

UNIVERSIDAD PRIVADA-DE-TACNA



INGENIERIA DE SISTEMAS

TITULO:

INFORME DE LABORATORIO N° 04

CURSO:

BASE DE DATOS II

DOCENTE(ING):

Patrick Cuadros Quiroga

Alumno:

Laura Atencio, Nilson Felix

(2015053846)

Índice

1. INFORMACIÓN GENERAL	1
1.1. Objetivos:.....	1
1.2. Requerimientos.....	1
2. MARCO TEORICO	2
3. PROCEDIMIENTO	3
3.1. Instalacion de Docker.....	3
3.2. Iniciando en Docker	5
3.3. Creando un contenedor.....	6
3.4. Adicionando una Persistencia.....	11
3.5. Creando un contenedor.....	13
3.6. Actividades Encargadas	18
4. CUESTIONARIO	21
4.1. ¿Con qué comando(s) exportaría la imagen de Docker de Microsoft SQL Server a otra PC o servidor?	21
4.2. ¿Con qué comando(s) podría generar dos volúmenes para un contenedor para distribuir en un volumen el Archivo de Datos (mdf) y en otro el Archivo Log (ldf)? . .	21
4.3. Genere un nuevo contenedor y cree la base de datos con las siguientes características	21
5. CONCLUSIONES	22
6. BIBLIOGRAFIA	23
7. WEBGRAFIA	24

1. INFORMACIÓN GENERAL

1.1. Objetivos:

- iniciar la Instalacion de una Instancia de Microsoft SQL

1.2. Requerimientos

Conocimientos

- Conocimientos básicos de administración de base de datos.
- Conocimientos básicos de SQL.

Hardware

- CPU SLAT-capable feature.
- Al menos 4GB de RAM.
- Virtualization activada en el BIOS..

Software

- Windows 10 64bit: Pro, Enterprise o Education, con al menos 4GB de RAM.
- Docker Desktop
- Microsoft SQL Server 2017 o superior

2. MARCO TEORICO

Docker es una tecnología que promete revolucionar la informática profesional. Para los que no lo sepan, se trata de una tecnología de **contenedores**, que básicamente consiste en la ejecución de sistemas operativos dentro de otros, obteniendo los sistemas «invitados» su propio sistema de ficheros, su propio espacio de usuarios, sus propios procesos y sus propias interfaces de red, pero compartiendo algunos elementos de la máquina anfitriona como el kernel.

Para separar los contenedores entre sí y de la máquina anfitriona, Docker utiliza las características de aislamiento del kernel Linux. Todo este enfoque permite a los contenedores ser mucho más ligeros que las máquinas virtuales, tanto en espacio en disco como de consumo de recursos. Además su naturaleza les otorga una gran portabilidad y seguridad. Su principal función es la de poder **empaquetar aplicaciones con todas las partes necesarias, incluyendo bibliotecas y dependencias**, pudiendo luego ser reutilizado otro tipo de aplicaciones, pudiendo recordar un poco el concepto de los paquetes Snap de Ubuntu, aunque aquí no llegaremos hasta ese punto.

Los contenedores no son algo nuevo, de hecho compañías como Google y Amazon llevan años utilizándolos, pero eran muy difíciles de manejar y requerían de profundos conocimientos para utilizarlos. Docker rompió esas barreras ofreciendo una forma más o menos sencilla de instalar, configurar y utilizar contenedores.

– Funcionamiento de Docker

- El propósito de los contenedores es la independencia, es decir, la capacidad de ejecutar varios procesos y aplicaciones por separado para hacer un mejor uso de su infraestructura y, al mismo tiempo, conservar la seguridad que tendrían con sistemas separados.

- Las herramientas del contenedor ofrecen un modelo de implementación basado en imágenes.

- Permite compartir una aplicación, o un conjunto de servicios, con todas sus dependencias en varios entornos.

– Ventajas de los contenedores Docker :

- Facilita el testing, facilita la tarea, puesto que si tenemos instalado Docker en nuestro ordenador y nos pasan un contenedor con una App a testear. Da igual cual sea el software que tengamos, docker nos permitirá abrir la app y poder probarla.
- Ahorra tiempo, al no obligarnos a instalar diferentes softwares para poder ejecutar una App.
- Es muy sencillo crear y eliminar contenedores.
- Son muy ligeros, lo que nos permite manejar diferentes contenedores dentro de una misma máquina.
- Al necesitar menos espacio y poderlos incluir en una misma máquina, implica que necesitemos menos ordenadores. Menos costes.
- Es open source.
- Nos proporcionan autonomía, al partir de que en cada contenedor tenemos todo lo necesario para ejecutar una aplicación.
- Portabilidad. Al almacenar los contenedores en discos duros, estos se pueden transportar de un lugar a otro sin problemas.
- Imágenes docker. Podríamos definir estas imágenes como sistemas operativos con aplicaciones instaladas. A este SO, podremos incluir nuestras imágenes para su posterior visualización en un equipo.
- Repositorios Docker. “Banco de imágenes docker” creadas por usuarios a las cuales podemos tener acceso.
- Con Docker, tenemos capacidad de ejecutar prácticamente todas las aplicaciones.
- Nos facilita el compartir nuestras aplicaciones a través de los contenedores.
- Se acelera el proceso de mantenimiento y desarrollo gracias a las facilidades para generar copias.
- Las aplicaciones se ejecutan sin variaciones. Sin importar el equipo ni el ambiente.
- Facilita las visualizaciones al cliente gracias a que no tiene que instalar nada más que docker en su ordenador.
- Es un entorno seguro y no ofrece variaciones.

Instancias

Las instancias de Docker son más ligeras. Para desplegar una app como imagen de una máquina virtual, lo más probable es que tengas que incluir un sistema operativo entero en la imagen. Con un contenedor, solo la app y unas cuantas capas de base tienen que ir dentro del contenedor. Esto se traduce en un proceso de montaje más sencillo, y, encima, la capacidad de poder alojar muchos más contenedores en un servidor físico único. Además, arrancan mucho más rápido (en centésimas de segundo) y te puedes permitir el lujo de lanzarlos y cerrarlos automáticamente según las necesidades.

