UNIVERSIDAD PRIVADA-DE-TACNA

INGENIERIA DE SISTEMAS

TITULO:

INFORME DE LABORATORIO No 07

CURSO:

BASE DE DATOS II

DOCENTE(ING):

Patrick Cuadros Quiroga

Integrantes:

Orestes Ramirez Ticona (2015053236)

Índice

1. INFORMACIÓN GENERAL

1.1. Objetivos:

- Conocer los fundamentos sobre Monitorización de Base de Datos mediante Auditoría.
- Poder instalar correctamente las consultas.

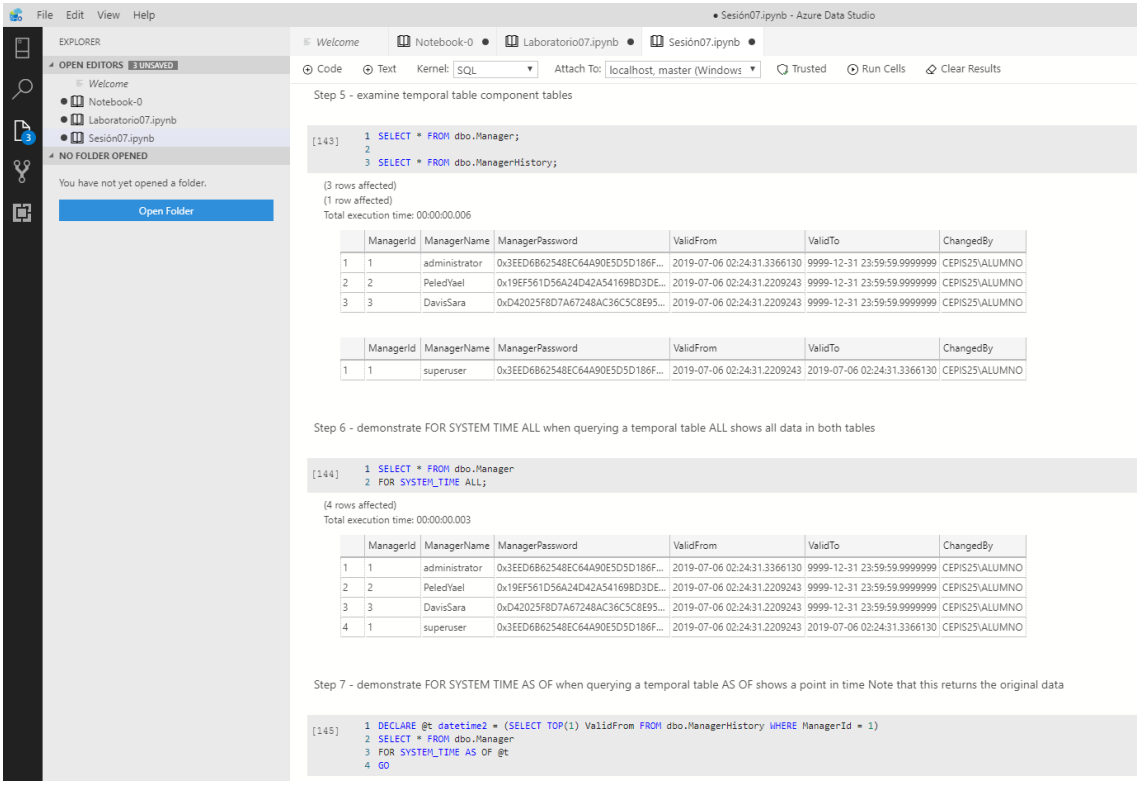
1.2. Equipos, materiales, programas y recursos utilizados:

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

2. MARCO TEORICO

2.1. AZURE DATA STUDIO

Azure Data Studio es una herramienta de base de datos multiplataforma para profesionales de datos que utilizan la familia de plataformas de datos en la nube y locales de Microsoft en Windows, MacOS y Linux.
Anteriormente publicado bajo el nombre de vista previa de SQL Operations Studio, Azure Data Studio ofrece una experiencia de edición moderna con IntelliSense, fragmentos de código, integración de control de fuente y un terminal integrado. Está diseñado teniendo en cuenta al usuario de la plataforma de datos, con un registro integrado de conjuntos de resultados de consulta y paneles personalizables.



