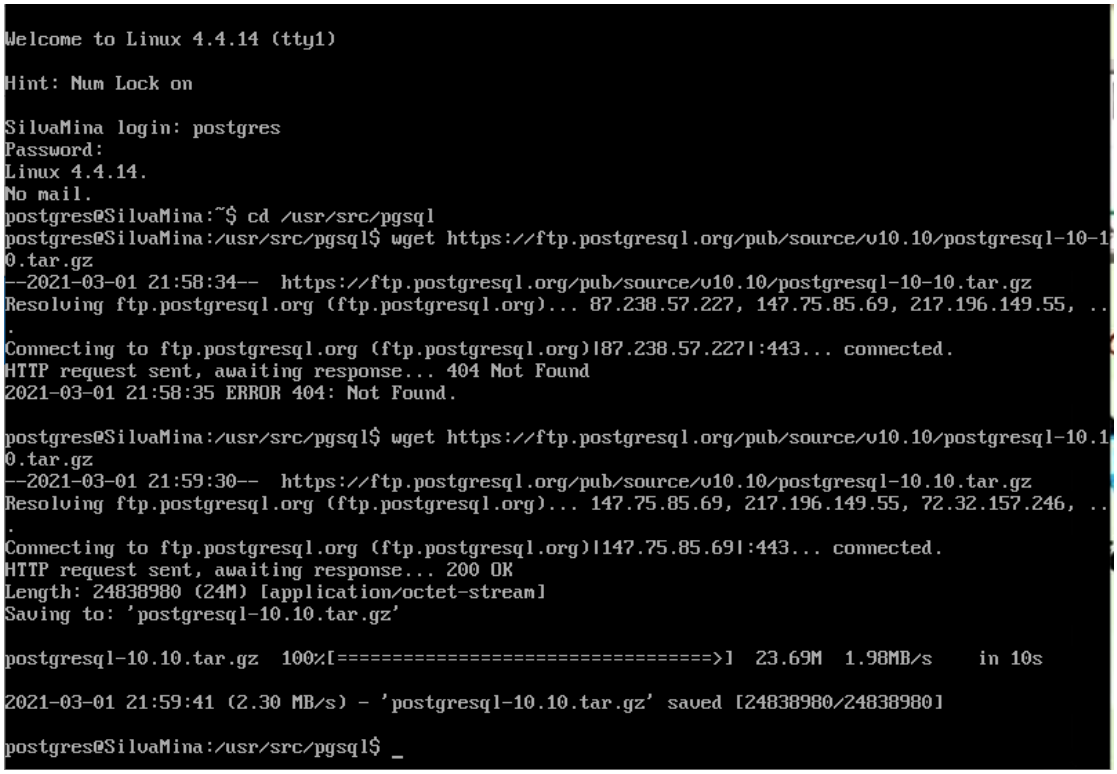
En lo que refiere a las consultas o queries, las soportadas por MySQL tienden a ser más simples, mientras que PostgreSQL soporta queries más complejas, lo cual le ha valido el título del sistema de bases de datos relacionales más avanzado del mercado.

En términos de rendimiento cada uno tiene su situación para brillar: MySQL es mejor si se manejan bajos volúmenes de datos, es decir, si tenemos bases de datos pequeñas o medianas a las cuales se hagan una cantidad de consultas no muy alta. Por su parte, PostgreSQL es mejor a la hora de manejar volúmenes de datos grandes y se suele usar más cuando se tienen bases de datos grandes y con alta cantidad de consultas.