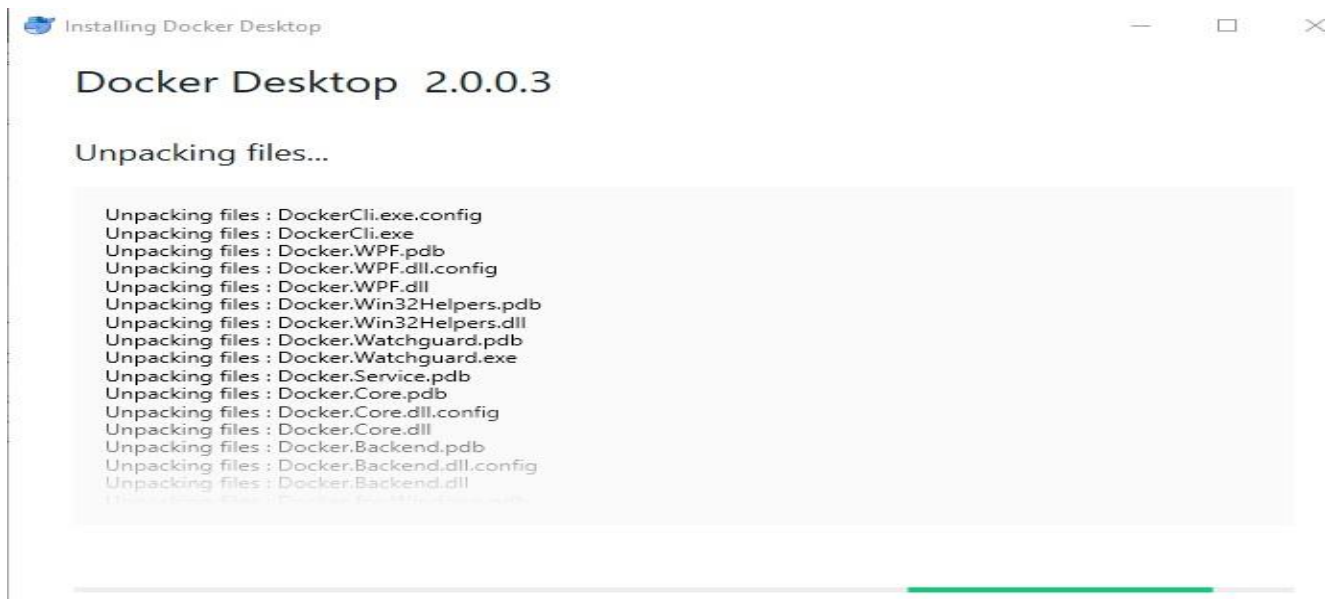
3. PROCEDIMIENTO

3.1. Instalacion de Docker

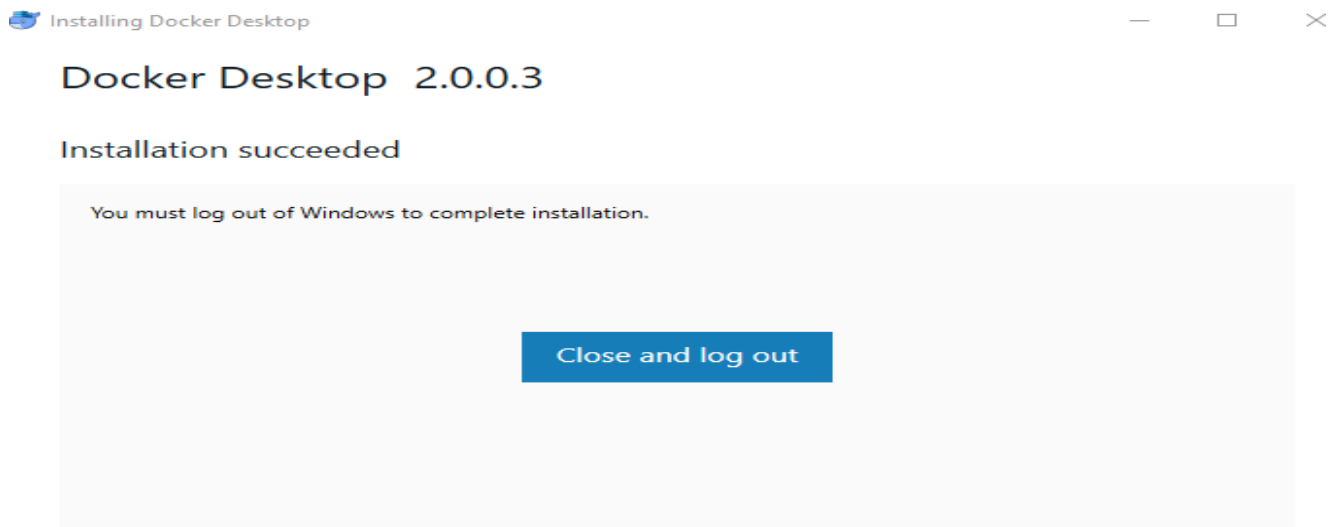
- Instalar Docker desde la siguiente direccion : <https://docs.docker.com/docker-for-windows/install/>



- Seguir el proceso de Instalacion :



- Reiniciar la PC :



- Comprobar que docker ha sido instalado correctamente:

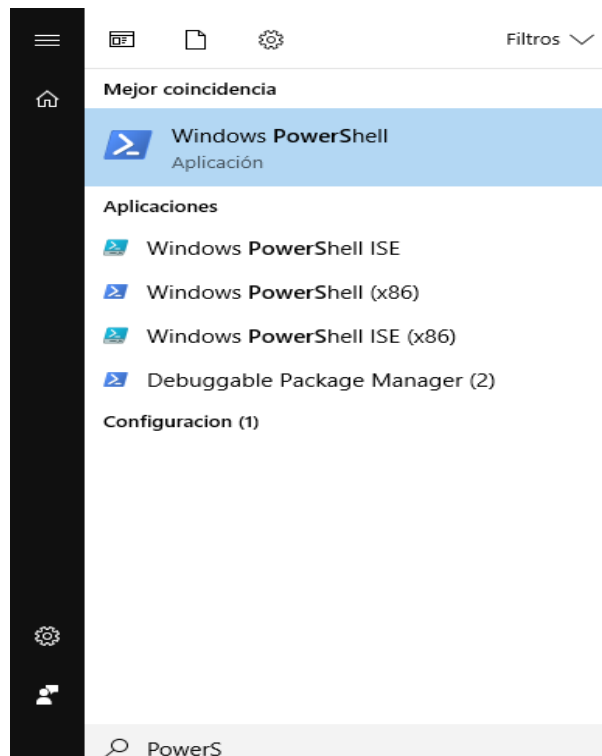


3.2. Iniciando en Docker

- Logearse con su cuenta y contraseña Respectiva:



- Iniciar la consola PowerShell de Windows:



3.3. Creando un contenedor Microsoft SQL para Linux

- En la ventana de PowerShell, escribir el siguiente comando: "docker search mssql"

```
PS C:\> docker search mssql
```

NAME	AUTOMATED	DESCRIPTION	STARS	OFFICIAL
microsoft/mssql-server-linux		Deprecated SQL Server on Linux Container Rep...	1123	
microsoft/mssql-server-windows-developer		Official Microsoft SQL Server Developer Edit...	316	
microsoft/mssql-server-windows-express		Official Microsoft SQL Server Express Editio...	300	
microsoft/mssql-tools		Official images for Microsoft SQL Server Com...	51	
rsmoorthy/mssql		MSSQL Database (version SQL2000)	11	
[OK]				
datagrip/mssql-server-linux		SQL Server and SQL Server tools on Linux(201...	9	
[OK]				
tsgkadot/mssql-tools		SQL Server tools on Linux (sqlcmd)	3	
[OK]				
microsoft/mssql-monitoring-influxdb		Sample Image for Influxdb, This image is des...	3	
jboesl/mssql-server-linux		mssql-server-linux with mssql-tools installe...	2	
[OK]				
mcroe/mssqldocker		Builds on microsoft/mssql-server-linux and a...	2	
[OK]				
awaragi/prometheus-mssql-exporter		prometheus-mssql-exporter	1	
[OK]				
mileiq/ubuntu16-python3-mssql-kafka		Base image built on top of mileiq/ubuntu16-p...	1	
microsoft/mssql-monitoring-collectd		This Sample image is designed to work with t...	1	

- Como primer comando usaremos: "docker version" para ver la version de docker que acabamos de instalar:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

PS C:\Users\Usuario> docker version
Client: Docker Engine - Community
 Version:      18.09.2
 API version:  1.39
 Go version:   go1.10.8
 Git commit:   6247962
 Built:        Sun Feb 10 04:12:31 2019
 OS/Arch:     windows/amd64
 Experimental: false

Server: Docker Engine - Community
 Engine:
  Version:      18.09.2
  API version:  1.39 (minimum version 1.12)
  Go version:   go1.10.6
  Git commit:   6247962
  Built:        Sun Feb 10 04:13:06 2019
  OS/Arch:     linux/amd64
  Experimental: false
PS C:\Users\Usuario>
```

- Ahora crearemos un contenedor con Microsoft SQL server para Linux, para esto usaremos primero el comando "docker search mssql":

```

PS C:\Users\Usuario> docker search mssql
NAME                                DESCRIPTION                                STARS                                OFFICIAL                                AUTOMATED
microsoft/mssql-server-linux        Deprecated SQL Server on Linux Container Rep... 1122
microsoft/mssql-server-windows-developer Official Microsoft SQL Server Developer Edit... 315
microsoft/mssql-server-windows-express Official Microsoft SQL Server Express Editio... 300
microsoft/mssql-tools                Official images for Microsoft SQL Server Com... 51
rsmoorthy/mssql                     MSSQL Database (version SQL2000)              11                                [OK]
datagrip/mssql-server-linux          SQL Server and SQL Server tools on Linux(201... 9                                [OK]
gantrior/mssql-server-2014-express-windows-with-iis mssql 2014 + IIS                             4
microsoft/mssql-monitoring-influxdb Sample Image for Influxdb, This image is des... 3
tsgkadot/mssql-tools                SQL Server tools on Linux (sqlcmd)             3                                [OK]
jboesl/mssql-server-linux           mssql-server-linux with mssql-tools installe... 2                                [OK]
mcmoe/mssqldocker                   Builds on microsoft/mssql-server-linux and a... 2                                [OK]
microsoft/mssql-monitoring-collectd This Sample image is designed to work with t... 1
awaragi/prometheus-mssql-exporter    prometheus-mssql-exporter                     1                                [OK]
mondora/sandman2-mssql              Docker image for running sandman2 to get a R... 0                                [OK]
ansibleplaybookbundle/mssql-apb     MS SQL Server on Linux (APB)                  0                                [OK]
r2dbc/r2dbc-mssql                   0
bitwarden/mssql                     The Bitwarden database.                       0
tchughesiv/mssql-server-linux        CentOS build                                  0                                [OK]
ansibleplaybookbundle/mssql-remote-apb An APB that deploys Microsoft SQL Server      0                                [OK]
softwareplant/mssql                 SQL Server test database                     0                                [OK]
mileiq/ubuntu16-python3-mssql-kafka Base image built on top of mileiq/ubuntu16-p... 0
ncia/anet-mssql-linux               Container image for running a mssql database... 0
langdon/fedora-mssqlserver           Microsoft SQL Server running on Fedora. You ... 0                                [OK]
liaisonintl/mssql-server-linux       mssql-server-linux                           0                                [OK]
astronomerio/mssql-source           MSSQL source.                                 0                                [OK]
PS C:\Users\Usuario>

```

- Luego descargaremos la imagen del contenedor de Microsoft SQL en un servidor Linux con el siguiente comando "docker pull microsoft/mssql-server-linux":

```
PS C:\> docker pull microsoft/mssql-server-linux
Using default tag: latest
latest: Pulling from microsoft/mssql-server-linux
59ab41dd721a: Downloading [=====>] 23.13MB/42.22MB
57da90bec92c: Download complete
06fe57530625: Download complete
5a6315cba1ff: Download complete
739f58768b3f: Download complete
0b751601bca3: Download complete
bcf04a22644a: Waiting
6b5009e4f470: Waiting
a9dca2f6722a: Waiting
```

- esperar un determinado tiempo a que descargue todo:

```
PS C:\Users\Usuario> docker pull microsoft/mssql-server-linux
Using default tag: latest
latest: Pulling from microsoft/mssql-server-linux
59ab41dd721a: Pull complete
57da90bec92c: Pull complete
06fe57530625: Pull complete
5a6315cba1ff: Pull complete
739f58768b3f: Pull complete
0b751601bca3: Pull complete
bcf04a22644a: Pull complete
6b5009e4f470: Pull complete
a9dca2f6722a: Pull complete
Digest: sha256:9b700672670bb3db4b212e8aef841ca79eb2fce7d5975a5ce35b7129a9b90ec0
Status: Downloaded newer image for microsoft/mssql-server-linux:latest
PS C:\Users\Usuario>
```

- Para ver la imagen que acabamos de descargar, usaremos el siguiente comando: "docker images":

```
PS C:\> docker images
```

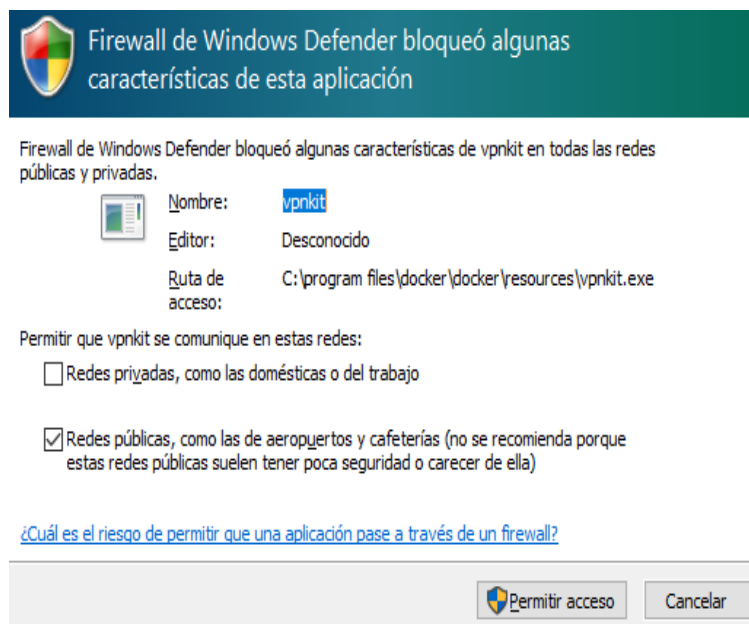
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
microsoft/mssql-server-linux	latest	314918ddaedf	5 months ago	1.35GB

```
PS C:\>
```

- Ahora crearemos credenciales los cuales usaremos mas adelante para autenticar nuestra entrada a SQL server, usaremos el siguiente comando:

```
PS C:\Users\Usuario> docker run -d -p 16111:1433 -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=Tacna.2019' --name SQLLNK01 microsoft/mssql-server-linux
64108766dc344a30ae93e0f7785737d718a8bb334f603ac590bd214df23772f7
PS C:\Users\Usuario>
```

- Accedemos a dar los permisos para el firewall de Windows



- Verificamos la correcta ejecucion del contenedor con el comando "docker ps":

```

PS C:\Users\Usuario> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                    NAMES
64108766dc34   microsoft/mssql-server-linux        "/opt/mssql/bin/sqls..." 7 minutes ago  Up 7 minutes  0.0.0.0:16111->1433/tcp  SQLLNK01
PS C:\Users\Usuario>

```

- Accedemos a Sql server con los siguientes credenciales:

SQL Server

Server type: Database Engine

Server name: (local),16111

Authentication: SQL Server Authentication

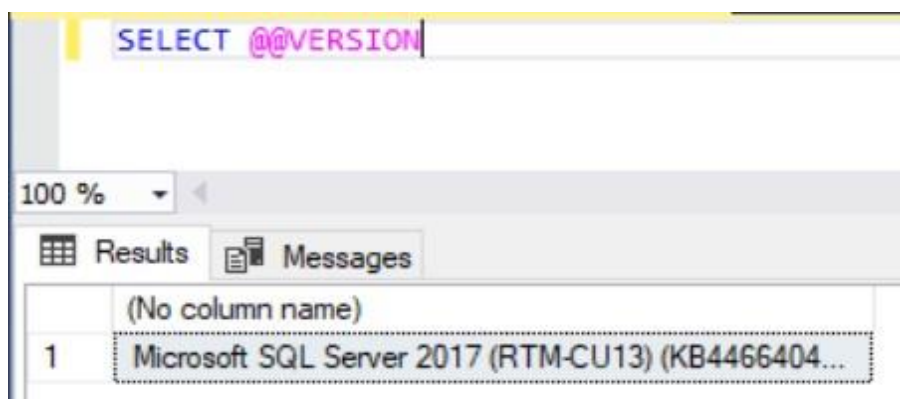
Login: sa

Password: *****

☐ Remember password

Connect Cancel Help Options >>

- En sql iniciaremos un nuevo query para hacer una consulta sobre la version:



- Ahora cerraremos Sql server y procederemos a eliminar el contenedor creado con el siguiente comando: "docker rm -f SQLLNx01" despues comprobaremos que este ha sido eliminado:

```
PS C:\> docker rm -f SQLLNx01
SQLLNx01
PS C:\>
```

3.4. Adicionando una Persistencia

- Crearemos un nuevo contenedor, verificaremos que este ha sido creado correctamente y luego iniciaremos sesion con los respectivos credenciales:

```
PS C:\Users\Usuario> docker run -d -p 16111:1433 -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=Tacna.2019' -v F:\DATALNX:\var/opt/mssql --name SQLLNK02 microsoft/mssql-server-linux
51ffc1db6e63d097766ce8c7deac1185050cbd4ab163707f7dffc607c862d6e
PS C:\Users\Usuario>
```

```
PS C:\Users\Usuario> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                    NAMES
51ffc1db6e63   microsoft/mssql-server-linux        "/opt/mssql/bin/sqls..." About a minute ago Up About a minute  0.0.0.0:16111->1433/tcp  SQLLNK02
PS C:\Users\Usuario>
```

Connect to Server

SQL Server

Server type: Database Engine

Server name: (local), 16111

Authentication: SQL Server Authentication

Login: sa

Password: *****

☐ Remember password











Connect Cancel Help Options >>

- Ahora crearemos una base de datos con el siguiente Script:

```
Query1.sql - (lo...111.master (sa (52))*) X
CREATE DATABASE BIBLIOTECA ON
PRIMARY (
    NAME = N'BIBLIOTECA',
    FILENAME = N'/var/opt/mssql/data/BIBLIOTECA.mdf',
    SIZE = 50MB ,
    FILEGROWTH = 10240KB
) LOG ON (
    NAME = N'BIBLIOTECA_log',
    FILENAME = N'/var/opt/mssql/data/BIBLIOTECA_log.ldf',
    SIZE = 10MB ,
    FILEGROWTH = 5MB
)
GO
```

- Verificaremos que la carpeta DATALNX contenga esta base de datos:

o local (C:) > DATALNX > data

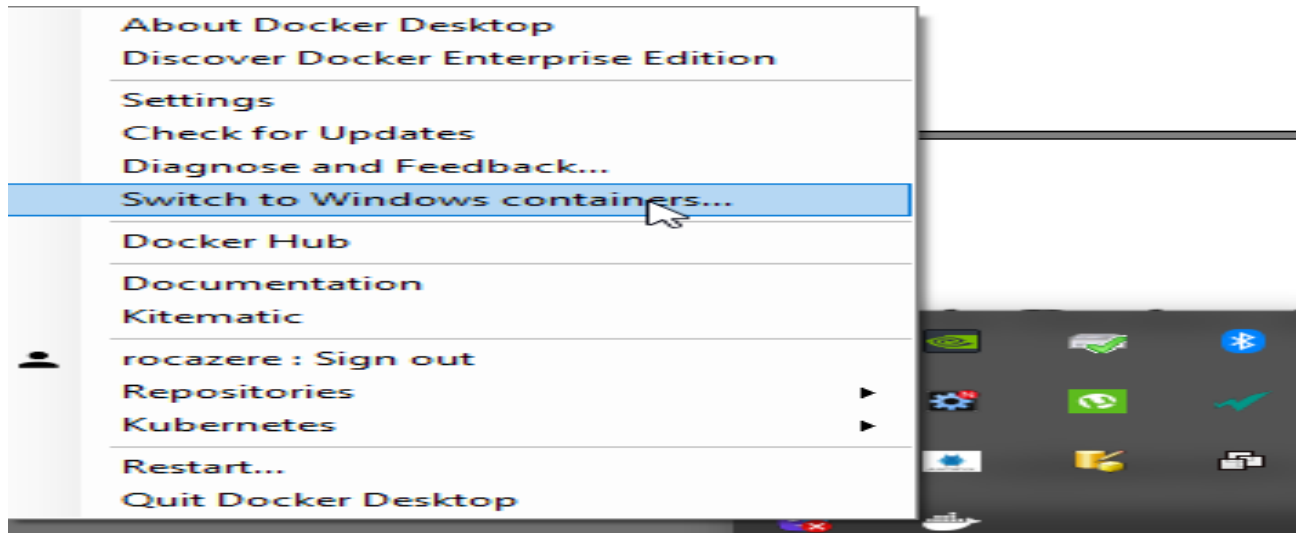
Nombre	Fecha de modifica...	Ti
 BIBLIOTECA	22/05/2019 19:36	SC
 BIBLIOTECA_log	22/05/2019 19:36	SC
 master	22/05/2019 19:35	SC
 mastlog	22/05/2019 19:37	SC
 model	22/05/2019 19:37	SC
 modellog	22/05/2019 19:35	SC
 msdbdata	22/05/2019 19:31	SC
 msdblog	22/05/2019 19:31	SC
 tempdb	22/05/2019 19:31	SC
 templog	22/05/2019 19:31	SC

Tipo: SQL Server Database Primary Data File
 Tamaño: 4.00 MB
 Fecha de modificación: 22/05/2019 19:35

- Por ultimo eliminaremos este contenedor.

3.5. Creando un contenedor con Microsoft SQL para Windows

- En la parte inferior derecha encontraremos el icono de Docker el cual al hacerle click derecho, abrira un menu desplegable en el que seleccionaremos Switch to windows containers... y esperaremos a que docker se reinicie:



- Ahora en la ventana de PowerShell usaremos los siguientes comandos::

```
Windows PowerShell
PS C:\Users\Usuario> docker search mssql
```

NAME	DESCRIPTION	STARS	OFFICIAL	AUTOMATED
microsoft/mssql-server-linux	Deprecated SQL Server on Linux Container Rep...	1122		
microsoft/mssql-server-windows-developer	Official Microsoft SQL Server Developer Edit...	315		
microsoft/mssql-server-windows-express	Official Microsoft SQL Server Express Editio...	300		
microsoft/mssql-tools	Official images for Microsoft SQL Server Com...	51		
rsmoorthy/mssql	MSSQL Database (version SQL2000)	11		[OK]
datagrip/mssql-server-linux	SQL Server and SQL Server tools on Linux(201...	9		[OK]
gantrior/mssql-server-2014-express-windows-with-iis	mssql 2014 + IIS	4		
microsoft/mssql-monitoring-influxdb	Sample Image for Influxdb, This image is des...	3		
tsgkadot/mssql-tools	SQL Server tools on Linux (sqlcmd)	3		[OK]
jboesl/mssql-server-linux	mssql-server-linux with mssql-tools installe...	2		[OK]
mcroe/mssqldocker	Builds on microsoft/mssql-server-linux and a...	2		[OK]
microsoft/mssql-monitoring-collectd	This Sample image is designed to work with t...	1		
awaragi/prometheus-mssql-exporter	prometheus-mssql-exporter	1		[OK]
mondora/sandman2-mssql	Docker image for running sandman2 to get a R...	0		[OK]
ansibleplaybookbundle/mssql-apb	MS SQL Server on Linux (APB)	0		[OK]
r2dbc/r2dbc-mssql		0		
bitwarden/mssql	The Bitwarden database.	0		
tchughesiv/mssql-server-linux	CentOS build	0		[OK]
ansibleplaybookbundle/mssql-remote-apb	An APB that deploys Microsoft SQL Server	0		[OK]
softwareplant/mssql	SQL Server test database	0		[OK]
mileiq/ubuntu16-python3-mssql-kafka	Base image built on top of mileiq/ubuntu16-p...	0		
ncia/anet-mssql-linux	Container image for running a mssql database...	0		
langdon/fedora-mssqlserver	Microsoft SQL Server running on Fedora. You ...	0		[OK]
liaisonintl/mssql-server-linux	mssql-server-linux	0		[OK]
astronomerio/mssql-source	MSSQL source.	0		[OK]

