

ASAX

SEGURIDAD SQL SERVER

MANUEL BEADE BOÁN

- 
- 01 Mecanismos de Seguridad en SQL Server.
 - 02 Docker

INDICE

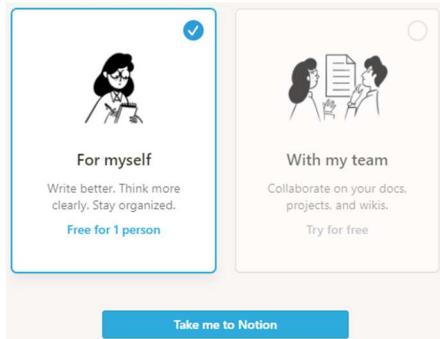
INTRODUCCIÓN	4
Enlaces de interés	5
SEGURIDAD	6
ENcriptación	6
TOOLS (VERACRYPT)	9
Instalando Veracrypt	9
Crear un contenedor de archivos cifrado.....	13
Encriptación de Columnas de Base de datos.....	21
Encriptación de Backup de Base de datos	29
Encriptación de Base de datos TDE (Transparent Data Encryption).....	35
Encryption vs Hashing.....	39
Dynamic Data Masking	42
Default	45
Partial	46
Random.....	47
Custom	48
Pruebas de Desenmascaramiento	49
Row Level Security	51
Funciones	54
Trigger recuperación primera evaluación	69
Always Encrypted.....	74
Auditoría	82
LEGISLACIÓN	88
Detectando y clasificando datos personales con SQL Server	89
Detectar: Data Discovery and Classification & Vulnerability and Assesment	90
STORED PROCEDURE + TEMPORAL TABLES RECUPERACIÓN	90
Clasificar	99
Administrando los accesos y el uso de los datos con SQL Server	102
Protegiendo datos en Reposo, en Uso y en Tránsito.....	104
Monitorizar y Reportar accesos no autorizados a Datos Personales	109
ATAQUES.....	110
DDos	110
SQL INJECTION	117
Tipos de inyección SQL.....	117
Como detectar las vulnerabilidades.....	118

Inyección SQL en diferentes partes de la consulta.....	118
Prevenir la inyección SQL.....	119
Ransomware	119
¿Cómo se propaga el ransomware?	119
Cómo protegerse contra el ransomware.....	119
Cómo recuperarse de un desastre	120
Tools.....	121
DDoS	121
SQL Injection	122
Ransomware	123
DOCKER.....	127
Comandos básicos	130
HERRAMIENTAS DEL PROYECTO.....	131
Instalación de Visual Code Studio	131
Instalación de Docker Engine.....	134
PRIMEROS PASOS CON DOCKER:	136
Comenzar con el contenedor Hellow-World	136
Uso de los comandos básicos de Docker	136
Lanzar un contenedor de MySQL	139
MySQL con/sin presencia de datos	145
Ejemplo creación de un contendor MySQL sin persistencia de datos	145
Ejemplo creación de un contendor MySQL sin persistencia de datos.....	146
Uso del flag -link y de user-defined bridge network.....	147
Utilizando un flag-link.....	147
Utilizando un user-defined bridge network	150
Lanzar un contenedor de phpMyAdmin.....	152
Utilizando el flag --link.....	152
Utilizando una user-defined bridge network	153
Lanzar múltiples contenedores: Wordpress + MySQL + phpMyAdmin	155
Creación de un contenedor Docker con MariaDB.....	160
Creación de un contenedor con PostgreSQL.....	161
MongoDB	163
Docker Hub	169
Creando un repositorio.....	170
Conectándose a Docker Hub.....	172
BIBLIOGRAFÍA	173

INTRODUCCIÓN

Esta parte es la continuación del proyecto presentado en la primera evaluación. Para tal cometido, haremos uso de una nueva herramienta: **Notion**. Notion es un organizador de tareas multiplataforma que permite de manera online u offline, en cualquier plataforma, apuntar y organizar todo y colaborar con quien se desee; en resumen, es una herramienta de gestión de proyectos. Para comenzar a utilizarla, nos inscribimos de manera gratuita en ella en <https://www.notion.so/>, dándole al botón **Sign up**.

Una vez metemos nuestro mail, aceptamos las políticas en **Agree**. Cuando hacemos esto, nos da a elegir si queremos que sea para nosotros. En mi caso elijo para mí.



Tras unos minutos de carga, comienza la aplicación con un **Getting Started**, que es como un tutorial de la página. Para nuestro proyecto, le damos a **+ Add a page BBM**. Se abre una nueva página, de la siguiente manera:

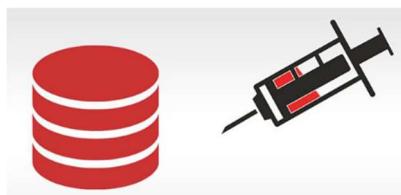
The image shows a screenshot of a new Notion page titled 'Untitled'. The page displays a template selection menu with the following options: 'Empty with icon' (selected), 'Empty', 'Templates', 'Import', 'DATABASE' (with sub-options: 'Table', 'Board', 'List', 'Calendar', 'Gallery', 'Timeline'), and 'BBM' (which is highlighted with a green dotted border). Below the menu, there is placeholder text: 'Press Enter to continue with an empty page, or pick a template (↑↓ to select)'. The overall interface is clean and modern, with a light gray background and white text.

Como explicar esto es muy extenso, dejo el resultado final de mi template.

Aquí iré poniendo este mismo trabajo, pero dividido en hojas para poder hacer una presentación final del trabajo.

La segunda parte del proyecto aborda la parte de seguridad en SQL. Aquí se explicarán temas como **encriptación**, **auditorías**, algunos tipos de **ataques a SQL Server**, y se abordará un poco el tema de la **Ley de Protección de Datos**.

El uso de herramientas será SQL Server en sí mismo, aunque haré una pequeña explicación y uso de **VeraCrypt**.



En esta parte del proyecto también se explicará qué es **Docker**. Aquí habrá una breve explicación de los comandos básicos, pasando por conectarse de diferentes maneras a las distintas plataformas (MySQL, MaríaDB...).

Una vez hecho esto, me logaré en Docker Hub y haré un link con él.

Enlaces de interés

<https://www.notion.so/SCRIPTS-2a1a6f83a094475da10bb65f56148db7>

<https://github.com/BBMASAX/BBMASPACE>

<https://hub.docker.com/repository/docker/bbminspace/bbm>

SEGURIDAD

ENCRIPCIÓN

Una estrategia de defensa exhaustiva, con niveles superpuestos de seguridad, es la mejor manera de enfrentarse a las amenazas a la seguridad. SQL Server proporciona una arquitectura de seguridad diseñada para permitir a los administradores de bases de datos y desarrolladores crear aplicaciones de base de datos seguras y contrarrestar las amenazas.

En cada versión de SQL Server se han introducido mejoras a las versiones anteriores con nuevas características y funcionalidades. No obstante, la seguridad no es una característica integrada más. Cada **aplicación tiene requisitos de seguridad propios**. Los desarrolladores tienen que saber cuál es la combinación de características y funcionalidades más apropiada para contrarrestar las amenazas conocidas, así como anticiparse a las que puedan ir apareciendo en el futuro.

Una **instancia de SQL Server contiene un conjunto jerárquico de entidades**, empezando por el servidor. Cada servidor contiene varias bases de datos y, a su vez, cada base de datos contiene un conjunto de **objetos susceptibles de ser protegidos**. Cada SQL Server protegible tiene permisos asociados que se pueden conceder a una entidad de seguridad, que es una persona, grupo o proceso al que se concede acceso a SQL Server. El marco de seguridad de SQL Server administra el acceso a las entidades protegibles a través de la **autenticación y la autorización**.

- La autenticación es el proceso de inicio de sesión en SQL Server por el que una entidad de seguridad solicita el acceso mediante el envío de credenciales que el servidor evalúa. La autenticación establece la identidad del usuario o proceso que se autentica.
- La autorización es el proceso en el que se determinan los recursos protegibles a los que puede obtener acceso una entidad de seguridad y las operaciones permitidas para dichos recursos.

Para este proyecto, vamos a ver de la parte de seguridad:

- Encriptación.
- Auditoría
- Algunos tipos de ataques

La encriptación es un recurso muy utilizado hoy en día para garantizar una **transferencia segura de datos y documentos**. Si bien no se puede garantizar que no se sustraiga información sensible, sí puede evitar que se utilice para el perjuicio de sus dueños legítimos.

La banca y los comercios en línea usan la encriptación de datos para evitar el manejo inapropiado de información de sus clientes (números de tarjetas de crédito, información sobre transacciones, datos personales, etc.).

De la misma forma, **muchos sistemas de mensajería recurren a esta herramienta para procurar comunicaciones más seguras** y evitar que las conversaciones sean interceptadas.

Si bien la terminología asociada a los procesos de cifrado es más común con el auge de la tecnología y la necesidad de proteger los datos que se manejan por internet, la realidad es que desde hace milenios se utilizan técnicas para proteger información valiosa.

Se sabe que, en la Antigüedad, los egipcios utilizaban mensajes cifrados, cuyos datos eran sustituidos, alterados o permutados, con fines militares.

Durante la Segunda Guerra Mundial, el ejército alemán utilizó una máquina de cifrado llamada Enigma, que le permitía enviar y recibir información sensible sin ser detectada.

Sin embargo, el británico Alan Turing y su equipo de trabajo lograron descifrar los mensajes encriptados, y con ello encontraron el camino para que los Aliados pudiesen lograr la victoria y poner fin a la guerra.

Los métodos de encriptado se clasifican según sus claves y sus algoritmos.

- **Encriptación según sus claves.** Hay dos tipos:

La **encriptación simétrica** es aquella donde se utiliza la misma clave tanto para cifrar como para descifrar los datos. Algunos de los sistemas de encriptación simétricos más populares son AES (*Advanced Encryption Standard*), DES (*Data Encryption Standard*) y Triple DES.

Encriptación asimétrica. Consta de una clave pública para cifrar y una clave privada para descifrar. Los métodos más conocidos son ElGamal (llamado así en honor a su creador, Taher ElGamal) y RSA (Rivest, Shamir y Adleman).

Este método se utiliza para el cifrado de mensajes vía email.

- **Encriptación según sus algoritmos**

Encriptación en flujo. Se utilizan claves muy largas para cifrar, las cuales pueden ser predeterminadas o creadas aleatoriamente usando un generador de claves. En muchos casos, el propio mensaje a encriptar forma parte de la clave, y esta debe mantenerse en secreto.

Con este tipo de encriptado, se pueden proteger conversaciones telefónicas y aplicaciones de audio y video que operen en tiempo real.

Encriptación por bloques. El mensaje o los datos a cifrar se descomponen en bloques de la misma longitud para proceder a la encriptación de cada uno de ellos. Este sistema, a su vez, puede ser simétrico o asimétrico.

Los sistemas de encriptación DES y Triple DES utilizan codificación por bloques.