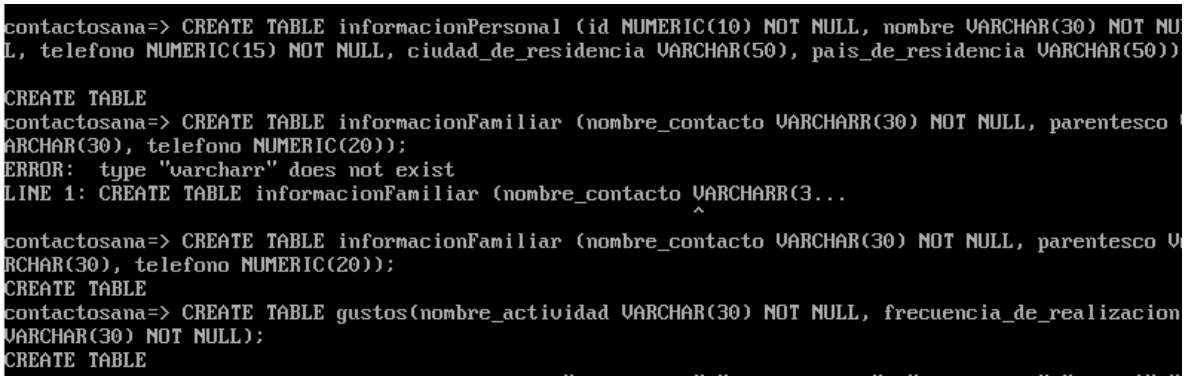
INSTALACIÓN DE SOFTWARE BASE

1. PostgreSQL – Slackware Linux

- Instalación



- Creación e inserción de datos.



```
contactosana=> INSERT INTO informacionPersonal VALUES(1192793544,'Ana Gabriela',3015645037,'Bogotá',
'Colombia');
INSERT 0 1
contactosana=> INSERT INTO informacionPersonal VALUES(1193156116,'Juan Andres',3143609112,'Bogotá','
Colombia');
INSERT 0 1
contactosana=> INSERT INTO informacionFamiliar VALUES('Alfonso Silva','Padre',3204898552);
INSERT 0 1
contactosana=> INSERT INTO informacionFamiliar VALUES('Gloria Machuca','Madre',3204878592);
INSERT 0 1
contactosana=> INSERT INTO gustos VALUES('yoga','diario');
INSERT 0 1
contactosana=> INSERT INTO gustos VALUES('Videojuegos','diario');
INSERT 0 1
```

```
contactosana=> SELECT * FROM informacionPersonal;
     id      |    nombre    |  telefono  | ciudad_de_residencia | pais_de_residencia
-------------+--------------+------------+----------------------+--------------------
 1192793544 | Ana Gabriela | 3015645037 | Bogotá               | Colombia
 1193156116 | Juan Andres  | 3143609112 | Bogotá               | Colombia
(2 rows)

contactosana=> SELECT * FROM informacionFamiliar;
 nombre_contacto | parentesco |  telefono
-----------------+------------+------------
 Alfonso Silva   | Padre      | 3204898552
 Gloria Machuca  | Madre      | 3204878592
(2 rows)

contactosana=> SELECT * FROM gustos;
 nombre_actividad | frecuencia_de_realizacion
------------------+---------------------------
 yoga             | diario
 Videojuegos      | diario
(2 rows)
```

2. MariaDB – FreeBSD

- Creación bases de datos.

```
root@localhost [mysql]> show databases;
+--------------------+
| Database           |
+--------------------+
| Ana                |
| Sebastian          |
| information_schema |
| mysql              |
| performance_schema |
| test               |
+--------------------+
```