– Editor de código SQL con IntelliSense

Azure Data Studio ofrece una experiencia moderna de codificación SQL centrada en el teclado que facilita sus tareas diarias con funciones integradas, como ventanas de pestañas múltiples, un editor de SQL enriquecido, IntelliSense, finalización de palabras clave, fragmentos de código, navegación de código y control de fuente integración (Git). Ejecute consultas SQL a pedido, vea y guarde los resultados como texto, JSON o Excel. Edite datos, organice sus conexiones de base de datos favoritas y explore los objetos de la base de datos en una experiencia familiar de búsqueda de objetos.

– Fragmentos de código de SQL inteligenter

-Los fragmentos de código SQL generan la sintaxis SQL adecuada para crear bases de datos, tablas, vistas, procedimientos almacenados, usuarios, inicios de sesión, roles, etc., y para actualizar los objetos de base de datos existentes. Use fragmentos de código inteligente para crear rápidamente copias de su base de datos para fines de desarrollo o prueba, y para generar y ejecutar scripts CREAR e INSERTAR.

- Tableros personalizables de servidores y bases de datos

-Cree ricos paneles personalizables para monitorear y solucionar rápidamente cuellos de botella de rendimiento en sus bases de datos. Para obtener información sobre los widgets de insight y los paneles de base de datos (y servidor), consulte Administrar servidores y bases de datos con widgets de insight .

- Gestión de la conexión (grupos de servidores)

- Los grupos de servidores proporcionan una forma de organizar la información de conexión para los servidores y las bases de datos con las que trabaja. Para más detalles, consulte Grupos de servidores .

- Extensibilidad y autoría de extensión.

- Mejore la experiencia de Azure Data Studio al ampliar la funcionalidad de la instalación básica. Azure Data Studio proporciona puntos de extensibilidad para las actividades de administración de datos, así como soporte para la creación de extensiones.

3. PROCEDIMIENTO

3.1. Parte 1: Iniciando Docker:

– Abrir el menu inicio y buscar la aplicación Docker for Windows.

```
[140]  1 CREATE TABLE dbo.Manager
       2 ( ManagerId int NOT NULL PRIMARY KEY,
       3   ManagerName nvarchar(50) NOT NULL,
       4   ManagerPassword varbinary(200) NOT NULL,
       5   ValidFrom datetime2 GENERATED ALWAYS AS ROW START NOT NULL,
       6   ValidTo datetime2 GENERATED ALWAYS AS ROW END NOT NULL,
       7   ChangedBy sysname NOT NULL CONSTRAINT DF_Employee_ChangedBy DEFAULT  (SUSER_SNAME()),
       8   PERIOD FOR SYSTEM_TIME (ValidFrom, ValidTo)
       9 ) WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = dbo.ManagerHistory));
      10 GO
```

– Una vez iniciado se podrá visualizar el icono de Docker en el área de notificación.

```
[141]  1 INSERT dbo.Manager (ManagerId, ManagerName, ManagerPassword)
       2 VALUES (1, N'superuser', 0x3EED6B62548EC64A90E5D5D186FC9E5C),
       3 (2, N'PeledYael', 0x19EF561D56A24D42A54169BD3DE23652),
       4 (3, N'DavisSara', 0xD42025F8D7A67248AC36C5C8E955FA71);
       5 GO
```

– Asimismo se podrá visualizar la ventana de bienvenida.

– Ingresar sus credenciales creadas en Docker Hub para iniciar sesión en el aplicativo..

```
[142]  1 UPDATE dbo.Manager
       2 SET ManagerPassword = 0x3EED6B62548EC64A90E5D5D186FCFFFF,
       3 ManagerName = 'administrator'
       4 WHERE ManagerId = 1
```

– Ubicar la aplicación PowerShell, ejecutarla como Administrador. En la ventana de comandos de PowerShell escribir lo siguiente: ”docker versión”

```
[143]  1 SELECT * FROM dbo.Manager;
       2
       3 SELECT * FROM dbo.ManagerHistory;
```

```
[144]  1 SELECT * FROM dbo.Manager
       2 FOR SYSTEM_TIME ALL;
```