Antes de ejecutar cualquier script, lo primero que hacemos es un *backup* de nuestra base de datos. Lo hacemos a través del procedimiento almacenado utilizado en nuestro proyecto anterior

```
CREATE PROC BBM_SPBACKUP
    @path VARCHAR(256)
AS
DECLARE @name VARCHAR(50), -- database name
-- @path VARCHAR(256), -- path for backup files
@fileName VARCHAR(256), -- filename for backup
@fileDate VARCHAR(20), -- used for file name
@backupCount INT

CREATE TABLE [dbo].#BBM_TEMPBACKUP
(intID INT IDENTITY (1, 1),
name VARCHAR(200))
-- Crear la Carpeta Backup
SET @path = 'C:\BBM_BACKUPS\' 
--- Includes the date in the filename
SET @fileDate = CONVERT(VARCHAR(20), GETDATE(), 112)
-- Includes the date and time in the filename
INSERT INTO [dbo].#BBM_TEMPBACKUP (name)
    SELECT name
    FROM master.dbo.sysdatabases
    WHERE name IN ('BBM_ASPACE')
-- WHERE name NOT IN ('master', 'model', 'msdb', 'tempdb')

SELECT TOP 1 @backupCount = intID
FROM [dbo].#BBM_TEMPBACKUP
ORDER BY intID DESC

-- Utilidad: Solo Comprobaci n Backups a realizar
print @backupCount

IF ((@backupCount IS NOT NULL) AND (@backupCount > 0))
BEGIN
    DECLARE @currentBackup INT
    SET @currentBackup = 1
    WHILE (@currentBackup <= @backupCount)
        BEGIN
            SELECT
                @name = name,
                @fileName = @path + name + '_' + @fileDate + '.BAK'
                FROM [dbo].#BBM_TEMPBACKUP
                WHERE intID = @currentBackup
            -- Utilidad: Solo Comprobaci n nombre Backup
            print @fileName
            -- does not overwrite the existing file
            BACKUP DATABASE @name TO DISK = @fileName
            SET @currentBackup = @currentBackup + 1
        END
    END
    GO
-- Ejecutar Procedimiento
-- Input Parameter 'C:\Backup\' 
EXEC BBM_SPBACKUP 'C:\BBM_BACKUPS\' 
GO

(1 row affected)
1
C:\BBM_BACKUPS\BBM_ASPACE_20210429.BAK
Processed 456 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_PRINCIPAL' on file 1.
Processed 112 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_SECONDARY' on file 1.
Processed 2 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_log' on file 1.
BACKUP DATABASE successfully processed 570 pages in 0.785 seconds (5.665 MB/sec).
```

TOOLS (VERACRYPT)

VeraCrypt es un software de código abierto para **cifrar archivos, carpetas, unidades USB extraíbles, discos duros completos, e incluso el disco duro donde se encuentra el propio sistema operativo instalado**. VeraCrypt es **multiplataforma**, actualmente es compatible con sistemas operativos Microsoft Windows, cualquier sistema basado en Linux, y también es compatible con macOS. Este software está basado en el popular **TrueCrypt 7.1a**.

Algunas de las principales características de VeraCrypt son las siguientes:

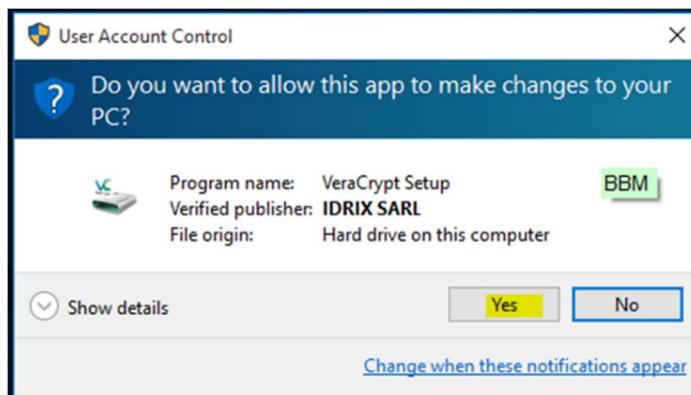
- **Creación de discos cifrados virtuales en un simple archivo:** podremos crear un archivo cifrado a modo de contenedor, en el cual esté toda la información importante. Este archivo lo podremos montar para su lectura y escritura con VeraCrypt, este método es ideal para moverlo a cualquier sitio e incluso para enviarlo por email, subirlo a un servidor FTP o Samba y más. Gracias a que tenemos un simple archivo que contiene toda la información confidencial, se puede guardar a buen recaudo grabándolo en un CD o DVD, e incluso copiarlo en un pendrive.
- **Cifrado de dispositivos de almacenamiento extraíble** como USB, tarjetas SD e incluso discos duros. En este caso, todo el dispositivo de almacenamiento extraíble estará completamente cifrado, Windows nos indicará que necesita formato el disco para poder leerlo, siempre debemos pinchar en cancelar y abrirlo con VeraCrypt, introduciendo la correspondiente clave de descifrado.
- **Cifrado de cualquier partición** de estos dispositivos de almacenamiento extraíble.
- **Cifrado de la partición o disco completo donde Windows esté instalado.** Esto nos permite hacer exactamente la misma función que Bitlocker, cifrará el disco duro o SSD por completo, para que tanto el sistema operativo como todos nuestros archivos estén a salvo frente a posibles robos.
- **El cifrado y el descifrado es automático y se hace en tiempo real**, siendo completamente transparente al usuario.
- **El cifrado y descifrado si utilizamos AES se puede acelerar** si el procesador del equipo soporta AES-NI, proporcionando una mayor velocidad de lectura y escritura.
- **Posibilidad de crear un volumen «oculto»** para evitar que un posible atacante nos fuerce a revelar la contraseña del volumen

Instalando Veracrypt

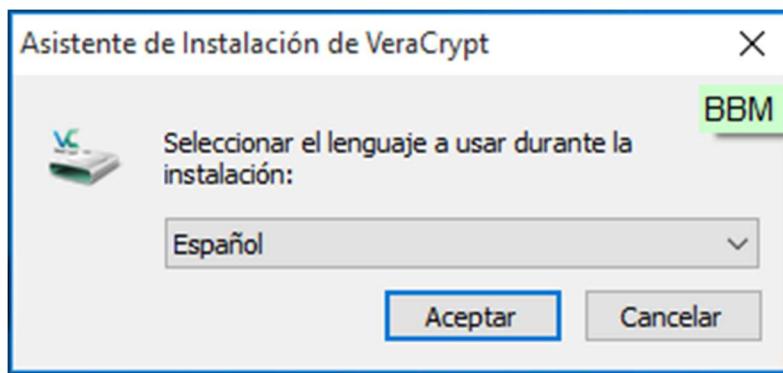
Nos vamos a esta página <https://www.veracrypt.fr/code/VeraCrypt/>, y en el botón **Downloads**, elegimos la opción de instalación de Windows.

-  **Windows:**
 - Installer for Windows 8 and later: [VeraCrypt Setup 1.24-Update7.exe](#) (4.5 MB) [\(PGP Signature\)](#) 
 - Portable version for Windows 8 and later: [VeraCrypt Portable 1.24-Update7.exe](#) (34.3 MB) [\(PGP Signature\)](#)
 - Installer for Windows XP, Vista and 7: [VeraCrypt Legacy Setup 1.24-Update7.exe](#) (34.5 MB) [\(PGP Signature\)](#)
 - Portable version for Windows XP, Vista and 7: [VeraCrypt Legacy Portable 1.24-Update7.exe](#) (34.3 MB) [\(PGP Signature\)](#)
 - Debugging Symbols: [VeraCrypt 1.24-Update7_Windows_Symbols.zip](#) (9.68 MB) [\(PGP Signature\)](#)

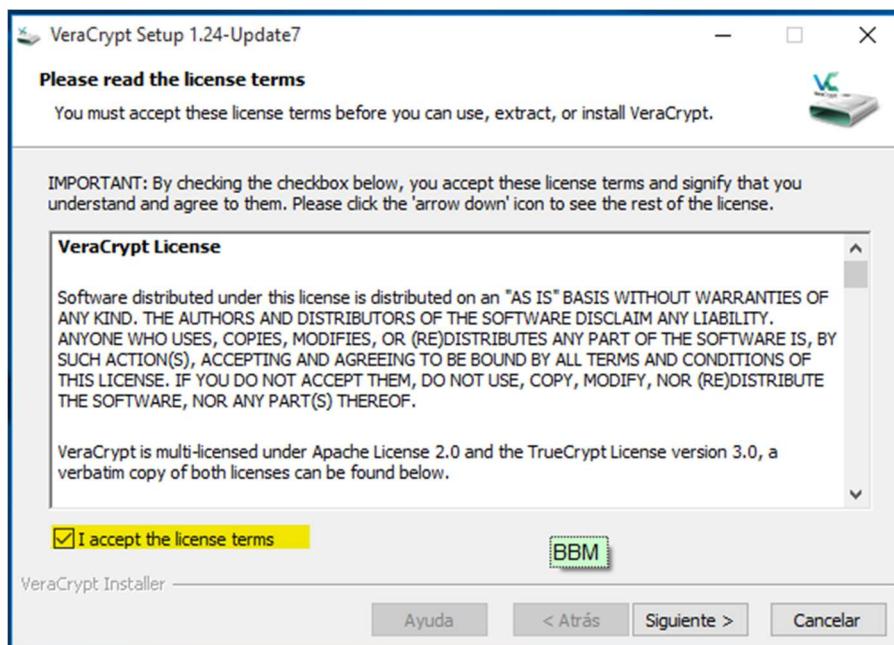
Nos sale la típica pantalla solicitando permisos de administrador. Le decimos que Sí.



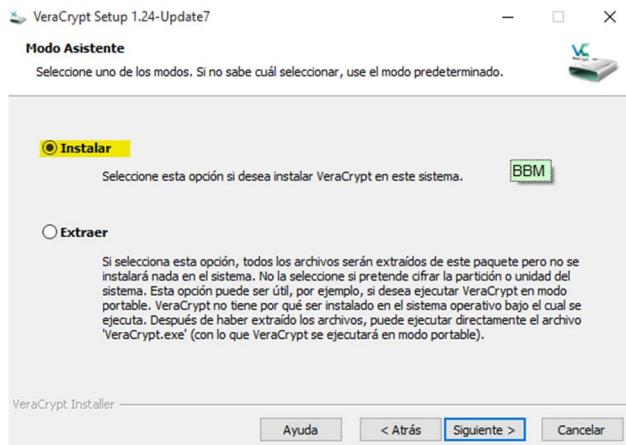
La instalación es como la típica que tienen todos los programas. En la primera ventana elegimos idioma (**español** en mi caso):



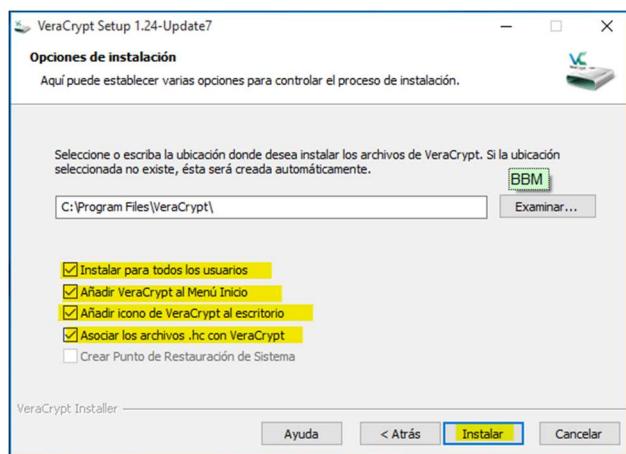
Para la segunda ventana nos leemos los términos de licencia y hacemos **check** en I accept the license terms.