- Creación usuarios – permisos de usuarios. (SilvaBriceno tiene todos lo permisos sobre la bese de datos Ana y puede hacer la función SELECT sobre la base de datos Sebastian, MinaEchavarria tiene todos los premisos sobre la base de datos Sebastian y sobre la base de datos Ana solo tiene el permiso DROP.

```
root@localhost [mysql]> CREATE USER "SilvaBriceno"@"localhost" IDENTIFIED BY "12
34";
Query OK, 0 rows affected (0.001 sec)

root@localhost [mysql]> CREATE USER "MinaEchavarria"@"localhost" IDENTIFIED BY "
1234";
Query OK, 0 rows affected (0.000 sec)

root@localhost [mysql]> GRANT ALL PRIVILEGES ON Ana.* TO "SilvaBriceno"@"localho
st";
Query OK, 0 rows affected (0.001 sec)

root@localhost [mysql]> GRANT ALL PRIVILEGES ON Sebastian.* TO "MinaEchavarria"@
"localhost";
Query OK, 0 rows affected (0.000 sec)
```

```
root@localhost [mysql]> GRANT SELECT ON Ana.* TO "MinaEchavarria"@"localhost";
Query OK, 0 rows affected (0.000 sec)

root@localhost [mysql]> GRANT DROP ON Sebastian.* TO "SilvaBriceno"@"localhost";

Query OK, 0 rows affected (0.000 sec)
```

```
root@localhost [mysql]> select User from mysql.user;
+----------------+
| User           |
+----------------+
|                |
| MinaEchavarria |
| SilvaBriceno   |
| mariadb.sys    |
| mysql          |
| root           |
|                |
+----------------+
7 rows in set (0.005 sec)
```

```
root@localhost [mysql]> SHOW GRANTS FOR "SilvaBriceno"@"localhost";
+-------------------------------------------------------------------------------
-------------------------------------+
| Grants for SilvaBriceno@localhost
                                     |
+-------------------------------------------------------------------------------
-------------------------------------+
| GRANT USAGE ON *.* TO `SilvaBriceno`@`localhost` IDENTIFIED BY PASSWORD '*A4B6
157319038724E3560894F7F932C8886EBFCF' |
| GRANT ALL PRIVILEGES ON `Ana`.* TO `SilvaBriceno`@`localhost`
                                     |
| GRANT DROP ON `Sebastian`.* TO `SilvaBriceno`@`localhost`
                                     |
+-------------------------------------------------------------------------------
-------------------------------------+
3 rows in set (0.000 sec)
```

```
root@localhost [mysql]> SHOW GRANTS FOR "MinaEchavarria"@"localhost";
+-----------------------------------------------------------------------------------
-----------------------------------------+
| Grants for MinaEchavarria@localhost
                                         |
+-----------------------------------------------------------------------------------
-----------------------------------------+
| GRANT USAGE ON *.* TO `MinaEchavarria`@`localhost` IDENTIFIED BY PASSWORD '*A4
B6157319038724E3560894F7F932C8886EBFCF' |
| GRANT ALL PRIVILEGES ON `Sebastian`.* TO `MinaEchavarria`@`localhost`
                                         |
| GRANT SELECT ON `Ana`.* TO `MinaEchavarria`@`localhost`
                                         |
+-----------------------------------------------------------------------------------
-----------------------------------------+
3 rows in set (0.000 sec)
```

- Creación de tablas e inserción de datos sobre la base de datos Ana.

Primera tabla.

```
root@localhost [(none)]> use Ana;
Database changed
root@localhost [Ana]> create table ViajesAna (lugar varchar(100) primary key not
 null, pais varchar(100) not null, descripcion varchar(200) not null);
Query OK, 0 rows affected (0.046 sec)

root@localhost [Ana]> describe ViajesAna
    -> ;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| lugar       | varchar(100) | NO   | PRI | NULL    |       |
| pais        | varchar(100) | NO   |     | NULL    |       |
| descripcion | varchar(200) | NO   |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
3 rows in set (0.001 sec)
```

```
root@localhost [Ana]> SELECT * FROM ViajesAna;
+------------+---------+-----------------+
| lugar      | pais    | descripcion     |
+------------+---------+-----------------+
| Montreal   | Canada  | Lugar precioso  |
| Nueva York | USA     | La gran manzana |
| Paris      | Francia | Ciudad del amor |
+------------+---------+-----------------+
3 rows in set (0.002 sec)
```

Segunda tabla.

```
root@localhost [Ana]> create table CondicionesLugar (lugar varchar(100) not null
, clima varchar(30) not null, estaciones varchar(2), vestuario_recomendado varch
ar(200), PRIMARY KEY (lugar), FOREIGN KEY (lugar) REFERENCES ViajesAna(lugar));
Query OK, 0 rows affected (0.007 sec)

root@localhost [Ana]> describe CondicionesLugar
    -> ;
+----------------------+--------------+------+-----+---------+-------+
| Field                | Type         | Null | Key | Default | Extra |
+----------------------+--------------+------+-----+---------+-------+
| lugar                | varchar(100) | NO   | PRI | NULL    |       |
| clima                | varchar(30)  | NO   |     | NULL    |       |
| estaciones           | varchar(2)   | YES  |     | NULL    |       |
| vestuario_recomendado | varchar(200) | YES  |     | NULL    |       |
+----------------------+--------------+------+-----+---------+-------+
4 rows in set (0.001 sec)

root@localhost [Ana]>
```

```
root@localhost [Ana]> SELECT * FROM CondicionesLugar;
+------------+-------+------------+----------------------------+
| lugar      | clima | estaciones | vestuario_recomendado      |
+------------+-------+------------+----------------------------+
| Montreal   | Frio  | si         | Ropa termica               |
| Nueva York | frio  | si         | Chaqueta abrigada y botas  |
| Paris      | frio  | si         | Chaqueta abrigada          |
+------------+-------+------------+----------------------------+
3 rows in set (0.001 sec)
```