– Verificar que el resultado sea el siguiente.

```
[146]  1 UPDATE dbo.ManagerHistory SET ChangedBy = 'maliciousUser';
       2 GO
       3 INSERT dbo.ManagerHistory (ManagerId, ManagerName, ManagerPassword)
       4 VALUES (99, N'superuser', 0x3EED6B62548EC64A90E5D5D186FC9E5C)
       5 GO
```

3.2. Parte 2: Creando un contenedor con Microsoft SQL Server para Linux

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

– El resultado deberá ser algo similar a lo siguiente.

```
[147]  1 UPDATE dbo.Manager
       2 SET ManagerPassword = 0x0A0B,
       3 ManagerName = 'hacked', ChangedBy = 'maliciousUser'
       4 WHERE ManagerId = 1
```

– Ahora ejecutar el comando:"docker pull microsoft/mssql-server-linux"

– Lo cual descargará la imagen del contenedor de Microsoft SQL Server en un servidor Linux

```
[148]  1 SELECT * FROM dbo.Manager;
       2
       3 SELECT * FROM dbo.ManagerHistory;
```

– Proceder a verificar la imagen con el siguiente comando:"docker images"

– Lo cual deberá visualizar lo siguiente:

```
[149]  1 ALTER TABLE dbo.Manager SET (SYSTEM_VERSIONING = OFF);
       2 DROP TABLE dbo.Manager;
       3 DROP TABLE dbo.ManagerHistory;
```

– Seguidamente ejecutar el comando:

```
       1 USE master;
       2 GO
       3 CREATE SERVER AUDIT MIASQL_Audit
       4     TO FILE (FILEPATH='D:\salesapp1\')
       5     WITH (QUEUE_DELAY = 5000);
       6 GO
```

– Como respuesta se visualizará un ID que corresponde al contenedor:

```
[184]  1 ALTER SERVER AUDIT MIASQL_Audit WITH (STATE = ON);
       2 GO
```

– Verificar que el contenedor se este ejecutando correctamente mediante el comando:"docker ps"

– Si se visualiza un cuadro de dialogo de permisos relacionados al firewall Windows, Aceptarlo para realizar la conexión.El resultado será similar al siguiente:

```
[185]  1 CREATE SERVER AUDIT SPECIFICATION AuditLogins
       2 FOR SERVER AUDIT MIASQL_Audit
       3 ADD (FAILED_LOGIN_GROUP),
       4 ADD (SUCCESSFUL_LOGIN_GROUP)
       5 WITH (STATE = ON);
       6 GO
```

– . Esperar unos segundos e iniciar la aplicación Microsoft SQL Server Management Studio, y conectar con los siguientes datos:

```
1 USE salesapp1;
2 GO
3 CREATE DATABASE AUDIT SPECIFICATION salesapp1_audit_spec
4 FOR SERVER AUDIT MIASQL_Audit
5 ADD (INSERT,UPDATE ON DATABASE::salesapp1 BY public),
6 ADD (SELECT ON SCHEMA::HR BY public),
7 ADD (SCHEMA_OBJECT_CHANGE_GROUP)
8 WITH (STATE = ON);
9 GO
```

– Iniciar una nueva consulta, escribir y ejecutar lo siguiente:

```
1 USE salesapp1
2 GO
3 ALTER DATABASE AUDIT SPECIFICATION salesapp1_audit_spec WITH (STATE = OFF);
4 GO
5 ALTER DATABASE AUDIT SPECIFICATION salesapp1_audit_spec
6 ADD (SCHEMA_OBJECT_PERMISSION_CHANGE_GROUP)
7 WITH (STATE = ON);
8 GO
```

– Deberá retornar algo similar a lo siguiente:

```
[188]   1 SELECT * FROM sys.server_audits;
```

– Cerrar la aplicación Microsoft SQL Server Management Studio.