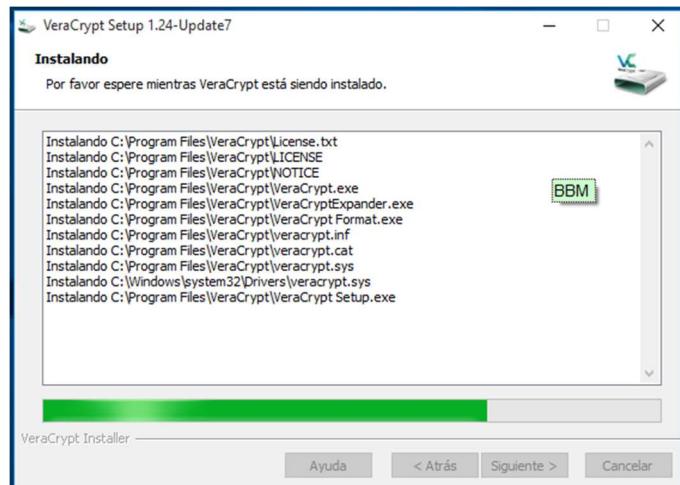
En la siguiente dejamos **Instalar** por defecto, pues si vamos a cifrar el sistema completo o la partición donde está el sistema operativo, se debe instalar obligatoriamente en el PC y no usarlo en modo *portable*.



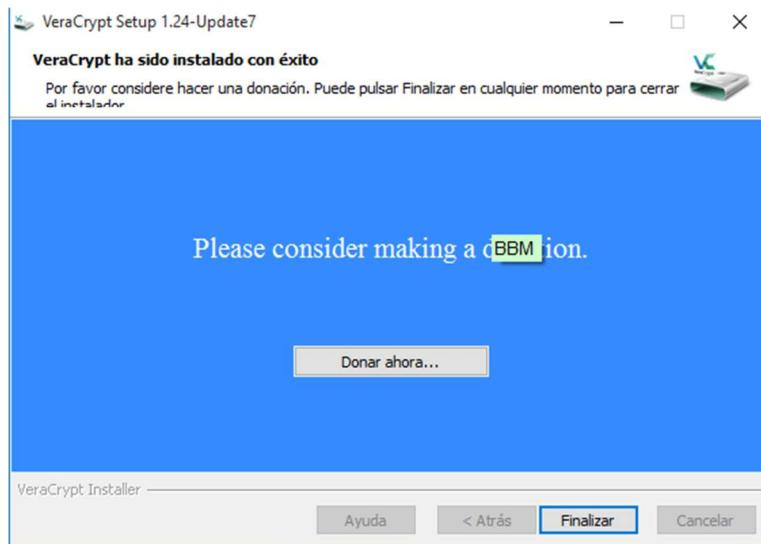
En la siguiente ventana elegimos la ruta y dejamos los **cuatro primeros check** marcados. Clic en **Instalar**.



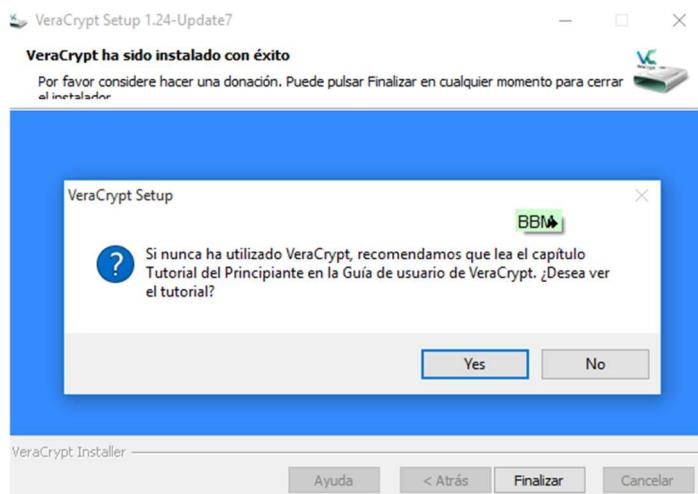
Esperamos que cargue y nos saldrá un mensaje indicando que se ha instalado con éxito.



Al darle a **OK**, nos sale una ventana para **Donar una cantidad monetaria**. Clic en **Finalizar**.



Nos salta un *pop-up* conforme nos indica que al no haberlo usado nunca deberíamos ver un tutorial. Clic en **Yes/No** en función de lo que se pretende.



Como podemos observar, nos sale una pantalla como esta, situada en la página: <https://veracrypt.fr/en/Beginner's%20Tutorial.html>

veracrypt.fr/en/Beginner%27s%20Tutorial.html

Aplicaciones Python BBMASAX/BBM: Re... [GitHub] Please veri... PE PCAP – Programmi... Aruba Central: Mod... Lista de lectura

VeraCrypt

Home Source Code Downloads Documentation Donate Forums

Documentation > Beginner's Tutorial

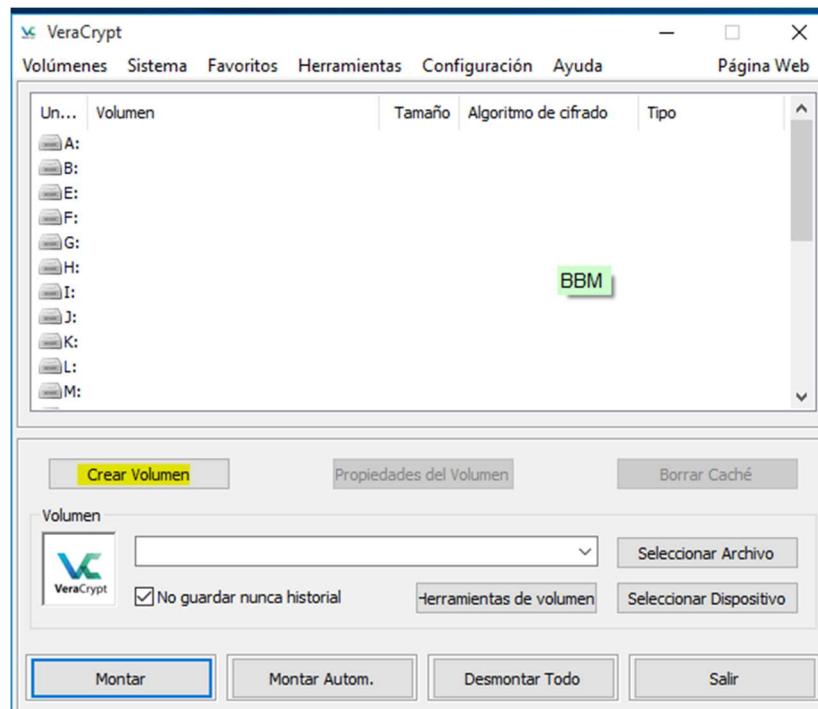
Beginner's Tutorial

How to Create and Use a VeraCrypt Container

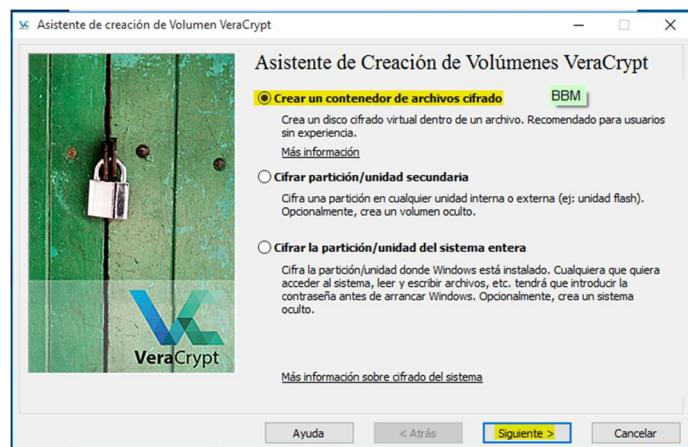
This chapter contains step-by-step instructions on how to create, mount, and use a VeraCrypt volume. We strongly recommend that you also read the other sections of this manual, as they contain important information.

Crear un contenedor de archivos cifrado

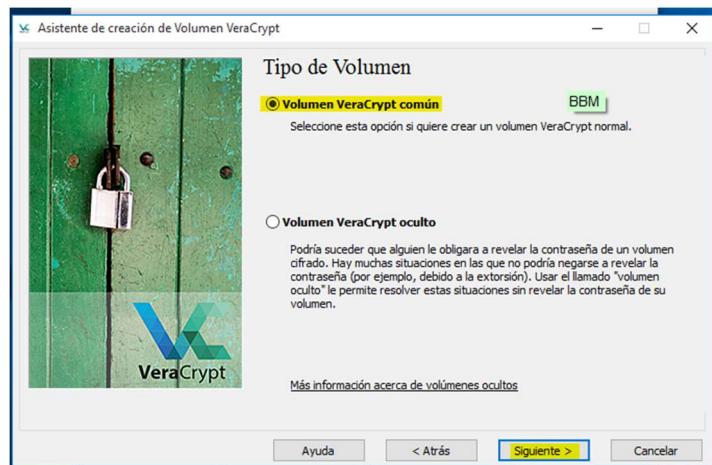
Para probar un poco el funcionamiento del programa, he seguido el **tutorial para principiantes** que indica. Hacemos doble clic en el icono  VeraCrypt. Nos aparecerá la ventana principal de VeraCrypt. En ella, hacemos clic en **Crear Volumen**.



Nos aparece el **Asistente de Creación de Volúmenes VeraCrypt**. En este caso, se elige qué tipo de volumen VeraCrypt se va a crear. Un volumen de este tipo puede residir en un **archivo** llamado **contenedor** (el que voy a usar en el ejemplo), en una **partición** o en un **disco**. Para este ejemplo, y al ser el del tutorial para principiantes, voy a elegir **Crear un contenedor de archivos cifrado**.



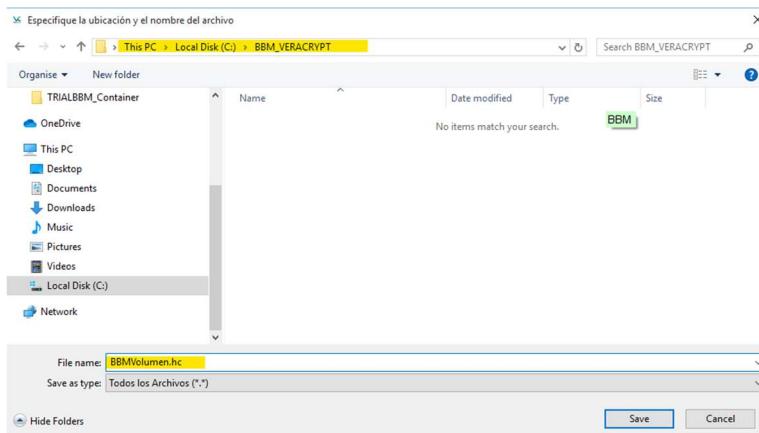
En este paso se elige si crear un volumen común u oculto. Dejamos la opción por defecto común.



En este paso, hay que especificar donde va a estar el **volumen de VeraCrypt** creado. Como es un archivo normal con extensión .hc, le damos a **Seleccionar Archivo** para ubicarlo.