Tercera tabla.

```
root@localhost [Ana]> CREATE TABLE CaracteristicasViaje (lugar varchar(100) not
null, tipo_viaje varchar(200) not null, compania varchar(50) not null, fecha_est
imada date, PRIMARY KEY(lugar), FOREIGN KEY (lugar) REFERENCES ViajesAna (lugar)
);
Query OK, 0 rows affected (0.004 sec)

root@localhost [Ana]> describe CaracteristicasViaje
    -> ;
+----------------+--------------+------+-----+---------+-------+
| Field          | Type         | Null | Key | Default | Extra |
+----------------+--------------+------+-----+---------+-------+
| lugar          | varchar(100) | NO   | PRI | NULL    |       |
| tipo_viaje     | varchar(200) | NO   |     | NULL    |       |
| compania       | varchar(50)  | NO   |     | NULL    |       |
| fecha_estimada | date         | YES  |     | NULL    |       |
+----------------+--------------+------+-----+---------+-------+
4 rows in set (0.000 sec)
```

```
root@localhost [Ana]> SELECT * FROM CaracteristicasViaje;
+------------+------------+---------+----------------+
| lugar      | tipo_viaje | compania | fecha_estimada |
+------------+------------+---------+----------------+
| Montreal   | Vacaciones | David    | 2022-06-10     |
| Nueva York | Trabajo    | Sola     | 2021-12-01     |
| Paris      | Conocer    | Amigos   | 2021-10-15     |
+------------+------------+---------+----------------+
3 rows in set (0.001 sec)
```

- Creación de tablas e inserción de datos sobre la base de datos Sebastian.

Primera tabla.

```
root@localhost [Sebastian]> describe ViajesSebastian
    -> ;
+-------------+--------------+------+-----+---------+-------+
| Field       | Type         | Null | Key | Default | Extra |
+-------------+--------------+------+-----+---------+-------+
| lugar       | varchar(100) | NO   | PRI | NULL    |       |
| pais        | varchar(100) | YES  |     | NULL    |       |
| descripcion | varchar(200) | YES  |     | NULL    |       |
+-------------+--------------+------+-----+---------+-------+
3 rows in set (0.001 sec)
```

```
root@localhost [Sebastian]> SELECT * FROM ViajesSebastian;
+--------+--------+-------------------------+
| lugar  | pais   | descripcion             |
+--------+--------+-------------------------+
| Cancun | Mexico | Clima fantastico        |
| Hawai  | USA    | Lugar cool para viajar  |
| Madrid | Spain  | Lugar turistico         |
+--------+--------+-------------------------+
3 rows in set (0.000 sec)
```

Segunda tabla

```
root@localhost [Sebastian]> describe CondicionesLugar;
+----------------------+-------------+------+-----+---------+-------+
| Field                | Type        | Null | Key | Default | Extra |
+----------------------+-------------+------+-----+---------+-------+
| lugar                | varchar(100) | NO   | PRI | NULL    |       |
| clima                | varchar(30)  | YES  |     | NULL    |       |
| estaciones           | varchar(2)   | YES  |     | NULL    |       |
| vestuario_recomendado | varchar(200) | YES  |     | NULL    |       |
+----------------------+-------------+------+-----+---------+-------+
4 rows in set (0.001 sec)
```

```
root@localhost [Sebastian]> SELECT * FROM CondicionesLugar;
+--------+----------+------------+--------------------------+
| lugar  | clima    | estaciones | vestuario_recomendado    |
+--------+----------+------------+--------------------------+
| Cancun | Caliente | no         | Ligero                   |
| Hawai  | Tropical | no         | Ropa ligera, no abrigada |
| Madrid | Frio     | si         | Ropa abrigada            |
+--------+----------+------------+--------------------------+
3 rows in set (0.000 sec)
```