– En PowerShell ejecutar el siguiente comando:"docker rm -f SQLLNX01"

```
[189]   1 SELECT * FROM sys.server_audit_specifications;
        2
        3 SELECT *
        4 FROM sys.server_audit_specification_details AS sd
        5 JOIN sys.dm_audit_actions AS aa
        6 ON aa.name = sd.audit_action_name COLLATE Latin1_General_CI_AS_KS_WS
```

– Verificar la eliminación del contenedor con ejecutando: docker ps

```
[190]   1 SELECT * FROM sys.database_audit_specifications;
        2
        3 SELECT *
        4 FROM sys.database_audit_specification_details AS sd
        5 JOIN sys.dm_audit_actions AS aa
        6 ON aa.name = sd.audit_action_name COLLATE Latin1_General_CI_AS_KS_WS
        7 AND aa.class_desc = sd.class_desc COLLATE Latin1_General_CI_AS_KS_WS
```

3.3. Parte 3: Adicionando persistencia

– En PowerShell ejecutar el siguiente comando.

```
[191]   1 USE master;
        2 GO
        3 ALTER SERVER AUDIT MIASQL_Audit WITH (STATE = OFF);
        4 DROP SERVER AUDIT MIASQL_Audit;
        5 GO
        6
        7 ALTER SERVER AUDIT SPECIFICATION AuditLogins WITH (STATE = OFF);
        8 DROP SERVER AUDIT SPECIFICATION AuditLogins
        9 GO
       10
       11 USE salesapp1;
       12 GO
       13 ALTER DATABASE AUDIT SPECIFICATION salesapp1_audit_spec WITH (STATE = OFF);
       14 DROP DATABASE AUDIT SPECIFICATION salesapp1_audit_spec
       15 GO
```

– luego visualizara la siguiente ventana ingresamos las credenciales.

[192]
```
USE master;
GO
CREATE SERVER AUDIT Custom_Audit
    TO FILE (FILEPATH='F:\Data\')
    WITH (QUEUE_DELAY = 5000);
GO
ALTER SERVER AUDIT Custom_Audit WITH (STATE = ON);
GO
```

– Como respuesta se visualizará un ID que corresponde al contenedor:

[193]
```
CREATE SERVER AUDIT SPECIFICATION UserDefinedEvents
FOR SERVER AUDIT Custom_Audit
ADD (USER_DEFINED_AUDIT_GROUP)
WITH (STATE = ON);
GO
```

– Verificar que el contenedor se este ejecutando correctamente mediante el comando:

[194]
```
EXEC sp_audit_write @user_defined_event_id = 999,
                    @succeeded = 1,
                    @user_defined_information = N'Example call to sp_audit_write';
```

– Esperar unos segundos e iniciar la aplicación Microsoft SQL Server Management Studio, y conectar con los siguientes datos:

[195]
```
SELECT user_defined_event_id, succeeded, user_defined_information
FROM sys.fn_get_audit_file ('F:\Data\Custom_Audit*',default,default)
WHERE user_defined_event_id = 999;

```

– Generar una base de datos de prueba en la aplicación Microsoft SQL Server Management Studio, según la siguiente imagen mediante el siguiente script:

```sql
USE salesapp1;
GO

CREATE PROC usp_OrderDetailDiscount
    @orderid int,
    @productid int,
    @discount numeric(4,3)
AS
    SET NOCOUNT ON

    IF @discount > 0.3
    BEGIN
        DECLARE @msg nvarchar(4000) =
          CONCAT('Order=',@orderid,':Product=',@productid,
                 ':discount=', @discount)


        EXEC sp_audit_write @user_defined_event_id = 998,
                            @succeeded = 1,
                            @user_defined_information = @msg;
    END

    UPDATE Sales.OrderDetails
    SET discount = @discount
    WHERE orderid = @orderid
    AND productid = @productid
GO
```

– Verificar el contenido la carpeta DATALNX

[197]
```sql
EXEC dbo.usp_OrderDetailDiscount @orderid = 10248,@productid =  11, @discount = 0.05
EXEC dbo.usp_OrderDetailDiscount @orderid = 10248,@productid =  42, @discount = 0.45
```

– En PowerShell ejecutar el siguiente comando:"docker rm -f SQLLNX02"

[198]
```sql
SELECT user_defined_event_id, succeeded, user_defined_information
FROM sys.fn_get_audit_file ('F:\Data\Custom_Audit*',default,default)
WHERE user_defined_event_id = 998;
```