```
PS C:\Users\Usuario>
```

- Instalaremos el contenedor de Microsoft sql para un servidor Windows:

Windows PowerShell

```
PS C:\Users\Usuario> docker pull microsoft/mssql-server-windows-developer
Using default tag: latest
latest: Pulling from microsoft/mssql-server-windows-developer
3889bb8d808b: Pulling fs layer
449343c9d7e2: Pulling fs layer
08883151461d: Download complete
bafeb45a72fc: Download complete
f5c5aa235c5b: Waiting
158fead2ffa0: Waiting
746db9597cec: Waiting
9e96edbd8781: Waiting
c6dabab6234f: Waiting
975d0dccd859: Waiting
5b747cfb01b7: Waiting
c77992bbfd0f: Waiting
```

- Comprobaremos la correcta instalacion del contenedor con el comando "docker images":

Windows PowerShell

```
PS C:\Users\Usuario> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
microsoft/mssql-server-windows-developer	latest	19873f41b375	16 months ago	15.1GB

```
PS C:\Users\Usuario>
```

- Crearemos nuevas credenciales para este nuevo contenedor Sql para servidores windows:

```
Windows PowerShell
PS C:\Users\Usuario> docker run -d -p 16111:1433 -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=Tacna.2019' -v D:\DATAWIN:C:\DATA --name SQLWIN01 microsoft/mssql-server-windows-developer
b1f8e0e7a08935f36caf90514383f92e7b8672aec266128f71fd7c07b5c91990
```

- Iniciaremos sesion en Sql con las credenciales que hemos creado:

Connect to Server

SQL Server

Server type: Database Engine

Server name: (local),16111

Authentication: SQL Server Authentication

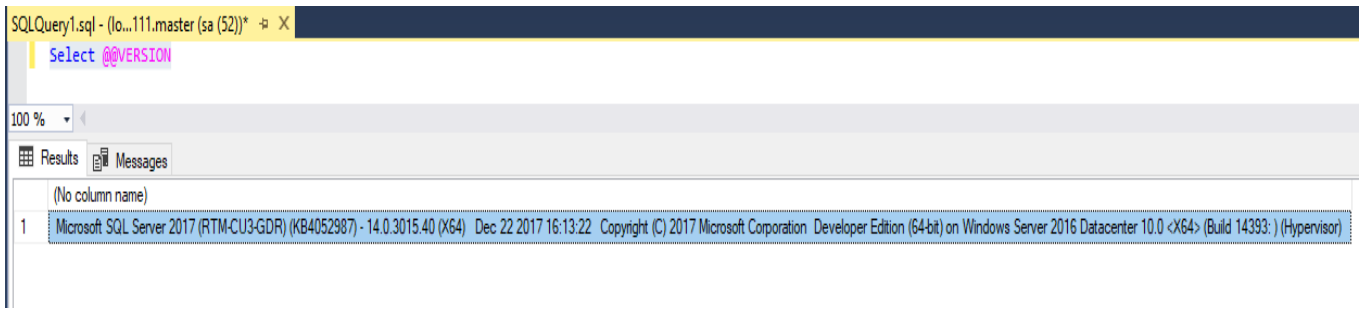
Login: sa

Password: *****

☐ Remember password

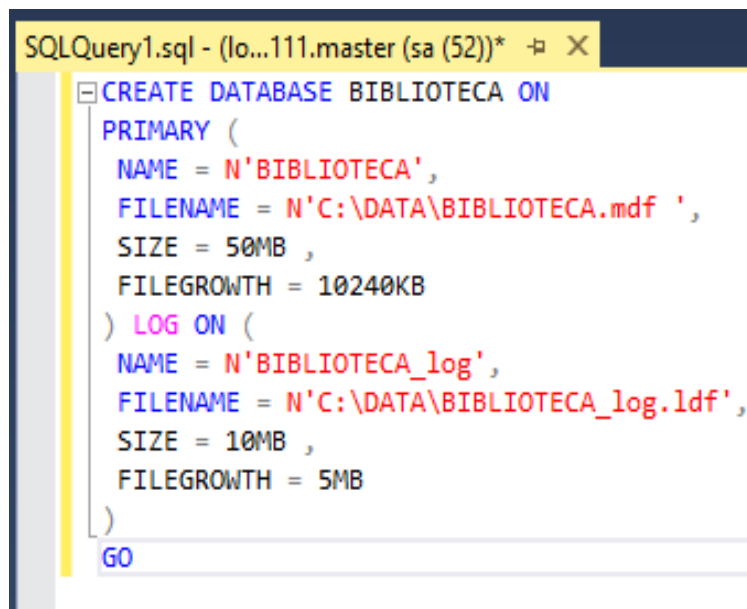
Connect Cancel Help Options >>

- revisamos la version:



The screenshot shows a SQL query window titled "SQLQuery1.sql - (lo...111.master (sa (52)))". The query is "Select @@VERSION". The results pane shows a single row with the text: "Microsoft SQL Server 2017 (RTM-CU3-GDR) (KB4052987) - 14.0.3015.40 (X64) Dec 22 2017 16:13:22 Copyright (C) 2017 Microsoft Corporation Developer Edition (64-bit) on Windows Server 2016 Datacenter 10.0 <X64> (Build 14393;) (Hypervisor)".