En mi caso lo pongo en la ruta **C:\BBM_VERACRYPT**, y como nombre **BBMVolumen.hc**.

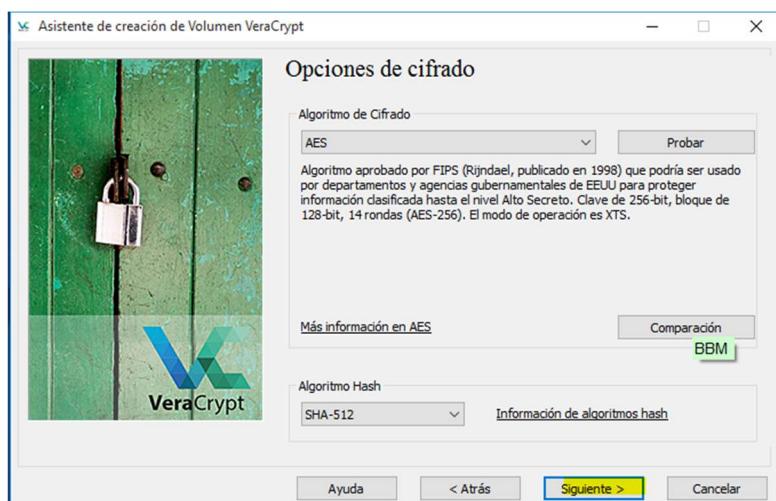


NOTA: Aquí sólo estamos creando el archivo, no encriptando nada.

Queda de la siguiente manera. Hacemos clic en **Siguiente**.



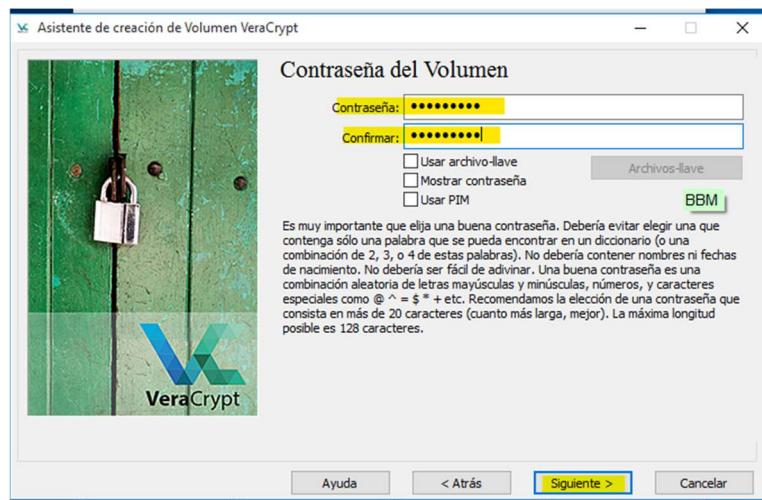
Aquí se pueden usar diferentes algoritmos de encriptación y *hashing*. Para este ejemplo, lo dejamos por defecto. Clic en **Siguiente**.



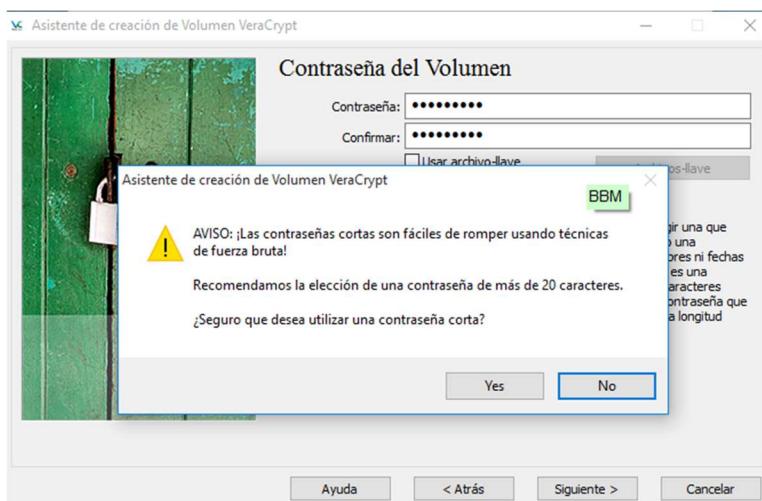
En esta ventana se especifica el tamaño del contenedor. Yo especifiqué 250 MB (aunque se le puede poner lo que se quiera mientras no supere los GB del disco). Clic en **Siguiente**.



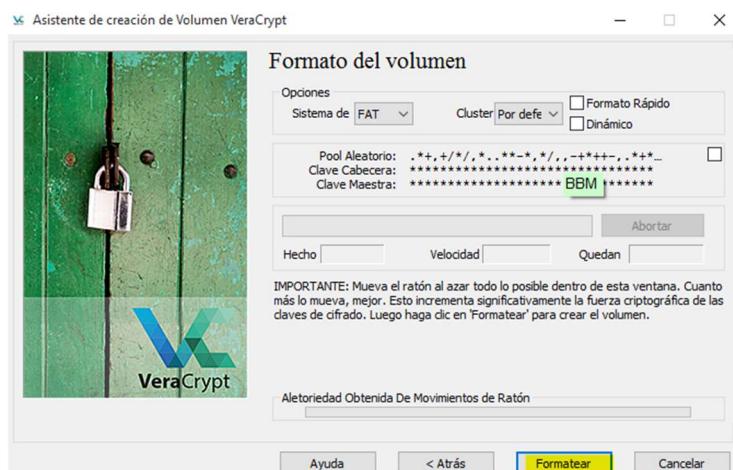
Aquí ponemos la contraseña. Debe ser siempre una contraseña fuerte



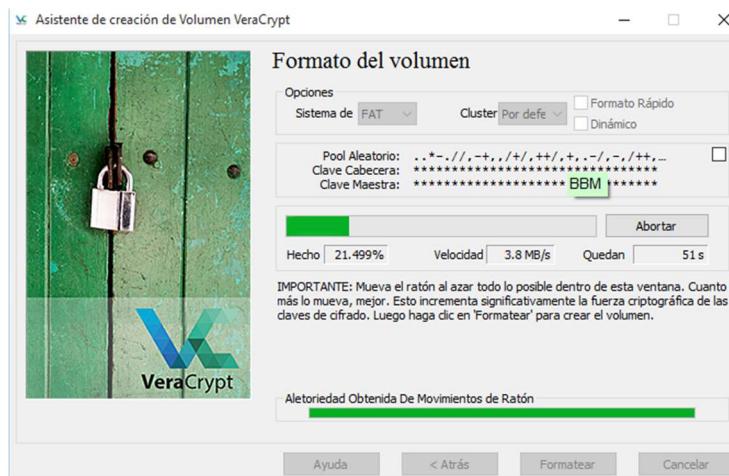
NOTA: Como en mi caso he puesto **Abcd1234**, me da un aviso de que la contraseña es corta, y me pregunta si la quiero usar. Le digo que **si**



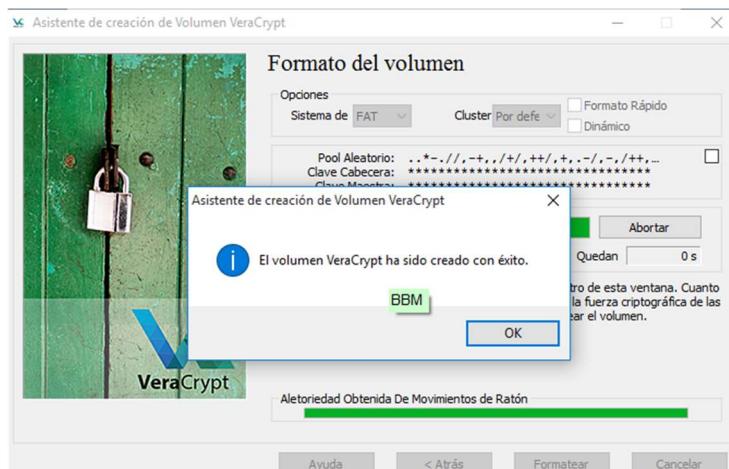
En esta ventana, hay que mover el ratón de la forma **más aleatoria posible** dentro de la ventana del **Asistente de creación de volumen** al menos hasta que el indicador de aleatoriedad se vuelva verde. Cuanto más tiempo mueva el ratón mejor, porque esto aumenta significativamente la fuerza criptográfica de las claves de cifrado (lo que aumenta la seguridad).



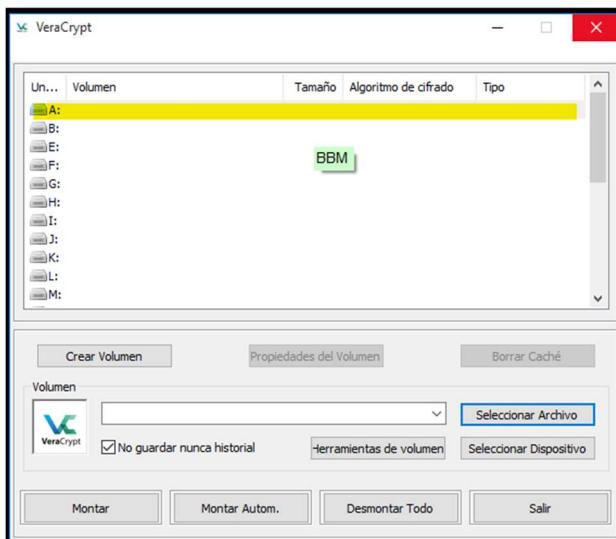
Una vez acaba y se llena la barra, hacemos clic en **Formatear**.



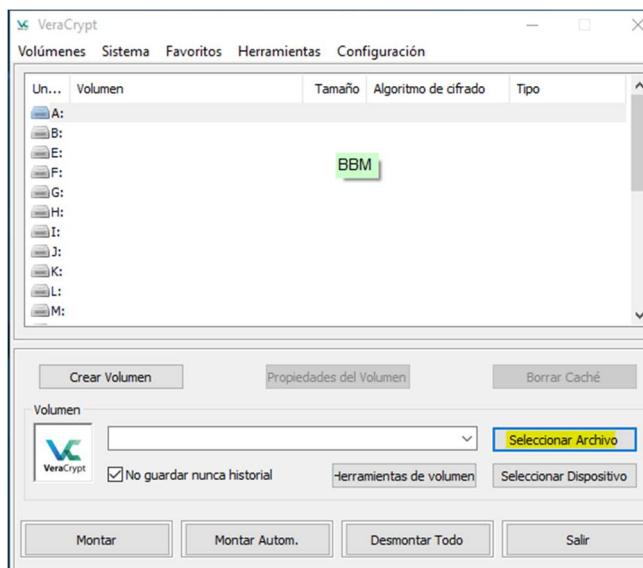
Cuando acaba, nos indica que hemos hecho con éxito la instalación. Le damos a **Exit** y en la siguiente ventana a **Ok**.