– Verificar la eliminación del contenedor con ejecutando:"docker ps"

[199]
```sql
USE master;
GO
ALTER SERVER AUDIT Custom_Audit WITH (STATE = OFF);
DROP SERVER AUDIT Custom_Audit;
GO

ALTER SERVER AUDIT SPECIFICATION UserDefinedEvents WITH (STATE = OFF);
DROP SERVER AUDIT SPECIFICATION UserDefinedEvents
GO
```

– antes se debe compartir o elnazar la carpeta con Docker

```
[200]
1 USE master;
2 GO
3 CREATE SERVER AUDIT File_Audit
4     TO FILE (FILEPATH='F:\Data\')
5     WITH (QUEUE_DELAY = 5000);
6 GO
7 ALTER SERVER AUDIT File_Audit WITH (STATE = ON);
8 GO
```

3.4. Parte 4: Creando un contenedor con Microsoft SQL Server para Windows

– En el icono de Docker en el área de notificación, hacer click con el botón derecho y utilizar la opción Switch to Windows Containers. Esperar a que Docker se reinicie.

```
[201]
1 CREATE SERVER AUDIT AppLog_Audit
2     TO APPLICATION_LOG
3     WITH (QUEUE_DELAY = 5000);
4 GO
5 ALTER SERVER AUDIT AppLog_Audit WITH (STATE = ON);
6 GO
```

```
[202]
1 USE salesapp1;
2 GO
3 CREATE DATABASE AUDIT SPECIFICATION sales_select_spec_file
4 FOR SERVER AUDIT File_Audit
5 ADD (SELECT ON SCHEMA::Sales BY public)
6 WITH (STATE = ON);
7 GO
8
9 CREATE DATABASE AUDIT SPECIFICATION sales_select_spec_applog
10 FOR SERVER AUDIT AppLog_Audit
11 ADD (SELECT ON SCHEMA::Sales BY public)
12 WITH (STATE = ON);
13 GO
```

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

```
[203]
1 SELECT TOP 10 * FROM Sales.Customers;
2 GO
```

– Ejecutar el siguiente comando; lo cual descargará la imagen del contenedor de Microsoft SQL Server en un servidor Linux.

```
[204]   1 SELECT *
        2 FROM sys.fn_get_audit_file ('F:\Data\File_Audit*',default,default)
```

– Proceder a verificar la imagen con el siguiente comando:

```
PS C:\> docker images
REPOSITORY                                  TAG                 IMAGE ID            CREATED
         SIZE
microsoft/mssql-server-windows-developer    latest              19873f41b375        16 months a
go       15.1GB
```