- Mediante el siguiente scrip generaremos una base de datos de prueba:



The screenshot shows a SQL query window titled "SQLQuery1.sql - (lo...111.master (sa (52)))". The script is as follows:

```
CREATE DATABASE BIBLIOTECA ON
PRIMARY (
    NAME = N'BIBLIOTECA',
    FILENAME = N'C:\DATA\BIBLIOTECA.mdf ',
    SIZE = 50MB ,
    FILEGROWTH = 10240KB
) LOG ON (
    NAME = N'BIBLIOTECA_log',
    FILENAME = N'C:\DATA\BIBLIOTECA_log.ldf',
    SIZE = 10MB ,
    FILEGROWTH = 5MB
)
GO
```

- Comprobaremos que la base de datos ha sido creada:

» Nuevo vol (D:) » DATAWIN

Nombre	Fecha de modifica...	Tipo	Tamaño
BIBLIOTECA	19/05/2019 16:29	SQL Server Databa...	51,200 KB
BIBLIOTECA_log	19/05/2019 16:29	SQL Server Databa...	10,240 KB

- Finalmente procederemos con la eliminacion del contenedor y verificaremos que esta ha sido eliminada:

Windows PowerShell

```
PS C:\Users\Usuario> docker rm -f SQLWIN01
SQLWIN01
PS C:\Users\Usuario>
```

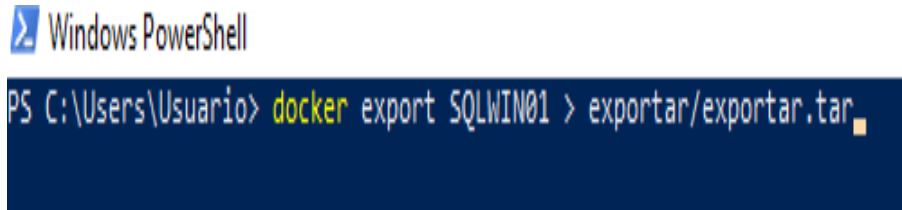
Windows PowerShell

```
PS C:\Users\Usuario> docker ps
CONTAINER ID    IMAGE    COMMAND    CREATED
PS C:\Users\Usuario>
```

3.6. Actividades Encargadas

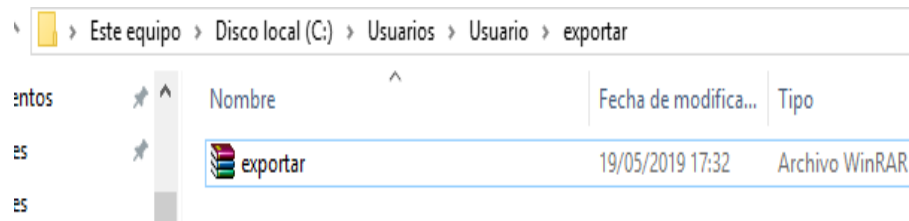
3.6.1. ¿Con qué comando(s) exportaría la imagen de Docker de Microsoft SQL Server a otra PC o servidor?

- uno de los comandos usados para exportar un contenedor seria:



```
Windows PowerShell
PS C:\Users\Usuario> docker export SQLWIN01 > exportar/exportar.tar
```

Podemos observar que hemos guardado un archivo .tar en nuestra carpeta usuarios. Luego esto podrá ser transportado a otra máquina ya sea Windows o Linux.



3.6.2. ¿Con qué comando(s) podría generar dos volúmenes para un contenedor?

- Los volúmenes pueden ser gestionados con el siguiente comando:

```
PS C:\Users\Usuario> docker volume create Datos
datos
PS C:\Users\Usuario> docker volume create Log
log
PS C:\Users\Usuario> _
```

- Con el siguiente comando, podremos ver donde estos han sido creados:

```
PS C:\Users\Usuario> docker volume inspect datos
[
  {
    "CreatedAt": "2019-05-19T17:40:12-05:00",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "C:\\ProgramData\\Docker\\volumes\\datos\\_data",
    "Name": "datos",
    "Options": {},
    "Scope": "local"
  }
]
PS C:\Users\Usuario> docker volume inspect log
[
  {
    "CreatedAt": "2019-05-19T17:49:35-05:00",
    "Driver": "local",
    "Labels": {},
    "Mountpoint": "C:\\ProgramData\\Docker\\volumes\\log\\_data",
    "Name": "log",
    "Options": {},
    "Scope": "local"
  }
]
PS C:\Users\Usuario> _
```

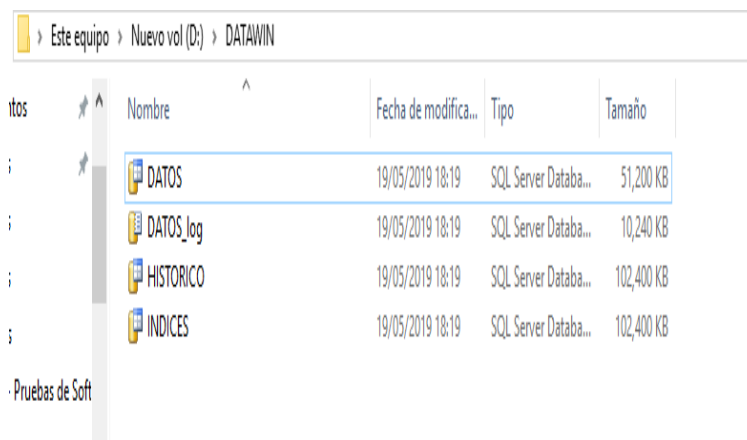
Ahora podremos usar estos volúmenes creados para crear nuestros archivos .mdfy .log en sus respectivos directorios.