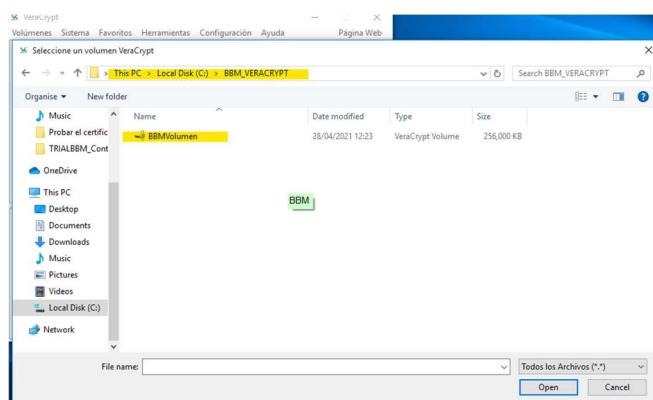
Ahora toca montar el volumen que acabamos de crear. En la ventana principal seleccionamos una de las letras donde el **contenedor será montado**.



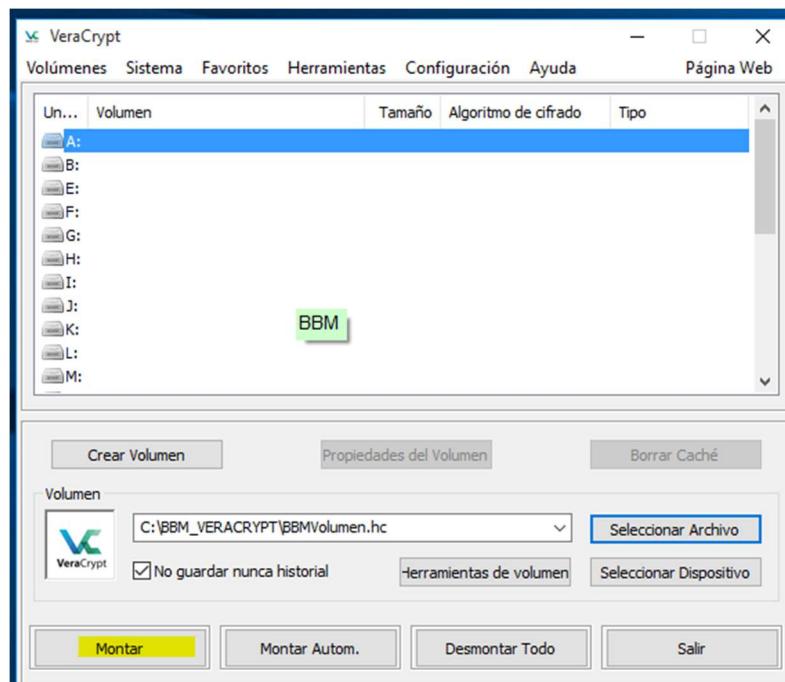
Hacemos clic en **Seleccionar Archivo**.



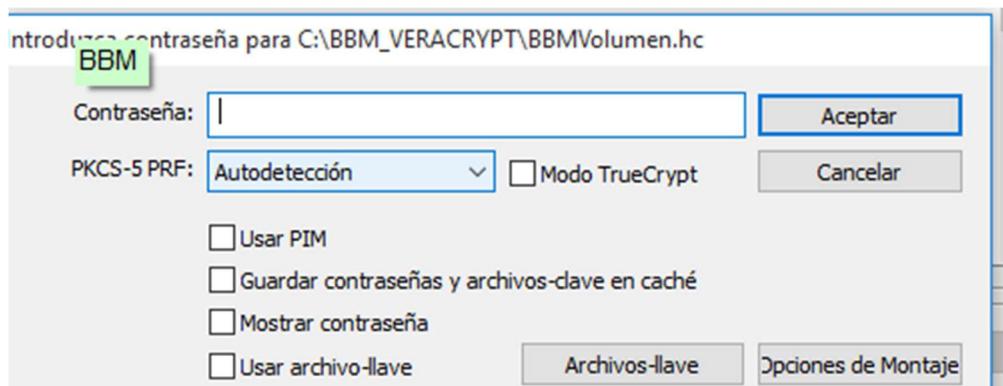
Vamos a la ruta donde instalamos el archivo .hc (en mi caso C:\BBM_VERACRYPT)



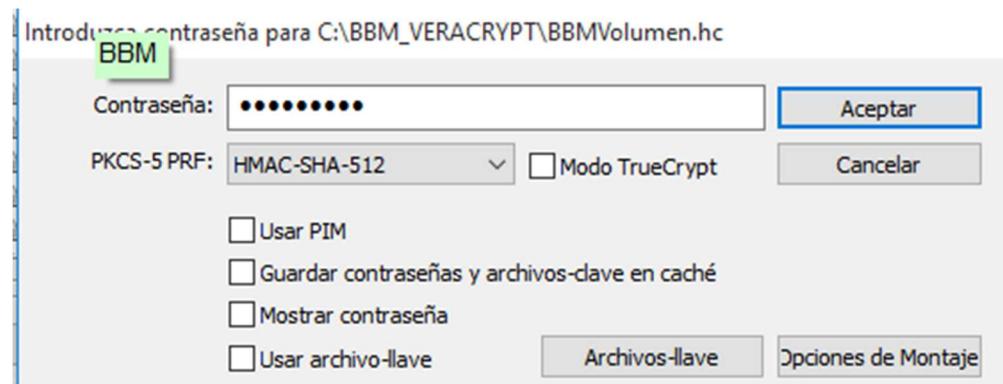
Hacemos clic en Montar.



Nos salta una ventana pidiendo una **contraseña** (que es la que introdujimos antes Abcd1234.)

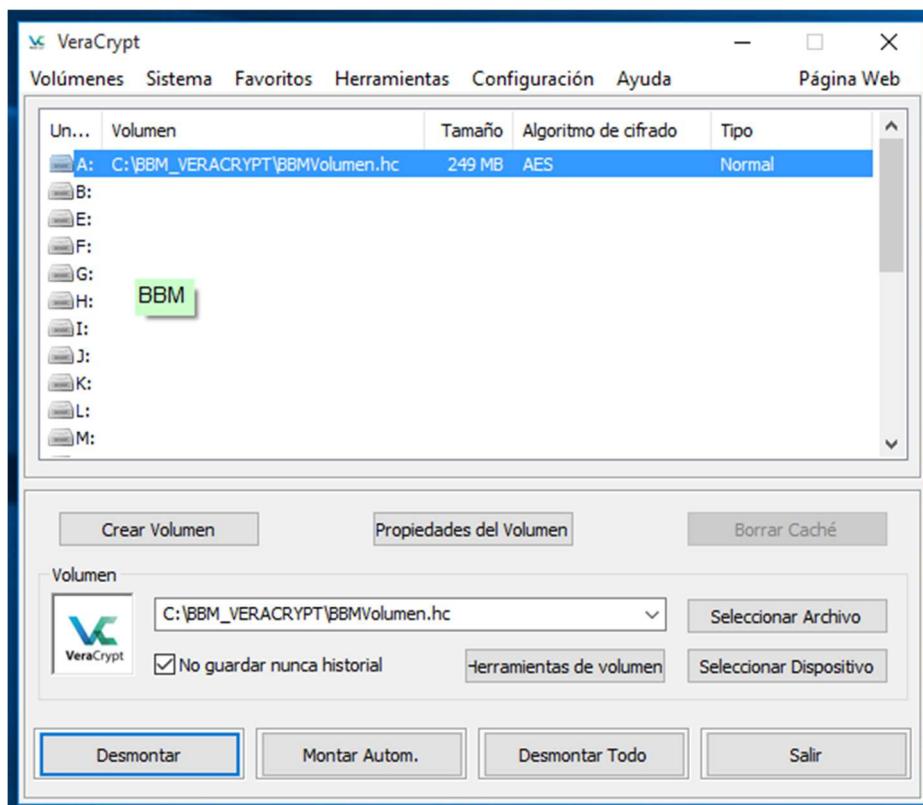


Seleccionamos el algoritmo utilizado durante la creación del volumen (si es el por defecto, será **HMAC-SHA-512**). Le damos a OK.



Se acaba de montar con éxito el contenedor como un disco virtual A: El disco virtual está completamente encriptado (incluidos los nombres de los archivos, las tablas de asignación, el espacio libre, etc.) y se comporta como un disco real. Se pueden guardar (o copiar, mover, etc.) archivos en este disco virtual y se cifrarán sobre la marcha a medida que se escriben.

Si abre un archivo almacenado en un volumen VeraCrypt, por ejemplo, en el reproductor multimedia, el archivo se descifrará automáticamente a la RAM (memoria) sobre la marcha mientras se lee.



NOTA: Cuando se abre un archivo almacenado en un volumen de VeraCrypt (o cuando escribe / copia un archivo a / desde el volumen de VeraCrypt) no se le pide la contraseña nuevamente. Solo se debe ingresar la contraseña correcta cuando monte el volumen.

Encriptación de Columnas de Base de datos

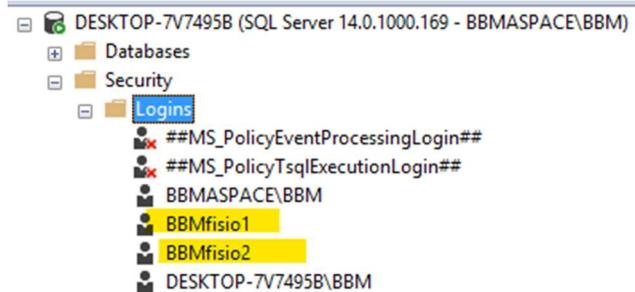
Este escenario es muy similar al del ejemplo de clase, pues mi proyecto versa sobre algo de tipo médico. Lo primero que hacemos es crear los usuarios a nivel de servidor con el procedimiento almacenado del sistema `sp_addlogin <nombre>, <contraseña>`:

```
USE master
GO

-- Creamos login con el procedimiento almacenado sp_addlogin a nivel
server

sp_addlogin 'BBMfisio1', 'Abcd1234.'
sp_addlogin 'BBMfisio2', 'Abcd1234.'
```

NOTA: los vemos almacenados en Security>Logins



Ahora creamos los logins a nivel de usuario. Para esto usamos el comando `CREATE USER <usuario> FOR LOGIN <usuario_nivel_server>`:

```
USE BBM_ASPACE
go

BBM_ASPACE
+ Database Diagrams
+ Tables
+ Views
+ External Resources
+ Synonyms
+ Programmability
+ Service Broker
+ Storage
+ Security
    + Users
        + BBMFIS01
        + BBMFIS02
```

`DROP USER IF EXISTS BBMFIS01
GO`

`DROP USER IF EXISTS BBMFIS02
GO`

`CREATE USER BBMFIS01 FOR LOGIN BBMfisio1 --A NIVEL
BASE DE DATOS
go`

`CREATE USER BBMFIS02 /*USUARIO BASE DE DATOS*/ FOR
LOGIN BBMfisio2 /*LOGIN DE SERVER*/
go`

Ahora damos permisos sobre la tabla `trial.BBM_Usuario`, sobre la cual vamos a trabajar:

```
SELECT * FROM trial.BBM_Usuario
GO
```

	Usuario_ID	Nombre_1	Nombre_2	Apellido_1	Apellido_2	DNI	Otros_Detalles
1	NO CUMPLE	RICHI	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
2	USERGPP012	PACO		GONZALEZ	PEREZ	11111012S	
3	USERTEST	NOMBRETEST	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
4	USERVNA013	ANA	MARIA	VAZQUEZ	NUNEZ	11111013N	

```
-- Grant access to the table to both doctors. CONCEDEMOS EL HECHO DE QUE  
CONSULTEN E INSERTEN
```

```
GRANT SELECT, INSERT ON trial.BBM_Usuario TO BBMFIS01;  
GRANT SELECT, INSERT ON trial.BBM_Usuario TO BBMFIS02;  
GO
```