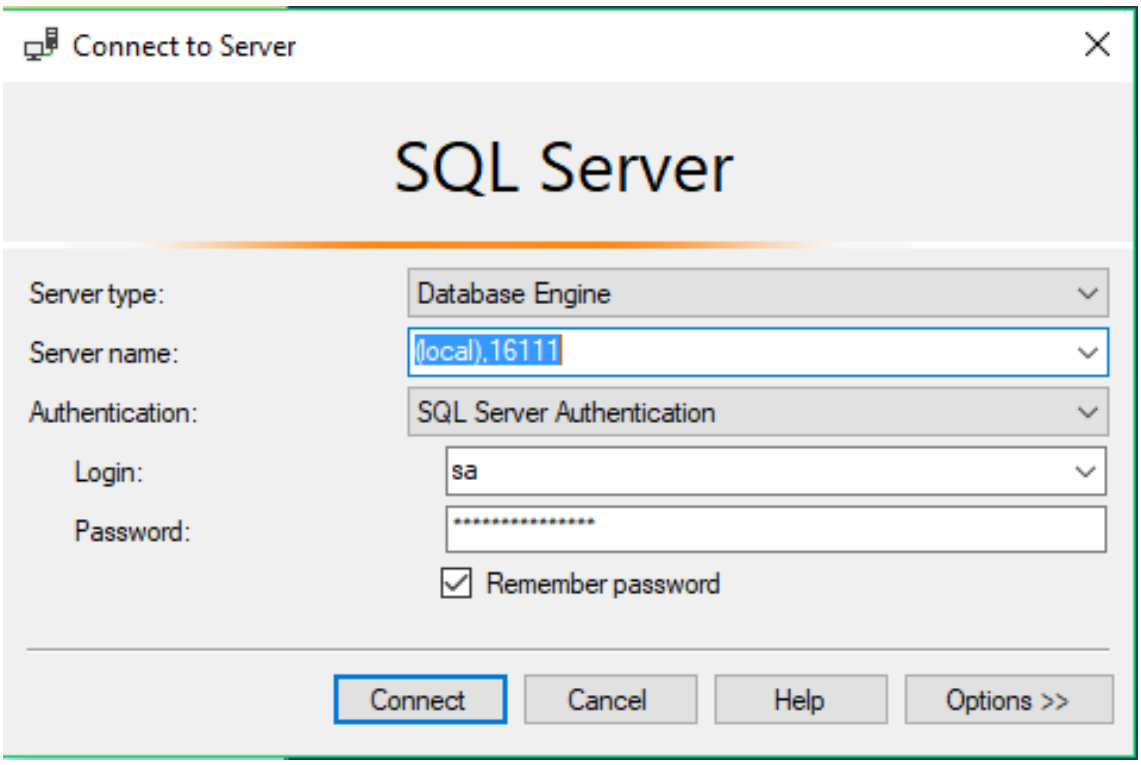
– Seguidamente ejecutar el comando; como respuesta se visualizará un ID que corresponde al contenedor

```
PS C:\> docker run -d -p 16111:1433 -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=Tacna.2019' -v D:\DATAW
IN:C:\DATA --name SQLWIN01 microsoft/mssql-server-windows-developer
9ef230cc27f464115834d33d9a042d4ad3c78599194566b2e31a0c301457bfec
```

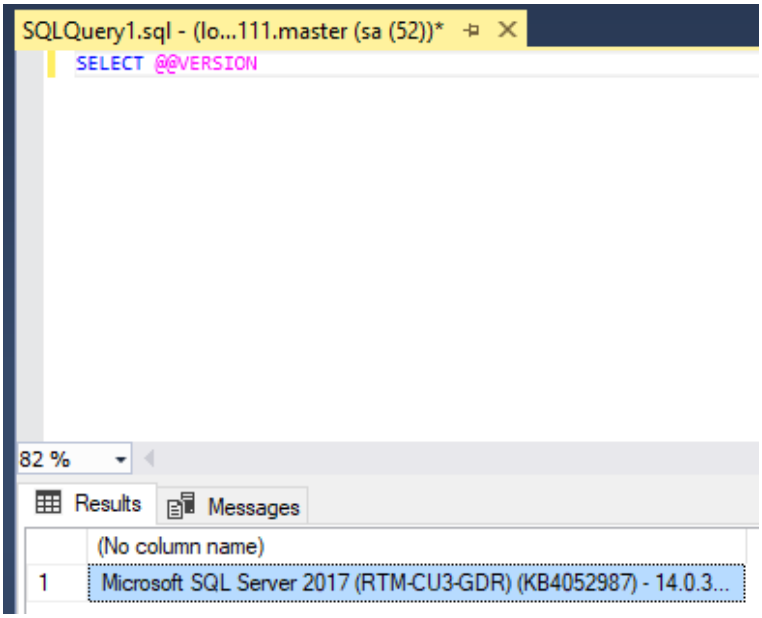
– Repetir el paso 10 y verificar que el contenedor este ejecutándose

```
PS C:\> docker ps
CONTAINER ID        IMAGE                                       COMMAND                  CREATE
D                   STATUS                          PORTS                     NAMES
9ef230cc27f4        microsoft/mssql-server-windows-developer    "powershell -Command…"   2 minu
tes ago             Up 22 seconds (health: starting)   0.0.0.0:16111->1433/tcp   SQLWIN01
```