Tercera tabla

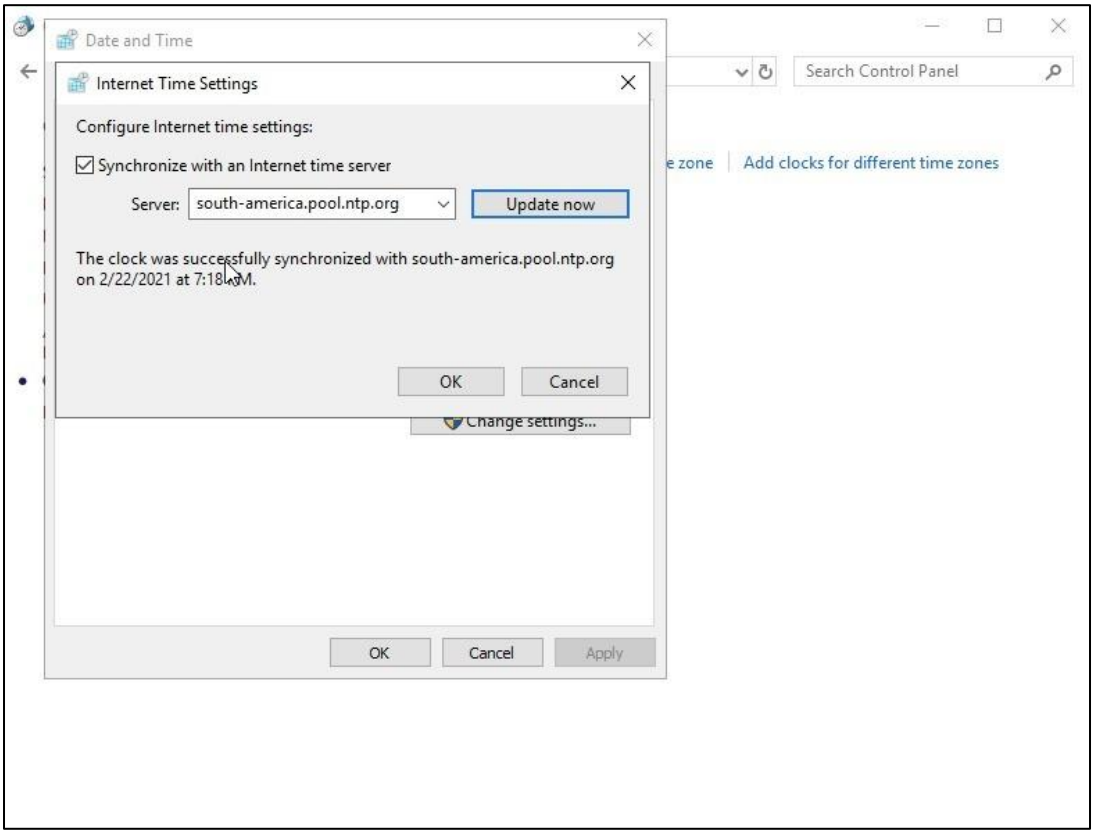
```
root@localhost [Sebastian]> describe CaracteristicasViaje;
+----------------+-------------+------+-----+---------+-------+
| Field          | Type        | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| lugar          | varchar(100) | YES  |     | NULL    |       |
| tipo_viaje     | varchar(200) | YES  |     | NULL    |       |
| fecha_estimada | date        | YES  |     | NULL    |       |
+----------------+-------------+------+-----+---------+-------+
3 rows in set (0.001 sec)
```

```
root@localhost [Sebastian]> SELECT * FROM CaracteristicasViaje;
+--------+-----------+----------------+
| lugar  | tipo_viaje | fecha_estimada |
+--------+-----------+----------------+
| Hawai  | Vacaciones | 2021-06-10     |
| Cancun | Vacaciones | 2021-12-23     |
| Madrid | Trabajo    | 2022-02-15     |
+--------+-----------+----------------+
3 rows in set (0.000 sec)
```