Para encriptar, hay dos pasos que son necesarios:

- Creación de **Master Key**. Es una clave simétrica que se usa para proteger las claves privadas de certificados y las claves asimétricas presentes en la base de datos. Al crearla, la clave maestra se cifra mediante el algoritmo AES_256 y una contraseña proporcionada por el usuario.

```
-- Create the Master Key
```

```
DROP MASTER KEY  
GO
```

```
CREATE master KEY encryption BY password = 'Abcd1234.' --> CREAMOS UNA  
CLAVE MAESTRA PROTEGIDA POR ESA PALABRA. HAY QUE TENER CUIDADO, PORQUE  
YA HAY UNA MASTER KEY EN MASTER. LO QUE SIGNIFICA QUE TENEMOS QUE ESTAR  
EN LA TABLA
```

```
GO
```

Si voy a **base de datos>security>symmetric keys**, en el **GUI** no se ve, y siempre tenemos la duda, este comando me lo resuelve. Por lo tanto, siempre tengo una **master key** en la base de datos, y una copia en master

The screenshot shows the SSMS Object Explorer on the left and the Object Explorer pane on the right. In the Object Explorer, a tree view shows the database structure of 'BBM_ASPACE'. Under the 'Security' node, the 'Symmetric Keys' node is highlighted with a yellow box. In the Object Explorer pane, the 'master' database is selected, and its security structure is shown, including the 'Symmetric Keys' node which also has a yellow box around it.

```
SELECT name KeyName,  
       symmetric_key_id KeyID,  
       key_length KeyLength,  
       algorithm_desc KeyAlgorithm  
FROM sys.symmetric_keys;  
GO
```

	KeyName	KeyID	KeyLength	KeyAlgorithm
1	##MS_DatabaseMasterKey##	101	256	AES_256

- Creación de **Certificado auto firma**. Es un elemento protegible de nivel de base de datos que sigue el estándar X.509 y admite los campos V1 de X.509. **CREATE CERTIFICATE** puede cargar un certificado desde un archivo, una constante binaria o un ensamblado. Esta instrucción también puede generar un par de claves y crear un certificado con firma personal. En este caso, creamos dos, uno por fisioterapeuta.

BBM_ASPACE

- Database Diagrams
- Tables
- Views
- External Resources
- Synonyms
- Programmability
- Service Broker
- Storage
- Security
- Users
- Roles
- Schemas
- Asymmetric Keys
- Certificates

```
-- Create a self-signed certificate. SE CREAN 2 CERTIFICADOS (UTILIZA PARA PROTEGERSE LA MASTER KEY) UNA PARA CADA USER. DEBEN HACERSE CON FECHA DE CADUCIDAD, PARA OBLIGAR A RENOVAR
```

```
CREATE CERTIFICATE BBMFIS01cert AUTHORIZATION
BBMFIS01
WITH subject = 'Abcd1234.', start_date = '01/01/2021'
GO
CREATE CERTIFICATE BBMFIS02cert AUTHORIZATION
BBMFIS02
WITH subject = 'Abcd1234.', start_date = '01/01/2021'
GO
```

Comprobamos que se crearon:

```
SELECT name CertName, -- ESTA SE USA PARA VER LOS CERTIFICADOS QUE TENEMOS
certificate_id CertID,
pvt_key_encryption_type_desc EncryptType,
issuer_name Issuer
FROM sys.certificates;
GO
```

	CertName	CertID	EncryptType	Issuer
1	BBMFIS01cert	256	ENCRYPTED_BY_MASTER_KEY	Abcd1234.
2	BBMFIS02cert	257	ENCRYPTED_BY_MASTER_KEY	Abcd1234.

Como último paso, creo una clave simétrica para cada uno. Comando **CREATE SYMMETRIC KEY <nombre_clave> WITH ALGORITHM = AES_256 ENCRYPTION BY CERTIFICATE <cerificado>**:

```
CREATE SYMMETRIC KEY BBMFIS01key
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE BBMFIS01cert
GO

CREATE SYMMETRIC KEY BBMFIS02key
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE BBMFIS02cert
GO
```

NOTA: Curiosamente estas sí se ven

```
SELECT *
FROM sys.symmetric_keys
GO
```

	name	principal_id	symmetric_key_id	key_length	key_algorithm	algorithm_desc	create_date	modify_date	key_guid
1	##MS_DatabaseMasterKey##	1	101	256	A3	AES_256	2021-04-29 10:35:52.020	2021-04-29 10:35:52.020	CCAD6000-4CE4-40CF-8285-299A27D196D9
2	BBMFIS01key	1	256	256	A3	AES_256	2021-04-29 10:38:51.587	2021-04-29 10:38:51.587	8D939E00-F7C5-4849-A450-2375B24B8AEC
3	BBMFIS02key	1	257	256	A3	AES_256	2021-04-29 10:38:51.590	2021-04-29 10:38:51.590	745E9A00-B706-4FD8-9B07-C1DAE54FE32C

En esta parte tuve un **problema**. Tuve que borrar toda la tabla porque me faltaba que los tipos a encriptar fuesen se tipo **varbinary**:

```
--IMPERSONO
EXECUTE AS User = 'BBMFIS01'

PRINT USER
GO
-- BBMFIS01

--ABRO LA CLAVE PARA PODER INSERTAR
OPEN SYMMETRIC KEY BBMFIS01key
    DECRYPTION BY CERTIFICATE BBMFIS01cert
GO

INSERT INTO [trial].[BBM_Usuario]
VALUES
    ('KEYFISO', 'USUARIO', 'INSERTADO', 'POR', 'BBMFIS01',
     Encryptbykey(Key_guid('BBMFIS01Key'), '12345678A'),
     Encryptbykey(Key_guid('BBMFIS01Key'), 'TEST'))
GO

Msg 8152, Level 16, State 8, Line 242
String or binary data would be truncated.
The statement has been terminated.

REVERT
```

Borro la tabla y las constraints:

```
DROP TABLE IF EXISTS trial.BBM_Usuario
GO
Msg 3726, Level 16, State 1, Line 254
Could not drop object 'trial.BBM_Usuario' because it is referenced by a FOREIGN KEY constraint.

ALTER TABLE trial.BBM_Expediente DROP CONSTRAINT
BBM_Usuario_Usuario_ID_FK

DROP TABLE IF EXISTS trial.BBM_Usuario
GO
```

Creamos la tabla con los valores **varbinary**.

```
/*CREAMOS TABLA CON VALORES BUENOS*/
CREATE TABLE [trial].[BBM_Usuario]
(
    [Usuario_ID] [INTEGER],
    [Nombre_1] [varchar](20),
    [Nombre_2] [varchar](20),
    [Apellido_1] [varchar](20),
    [Apellido_2] [varchar](20),
    [DNI] [Varbinary](1000),
    [Otros_Detalles]
    [varbinary](4000),
)
GO
```

	trial.BBM_Usuario
	Columns
	Usuario_ID (int, null)
	Nombre_1 (varchar(20), null)
	Nombre_2 (varchar(20), null)
	Apellido_1 (varchar(20), null)
	Apellido_2 (varchar(20), null)
	DNI (varbinary(1000), null)
	Otros_Detalles (varbinary(4000), null)

Para insertar los valores, me **impersono** en los usuarios, **abro** la clave simétrica, inserto los valores con el comando `Encryptbykey(Key_guid('CLAVE'), 'VALOR_A_INSERTAR')`, **compruebo** lo insertado. Para finalizar, **cierro** la clave y **vuelvo** a **dbo**.

```

/*INSERTAMOS VALORES IMPERSONANDO*/

EXECUTE AS User = 'BBMFIS01'

PRINT USER
GO

--ABRO LA CLAVE PARA PODER INSERTAR
OPEN SYMMETRIC KEY BBMFIS01key
    DECRYPTION BY CERTIFICATE BBMFIS01cert
GO

SELECT *
FROM sys.openkeys
GO

INSERT INTO [trial].[BBM_Usuario]
VALUES
    (1,'USUARIO','INSERTADO','POR','FISIO01',
Encryptbykey(Key_guid('BBMFIS01Key'), '123456789'),
Encryptbykey(Key_guid('BBMFIS01Key'), 'TEXTO OCULTO'))
GO

INSERT INTO [trial].[BBM_Usuario]
VALUES
    (2,'USUARIO','INSERTADO','POR','FISIO01',
Encryptbykey(Key_guid('BBMFIS01Key'), '1234567891'),
Encryptbykey(Key_guid('BBMFIS01Key'), 'TEXTO OCULTO'))
GO

INSERT INTO [trial].[BBM_Usuario]
VALUES
    (3,'USUARIO','INSERTADO','POR','FISIO01',
Encryptbykey(Key_guid('BBMFIS01Key'), '12345674'),
Encryptbykey(Key_guid('BBMFIS01Key'), 'TEXTO OCULTO'))
GO

-- SACO EL LISTADO ENcriptando el CODIGO DEL PACIENTE Y LA ENFERMEDAD
-- QUE SUFRE. VEMOS QUE EST`TODO ENcriptado --> MS ADELANTE, CON BBMFIS01
-- USAMOS EL CERTIFICADO PARA VERLO
SELECT * FROM trial.BBM_Usuario
GO



|   | Results | Messages                                                                                                                              |
|---|---------|---------------------------------------------------------------------------------------------------------------------------------------|
|   |         |                                                                                                                                       |
| 1 | 1       | USUARIO INSERTADO POR FISIO01 0x009E938DC5F74948A4502375B24B8AEC020000001E76EA... 0x009E938DC5F74948A4502375B24B8AEC020000028B2F0A... |
| 2 | 2       | USUARIO INSERTADO POR FISIO01 0x009E938DC5F74948A4502375B24B8AEC020000098815EA... 0x009E938DC5F74948A4502375B24B8AEC02000002AF4EA...  |
| 3 | 3       | USUARIO INSERTADO POR FISIO01 0x009E938DC5F74948A4502375B24B8AEC02000003D90785... 0x009E938DC5F74948A4502375B24B8AEC0200000B7A8B5...  |



-- Close all opened keys Y VUELVO A dbo. SI DEJO ABIERTO TODO CON LAS
-- SESIONES, COMO NO HAY AUTORIZACIM, NO PASAR' NADA
CLOSE ALL symmetric keys
GO

REVERT
GO

PRINT USER

```

```
-- BBMFIS02
EXECUTE AS user = 'BBMFIS02'
GO

PRINT USER
-- ABRO LAS CLAVES
OPEN symmetric KEY BBMFIS02key decryption BY certificate BBMFIS02cert
GO

-- view the list of open keys in the session
SELECT *
FROM sys.openkeys
GO
```