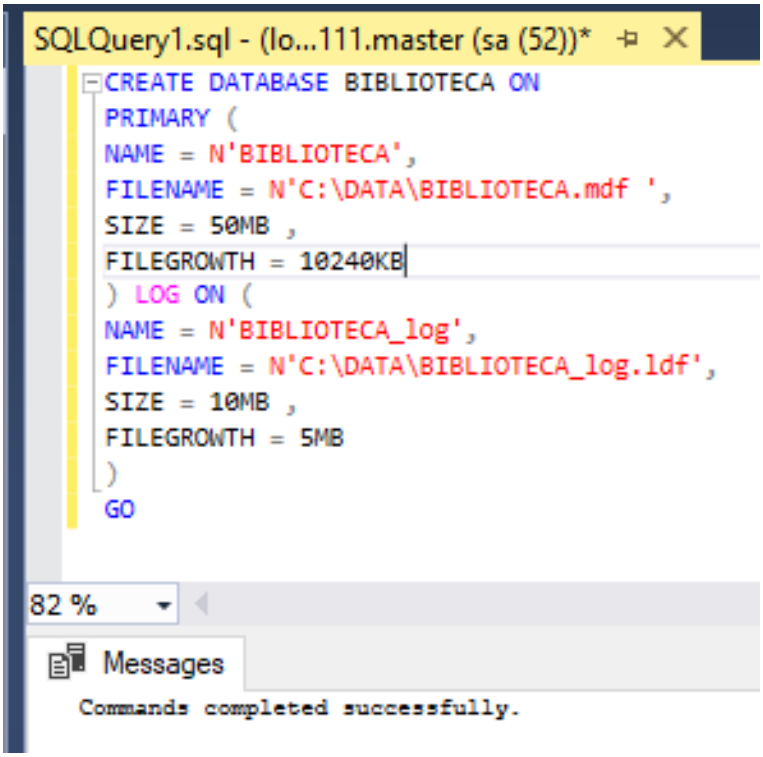
– Repetir el paso 11 y conectar al servidor



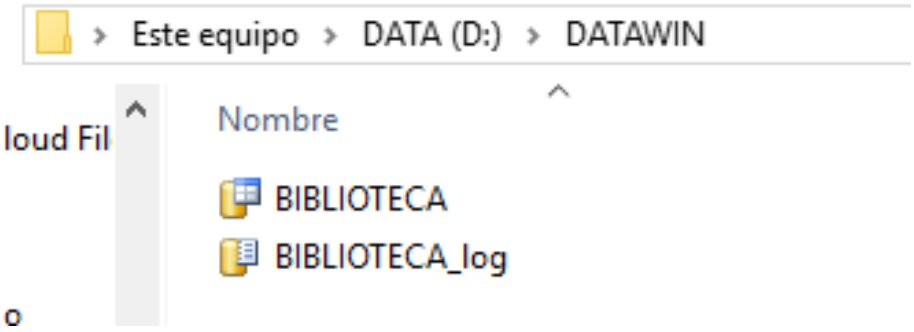
– Iniciar una nueva consulta, escribir y ejecutar lo siguiente:



– Generar una base de datos de prueba en la aplicación Microsoft SQL Server Management Studio



– Verificar el contenido de la carpeta DATAWIN



– En PowerShell ejecutar el siguiente comando y verificar la eliminación del contenedor con ejecutando

– Cerrar la aplicación Microsoft SQL Server Management Studio.

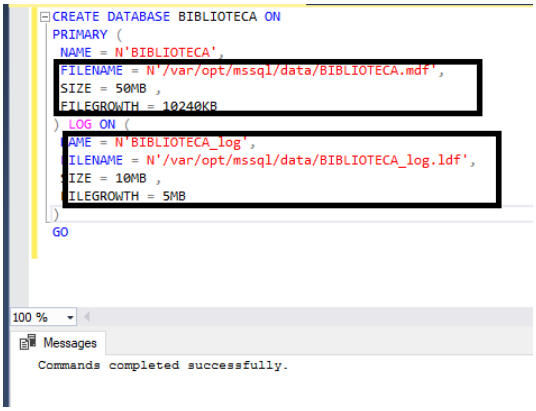
4. ANALISIS E INTERPRETACION DE RESULTADOS

4.1. Parte 1: Actividades Encargadas

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

```
PS C:\> docker run -d -p 16111:1433 -e 'ACCEPT_EULA=Y' -e 'SA_PASSWORD=Tacna.2019' --name SQLLNX01 microsoft/mssql-server-linux
b34721cfdaae2f5811c876da2c00965ffdcefdeb5bf0a6fe801a02d11ea9ebb5
PS C:\>
```

– ¿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)?