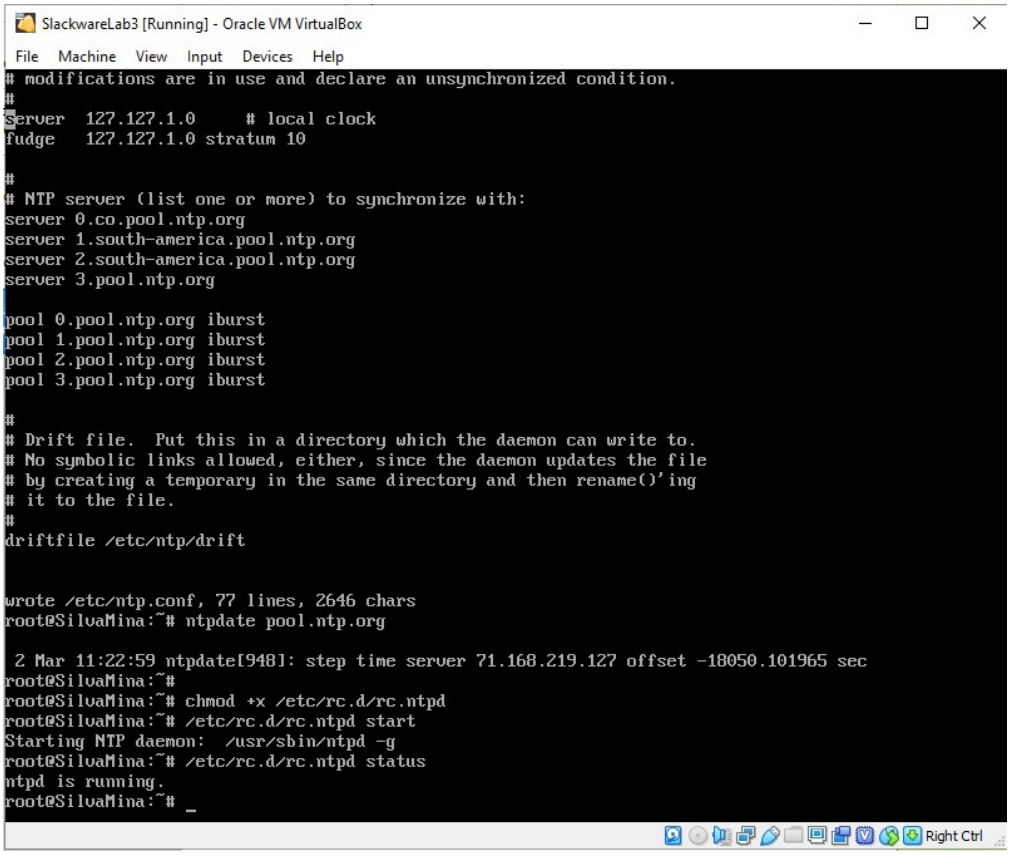
3. NTP SERVER

El NTP se describe en este documento como un protocolo para sincronizar varios relojes de red usando un conjunto de clientes y servidores repartidos. Como predecesores, se hace mención al mensaje ICMP Timestamp y al Time Protocol, ya que sus funciones fueron incluidas en el Network Time Protocol. El NTP se basa en el protocolo de datagramas de usuario (User Datagram Protocol o UDP), que permite enviar datagramas sin que se haya establecido previamente una conexión. Es decir, utiliza UDP como capa de transporte usando el puerto 123.

El NTP proporciona los mecanismos de protocolo básicos necesarios para sincronizar los relojes de los diferentes sistemas con una precisión del orden de nanosegundos. Además, contiene indicaciones para especificar la precisión y las posibles fuentes de error del reloj del sistema local, así como las propiedades del reloj de referencia. No obstante, este protocolo se limita a especificar la arquitectura de la representación de datos y los formatos de mensaje, sin que por sí mismo lleve a cabo la sincronización y el algoritmo de filtrado.



WINDOWS SERVER 2016



SLACKWARE

CONCLUSIONES:

- La instalación de MARIA DB fue mucho más sencilla ya que la máquina contaba con los paquetes necesario y solo se tuvo que seguir la guía.
- La instalación de postgres en slackware fue más complicada, se tuvo que hacer instalación de nueva máquina con los paquetes necesarios para realizarla.

REFERENCIAS:

(229) Instalando o Servidor PostgreSQL no Linux Slackware - YouTube
Easily Install MariaDB 10.5 on FreeBSD 12.1 - kifarunix.com
Instalación de PostgreSQL en Slackware Linux | LogicalBricks Blog
pool.ntp.org: NTP Servers in Global, pool.ntp.org
NIST Internet Time Service
Cómo configurar servidor NTP Windows Server 2016 - Solvetic
¿Qué es PostgreSQL? - Para qué sirve, Características e Instalación (infranetworking.com)
Que es MariaDB y mejoras sobre MySQL | Nerion
NTP: los secretos del Network Time Protocol - IONOS
29.12. NTP (freebsd.org)
Servidores de Tiempo en Slackware (trulinux.com)
FreeBSD Install OpenNTPD NTP Server / Client To Synchronize The Local Clock - nixCraft (cyberciti.biz)