	Results	Messages
1	database_id database_name key_id key_name key_guid opened_date status	12 BBM_ASPACE 257 BBMFIS02key 745E9A00-B706-4FD8-9B07-C1DAE54FE32C 2021-04-30 09:58:08.093 1

```
-- INSERTO LO DEL DOCTOR 2, QUE ES EL QUE TIENE PERMISOS
```

```
INSERT INTO [trial].[BBM_Usuario]
VALUES
(1, 'USUARIO', 'INSERTADO', 'POR', 'FISIO02',
Encryptbykey(Key_guid('BBMFIS02Key')), '123456789'),
Encryptbykey(Key_guid('BBMFIS02Key')), 'TEXTO OCULTO')
GO
```

```
INSERT INTO [trial].[BBM_Usuario]
VALUES
(2, 'USUARIO', 'INSERTADO', 'POR', 'FISIO02',
Encryptbykey(Key_guid('BBMFIS02Key')), '1234567891'),
Encryptbykey(Key_guid('BBMFIS02Key')), 'TEXTO OCULTO')
GO
```

```
INSERT INTO [trial].[BBM_Usuario]
VALUES
(3, 'USUARIO', 'INSERTADO', 'POR', 'FISIO02',
Encryptbykey(Key_guid('BBMFIS02Key')), '12345674'),
Encryptbykey(Key_guid('BBMFIS02Key')), 'TEXTO OCULTO')
GO
```

```
SELECT * FROM trial.BBM_Usuario
WHERE Apellido_2 = 'FISIO02'
```

	Results	Messages
1	Usuario_ID Nombre_1 Nombre_2 Apellido_1 Apellido_2 DNI Otros_Detalles	1 USUARIO INSERTADO POR FISIO02 0x009A5E7406B7D84F9B07C1DAE54FE32C0200000046E7DD... 0x009A5E7406B7D84F9B07C1DAE54FE32C0200000071DE102...
2	2 USUARIO INSERTADO POR FISIO02 0x009A5E7406B7D84F9B07C1DAE54FE32C0200000047C8529... 0x009A5E7406B7D84F9B07C1DAE54FE32C02000000CE50DF...	
3	3 USUARIO INSERTADO POR FISIO02 0x009A5E7406B7D84F9B07C1DAE54FE32C02000000C2BF82B... 0x009A5E7406B7D84F9B07C1DAE54FE32C02000000CF739F8...	

```
--PARA FINALIZAR CIERRO.
CLOSE ALL symmetric keys
GO
```

```
SELECT * FROM trial.BBM_Usuario
GO
```

	Results	Messages
1	Usuario_ID Nombre_1 Nombre_2 Apellido_1 Apellido_2 DNI Otros_Detalles	1 USUARIO INSERTADO POR FISIO01 0x009E938DC5F74948A4502375B24B8AEC020000001E76EAC... 0x009E938DC5F74948A4502375B24B8AEC0200000028B2F0A...
2	2 USUARIO INSERTADO POR FISIO01 0x009E938DC5F74948A4502375B24B8AEC0200000098815EA... 0x009E938DC5F74948A4502375B24B8AEC020000002AF4EAC...	
3	3 USUARIO INSERTADO POR FISIO01 0x009E938DC5F74948A4502375B24B8AEC020000003D90785... 0x009E938DC5F74948A4502375B24B8AEC0200000087A8B5A...	
4	1 USUARIO INSERTADO POR FISIO02 0x009A5E7406B7D84F9B07C1DAE54FE32C02000000F63350D... 0x009A5E7406B7D84F9B07C1DAE54FE32C0200000021CC45...	
5	2 USUARIO INSERTADO POR FISIO02 0x009A5E7406B7D84F9B07C1DAE54FE32C02000000D62EF... 0x009A5E7406B7D84F9B07C1DAE54FE32C0200000057E3FF5...	
6	3 USUARIO INSERTADO POR FISIO02 0x009A5E7406B7D84F9B07C1DAE54FE32C02000000B74FEC4... 0x009A5E7406B7D84F9B07C1DAE54FE32C0200000082AD2...	

Ahora hago las comprobaciones impersonándome en cada fisio y observando que lo que insertó el otro no lo puede ver.

```
-- AHORA DESENCRIPTO PARA VER LAS ENCRIPACIÓNES DE LAS TABLAS
REVERT
GO
print user

-- ME IMPERSONO
EXECUTE AS user = 'BBMFIS01'
GO
OPEN SYMMETRIC KEY BBMFIS01key decryption BY CERTIFICATE BBMFIS01cert
-- SIEMPRE Y CUANDO DISPONGA EL CERTIFICADO SE VER'
GO
```

```
SELECT Usuario_ID, --> COMPROBAMOS LO QUE VE EL DOCTOR 1. LO QUE
EST`ENCRIPACIÓD POR 2 SE VE COMO NULO
Nombre_1,
Apellido_2,
CONVERT(VARCHAR, DecryptByKey(DNI)) AS DNI,
CONVERT (VARCHAR, DecryptByKey(Otros_Detalles)) AS Detalles
FROM trial.BBM_Usuario
GO
```

	Usuario_ID	Nombre_1	Apellido_2	DNI	Detalles
1	1	USUARIO	FISIO01	123456789	TEXTO OCULTO
2	2	USUARIO	FISIO01	1234567891	TEXTO OCULTO
3	3	USUARIO	FISIO01	12345674	TEXTO OCULTO
4	1	USUARIO	FISIO02	NULL	NULL
5	2	USUARIO	FISIO02	NULL	NULL
6	3	USUARIO	FISIO02	NULL	NULL

```
CLOSE ALL SYMMETRIC keys
GO
```

```
-- HACEMOS EL DOCTOR 2
REVERT
GO
```

```
PRINT USER
```

```
EXECUTE AS USER = 'BBMFIS02'
GO
```

```
OPEN SYMMETRIC KEY BBMFIS02key decryption BY CERTIFICATE BBMFIS02cert
GO
```

```

SELECT Usuario_ID,
Nombre_1,
Apellido_2,
CONVERT(VARCHAR, Decryptbykey(DNI)) AS DNI,
CONVERT(VARCHAR, Decryptbykey(Otros_Detalles)) AS Detalles
FROM trial.BBM_Usuario
GO

```

Results Messages

	Usuario_ID	Nombre_1	Apellido_2	DNI	Detalles
1	1	USUARIO	FISIO01	NULL	NULL
2	2	USUARIO	FISIO01	NULL	NULL
3	3	USUARIO	FISIO01	NULL	NULL
4	1	USUARIO	FISIO02	123456789	TEXTO OCULTO
5	2	USUARIO	FISIO02	1234567891	TEXTO OCULTO
6	3	USUARIO	FISIO02	12345674	TEXTO OCULTO

```

-- Null id Patient 1 2 3
CLOSE ALL SYMMETRIC keys
GO

```

```
REVERT
```

```
PRINT USER
```

Para cerrar el ejercicio, me cargo el certificado del FISIO02 y comproibo las consultas

```

DROP SYMMETRIC KEY [BBMFIS02key] -- como est n anidados, primero tengo
que borrar la clave sim rica
GO

```

```

DROP CERTIFICATE [BBMFIS02cert]
GO

```

```

EXECUTE AS USER = 'BBMFIS02'
GO

```

```

OPEN SYMMETRIC KEY BBMFIS02key decryption BY CERTIFICATE BBMFIS02cert
GO

```

Messages

Msg 15151, Level 16, State 1, Line 518
Cannot find the symmetric key 'BBMFIS02key', because it does not exist or you do not have permission.

```

SELECT Usuario_ID,
Nombre_1,
Apellido_2,
CONVERT(VARCHAR, Decryptbykey(DNI)) AS DNI,
CONVERT(VARCHAR, Decryptbykey(Otros_Detalles)) AS Detalles
FROM trial.BBM_Usuario
GO

```

Results Messages

	Usuario_ID	Nombre_1	Apellido_2	DNI	Detalles
1	1	USUARIO	FISIO01	NULL	NULL
2	2	USUARIO	FISIO01	NULL	NULL
3	3	USUARIO	FISIO01	NULL	NULL
4	1	USUARIO	FISIO02	NULL	NULL
5	2	USUARIO	FISIO02	NULL	NULL
6	3	USUARIO	FISIO02	NULL	NULL

```
REVERT
```

```
PRINT USER
```

Encriptación de Backup de Base de datos

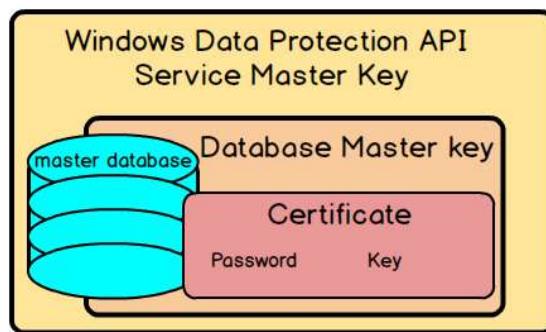
La encriptación de copias de seguridad SQL Server es introducida en SQL Server 2014 y soporta la encriptación de copias de seguridad de base de datos directamente desde el motor de la base de datos. En el caso de la característica de Encriptación de Copia de Seguridad, la encriptación/des encriptación es realizada sólo cuando se hace una copia de seguridad y cuando se restaura una base de datos, por tanto, no hay problemas de desempeño.

SQL Server tiene una infraestructura de encriptación jerárquica donde cada capa en la jerarquía encripta la capa debajo.

La primera capa de jerarquía es la **Clave Maestra de Servicio** (*Service Master Key*, SMK). La Clave Maestra de Servicio es generada automáticamente durante la instalación de SQL Server y almacenada en la base de datos maestra del sistema. La SMK es única para cada instancia SQL Server. La Clave Maestra de Servicios es encriptada basándose en las credenciales para la cuenta de servicio SQL Server y la clave Windows Data Protection API (DPAPI).

La siguiente capa es una **Clave Maestra de Base de Datos** (*Database Master Key*, DMK) de a base de datos maestra. La Clave Maestra de Base de Datos es única para cada base de datos maestra del sistema para cada instancia SQL Server. La Clave Maestra de Base de Datos es encriptada usando la Clave Maestra de Servicio.

El siguiente nivel en la jerarquía es un **certificado** que puede contener una clave privada que es protegido por la Clave Maestra de Base de Datos, o una clave asimétrica (si se usa una clave asimétrica para encriptar los datos de respaldo, sólo claves asimétricas que residen en el proveedor Administración Extensible de Claves (EKM) son soportadas).



La característica de encriptación de copias de seguridad de SQL Server provee encriptación de datos con los algoritmos AES 128, AES 192, AES 256 y Triple DES (3DES).

El **Estándar de Cifrado de Datos**, también conocido como Algoritmo de Cifrado de Datos (DEA) es desarrollado en los setenta y publicado en 1977. El Estándar de Cifrado de Datos es un bloque de cifrado que encripta datos en bloques de 64 bits. DES es un algoritmo simétrico, lo que significa que el mismo algoritmo y clave son usados para el cifrado y el descifrado. Un tamaño de clave de 56-bits. Una máquina de ataque de fuerza bruta DES puede encontrar una clave en aproximadamente 3.5 horas. Y con el tiempo DES sólo se volverá menos seguro.

Para mejorar la seguridad hay variantes de DES como Triple DES (3DES). Triple DES es una manera de usar el Estándar de Cifrado de Datos tres veces, y tiene una longitud de clave de 168 bits, pero también ha demostrado ser inefectivo contra ataques de fuerza bruta.