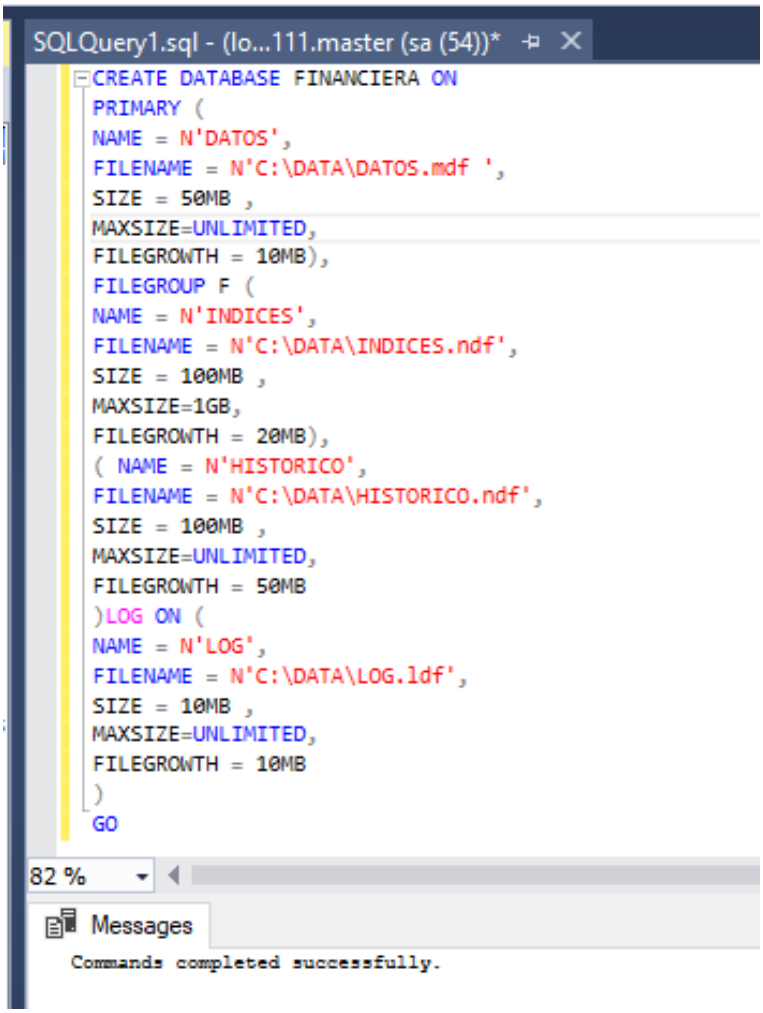
– Genere un nuevo contenedor y cree la base de datos con las siguientes características.

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

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


SQLQuery1.sql - (lo...111.master (sa (54))*
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 hemos observado y experimentado con docker, y nos resulta que es muy util al momento de instalar multiples bases de datos y que no existe la necesidad de armar o instalar múltipler ordenadores físicos o virtuales.

- Es por eso que resulta factible en muchos aspectos como migrar de version, tener varias bases de datos disponibles o además que existieran y comparen diferentes versiones de bases de datos a la vez

6. REFERENCIAS

[1] Hat, R. (2017). ¿Qué es Docker?. Recuperado de https://www.redhat.com/es/topics/containers/what-is-docker

[2] Why Docker? — Docker. (2017). Recuperado de https://www.docker.com/why-docker

[3] Araujo, J. (2017). ¿Qué es Docker? ¿Que son los contenedores? y ¿Por que no usar VMs?. Recuperado de https://platzi.com/contributions/guia-del-curso-de-docker/

[4] Oterino, A. (2015). ¿Qué es Docker? ¿Para qué se utiliza? Explicado de forma sencilla. Recuperado de https://www.javiergarzas.com/2015/07/que-es-docker-sencillo.html