3.6.3. Genere un nuevo contenedor con las siguientes características:

– El Script es:

```
SQLQuery1.sql - (lo...111.master (sa (52))) *  X
CREATE DATABASE FINANCIERA ON
PRIMARY (
    NAME = N'DATOS',
    FILENAME = N'C:\DATA\DATOS.mdf ',
    SIZE = 50MB ,
    FILEGROWTH = 10240KB
),
(
    NAME = N'INDICES',
    FILENAME = N'C:\DATA\INDICES.ndf ',
    SIZE = 100MB ,
    FILEGROWTH = 1000MB
),
(
    NAME = N'HISTORICO',
    FILENAME = N'C:\DATA\HISTORICO.ndf ',
    SIZE = 100MB ,
    FILEGROWTH = 51200KB
)
LOG ON (
    NAME = N'DATOS_log',
    FILENAME = N'C:\DATA\DATOS_log.ldf',
    SIZE = 10MB ,
    FILEGROWTH = 10240KB
)
GO
```

– Verificamos que haya sido creado correctamente:



Este equipo > Nuevo vol (D:) > DATAWIN				
itos	Nombre	Fecha de modifica...	Tipo	Tamaño
:	DATOS	19/05/2019 18:19	SQL Server Databa...	51,200 KB
:	DATOS_log	19/05/2019 18:19	SQL Server Databa...	10,240 KB
:	HISTORICO	19/05/2019 18:19	SQL Server Databa...	102,400 KB
:	INDICES	19/05/2019 18:19	SQL Server Databa...	102,400 KB
Pruebas de Soft				

4. ANALISIS E INTERPRETACION DE RESULTADOS

4.1. Parte 1: Actividades Encargadas

- ¿Con qué comando(s) exportaría la imagen de Docker de Microsoft SQL Server a otra PC o servidor?
- ¿Con qué comando(s) podría generar dos volúmenes para un contenedor para distribuir en un volumen el Archivo de Datos (.mdf) y en otro el Archivo Log (.ldf)?

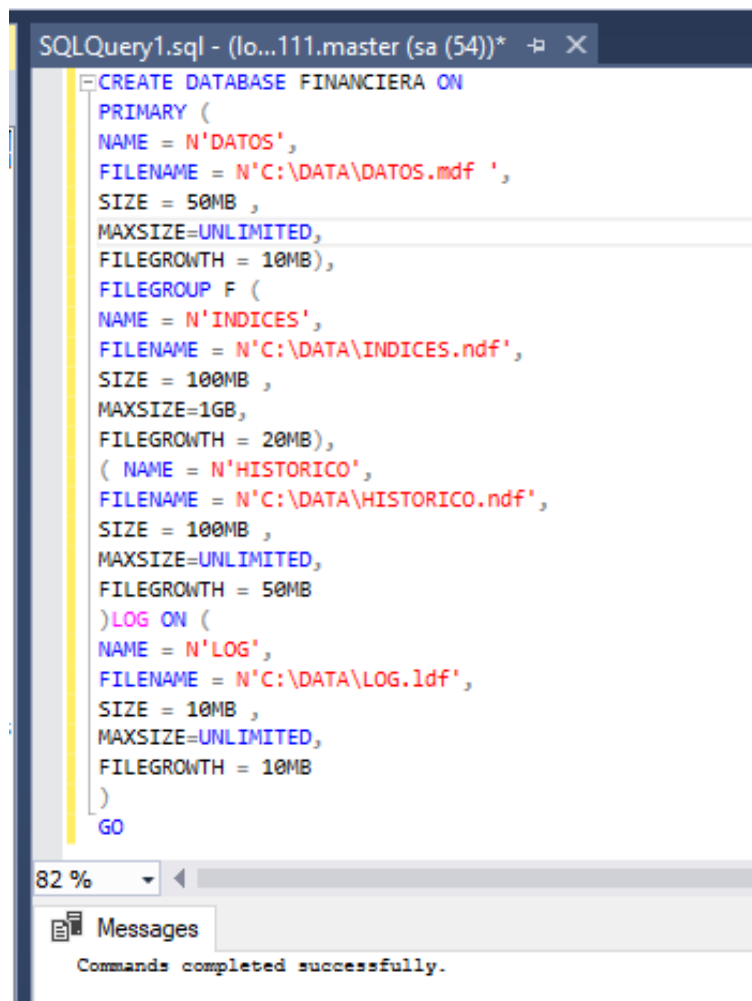
4.1.1. Genere un nuevo contenedor y cree la base de datos con las siguientes características.

Nombre : FINANCIERA

Archivos:

- DATOS (mdf) : Tamaño Inicial : 50MB, Incremento: 10MB, Ilimitado
- INDICES (ndf) Tamaño Inicial : 100MB, Incremento: 20MB, Maximo: 1GB
- HISTORICO (ndf) Tamaño Inicial : 100MB, Incremento: 50MB, Ilimitado
- LOG (ldf) Tamaño Inicial : 10MB, Incremento: 10MB, Ilimitado

4.1.2. ¿Cuál sería el script SQL que generaría esta base de datos?



```
SQLQuery1.sql - (lo...111.master (sa (54))*) X
CREATE DATABASE FINANCIERA ON
PRIMARY (
NAME = N'DATOS',
FILENAME = N'C:\DATA\DATOS.mdf',
SIZE = 50MB,
MAXSIZE=UNLIMITED,
FILEGROWTH = 10MB),
FILEGROUP F (
NAME = N'INDICES',
FILENAME = N'C:\DATA\INDICES.ndf',
SIZE = 100MB,
MAXSIZE=1GB,
FILEGROWTH = 20MB),
( NAME = N'HISTORICO',
FILENAME = N'C:\DATA\HISTORICO.ndf',
SIZE = 100MB,
MAXSIZE=UNLIMITED,
FILEGROWTH = 50MB
)LOG ON (
NAME = N'LOG',
FILENAME = N'C:\DATA\LOG.ldf',
SIZE = 10MB,
MAXSIZE=UNLIMITED,
FILEGROWTH = 10MB
)
GO

82 %
Messages
Commands completed successfully.
```


5. CONCLUSIONES

- En conclusión se puede observar que Docker es una herramienta muy útil a la hora de crear contenedores ya que hoy en día esta herramienta es mejor que los virtualizadores ya que no ocupa muchos recursos y por lo tanto es posible realizar muchísimos contenedores a diferencia de los virtualizadores, y también nos resulta que es muy útil al momento de instalar múltiples bases de datos y que no existe la necesidad de armar o instalar múltiples ordenadores físicos o virtuales.
- Es por eso que resulta factible en muchos aspectos como migrar de versión, tener varias bases de datos disponibles o además que existieran y comparen diferentes versiones de

6. WEBGRAFIA

- <https://blog.ipswitch.com/es/como-crear-su-primer-contenedor-de-windows-con-docker>
- <https://docs.microsoft.com/es-es/virtualization/windowscontainers/quick-start/quick-start-windows-10>