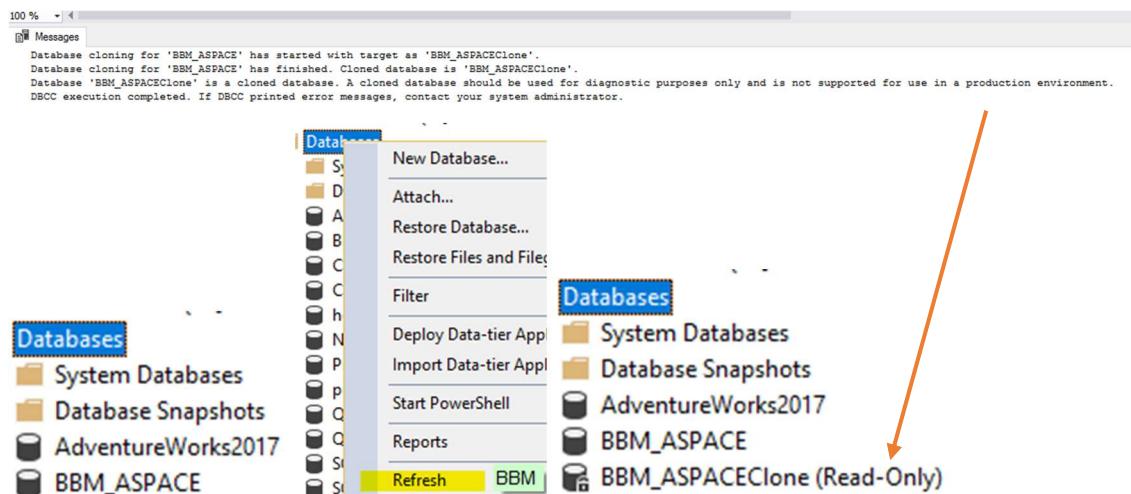
El **Estándar de Cifrado Avanzado** (*Advanced Encryption Standard*, AES), también conocido como Rijndael, el cual es su nombre original, es una especificación establecida en 2001 para el cifrado de datos electrónicos. Consiste en tres bloques de cifrado, AES 128, AES 192 y **AES 256** y cada bloque de cifrado encripta datos en bloques de 128 bits. Una longitud de clave es 128, 192 y 256 bits, respectivamente. AES es, como DES, también un algoritmo simétrico.

El Estándar de Cifrado Avanzado es usado mundialmente y adoptado por el gobierno de los Estados Unidos y desplaza al Estándar de Cifrado de Datos (DES) en términos de seguridad.

La mejor opción es usar cifrado **AES 256** dado que usar un cifrado más fuerte requiere más poder de CPU para cifrar los datos y también más poder de CPU para descifrarlos en el caso de un intento de romper la encriptación.

Primero hacemos un backup de la base de datos original y un clonado de la base de datos:

```
DBCC CLONEDATABASE ('BBM_ASPACE', 'BBM_ASPACEClone')
```



Creamos la **Master Key** (password **Abcd1234.**) y el certificado:

```
USE MASTER;
GO
-- create master key and certificate in database master
-- CREAMOS LA MASTER KEY.
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Abcd1234.';
GO
```

The screenshot shows the 'Messages' tab with the following error message:

```
Msg 15578, Level 16, State 1, Line 29
There is already a master key in the database. Please drop it before performing this statement.
```

```

Databases
  System Databases
    master
      Tables
      Views
      Synonyms
      Programmability
      Service Broker
      Storage
      Security
        Users
        Roles
        Schemas
        Asymmetric Keys
        Certificates BBM
          ##MS_AgentSigningCertificate#
          ##MS_PolicySigningCertificate#
          ##MS_SchemaSigningCertificate#
          ##MS_SmcExtendedSigningCerti#
          ##MS_SQLAuthenticatorCertific#
          ##MS_SQLReplicationSignin#
          ##MS_SQLResourceSignin#
          SQL_criptar_BBMDBcert

```

```

DROP CERTIFICATE SQL_criptar_BBMDBCert
GO
CREATE CERTIFICATE SQL_criptar_BBMDBCert --> SON
CERTIFICADOS SELF-SIGN. Es posible hacerlo en producci n
WITH SUBJECT = 'SQL_criptar_BBMDB Backup Certificate';
GO

-- export the backup certificate to a file --> Es una
recomendaci n Porque cuando pierdes un certificado del
backup, ser  recuperable

```

Creamos la carpeta **BBMBACKCERT** donde se incorporar n los backups de los certificados y la base de datos:

```

/*****CREAMOS LA CARPETA BBMBACKCERT****/
BACKUP CERTIFICATE SQL_criptar_BBMDBCert
TO FILE = 'c:\BBMBACKCERT\SQL_criptar_BBMDBCert.cert'
WITH PRIVATE KEY (
    FILE = 'c:\BBMBACKCERT\SQL_criptar_BBMDBCert.key', -->
nuestra master key creada anteriormente
    ENCRYPTION BY PASSWORD = 'Abcd1234.')
GO

--MI OBJETIVO ES GUARDAR LOS backup the database with encryption. LA
IDEA ES PROTEGERLA ENRIPTANDO EL BACKUP
BACKUP DATABASE BBM_ASPACE
TO DISK = 'c:\BBMBACKCERT\BBM_ASPACE.bak'
WITH ENCRYPTION (ALGORITHM = AES_256, SERVER CERTIFICATE =
SQL_criptar_BBMDBcert) --> CON ESTO ME HACE EL CERTIFICADO
GO


```

Messages

```

Processed 456 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_PRINCIPAL' on file 1.
Processed 112 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_SECONDARY' on file 1.
Processed 2 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_log' on file 1.
BACKUP DATABASE successfully processed 570 pages in 0.714 seconds (6.231 MB/sec).

```

Una prueba ser  cargar  la base de datos y hacer el restore con el certificado. El final ser  aqu . Primero vamos a intentar **cargarnos el .mdf de la base de datos** situada en **C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA**. Ponemos offline la base de datos con el comando:

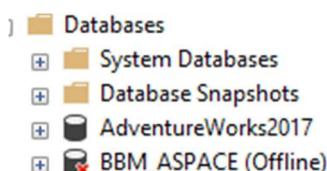
```

ALTER DATABASE BBM_ASPACE SET OFFLINE WITH ROLLBACK IMMEDIATE --> PONGO
LA BASE DE DATOS OFFLINE ECHANDO A QUIEN EST  CONECTADO
go

```

```
-- Failed to restart the current database. The current database is
switched to master.
```

Si hacemos refresh, vemos que nuestra base tiene como una x y pone offline. Si hago clic derecho en la base de datos>**task** no me deja hacer **take offline**. Eso tambi n es un indicativo



Me cargo el archivo .mdf:

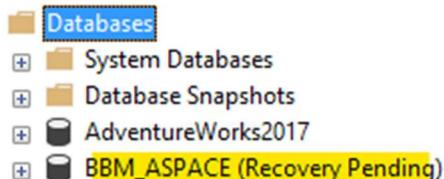
	BBM_ASPACE	03/05/2021 20:28	SQL Server Database Transaction Log File	10,240 KB
	BBM_ASPACE	03/05/2021 20:28	SQL Server Database Primary Data File	10,240 KB
	BBM_ASPACE		Transaction Log File	8,192 KB
	BBM_ASPACE		Data File	8,192 KB
	BBM_ASPACE		Primary Data File	10,240 KB
	BBM_ASPACE		Secondary Data File	8,192 KB
	BBM_ASPACE		Transaction Log File	10,240 KB
	BBM_ASPACE_136225236	03/05/2021 20:22	SQL Server Database Transaction Log File	8,192 KB
	BBM_ASPACE_3432010576	03/05/2021 20:22	SQL Server Database Primary Data File	8,192 KB
	BBM_ASPACE_SECONDARY	13/03/2021 12:17	SQL Server Database Secondary Data File	10,240 KB
	BBM_ASPACE_SECONDARY_3127518758	03/05/2021 20:22	SQL Server Database Secondary Data File	8,192 KB

Da un error, pero si le damos a refresh, se pone en estado **recovery pending**. Si miramos con botón derecho en task, me deja take offline, con lo que está online:

```
ALTER DATABASE BBM_ASPACE SET ONLINE WITH ROLLBACK IMMEDIATE
go
```

```
Msg 5120, Level 16, State 101, Line 91
Unable to open the physical file "C:\DATABASE_BBMASPACE\BBM_ASPACE.mdf". Operating system error 2: "2(The system cannot find the file specified.)".
Msg 5181, Level 16, State 5, Line 91
Could not restart database "BBM_ASPACE". Reverting to the previous status.
Msg 5069, Level 16, State 1, Line 91
ALTER DATABASE statement failed.
```

```
--GUI BBM_ASPACE RECOVERY PENDING
```



```
-- BUT LOOKING TO GUI IS ONLINE
```

Hago también un **backup del log** que guardo en la carpeta BBMBACKCERT:

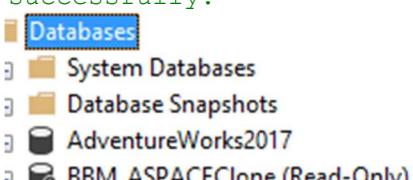
```
/*HAGO UN BACKUP DEL LOG PARA TEST*/
USE master;
GO
-- attempt to take TailLogDB online
BACKUP LOG BBM_ASPACE
TO DISK = 'c:\BBMBACKCERT\BBM_ASPACETailLogDB.log'
WITH CONTINUE_AFTER_ERROR, ENCRYPTION (ALGORITHM = AES_256, SERVER
CERTIFICATE = SQL_encryptar_BBMDCCert)
go

Processed 31 pages for database 'SQL_encryptar_BBMM', file 'SQL_encryptar_BBMM_log' on file 1.
BACKUP LOG successfully processed 31 pages in 0.108 seconds (2.192 MB/sec).
```

Borramos todo para probar:

```
DROP DATABASE BBM_ASPACE; --> REFRESCO DESPUÉS DE EJECUTAR
GO

-- Commands completed successfully.


/****PARA DARLE EMOCIM, ME CARGO TODO****/
USE master
GO
-- RECREATE master key and certificate

DROP CERTIFICATE SQL_encriptar_BBMDBCert
GO
DROP SYMMETRIC KEY SQL_encriptar_BBMDBCert --> a VECES FALLA
GO
DROP MASTER KEY
GO
```

Creo la **master key** y restauramos el **certificado** creándolo desde el backup:

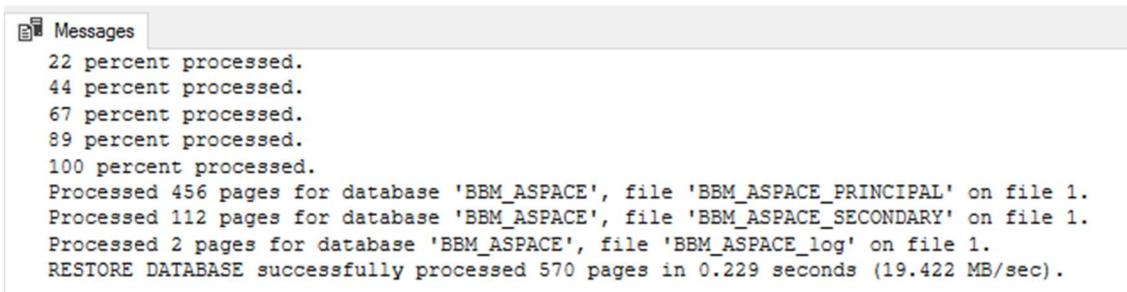
```
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'Abcd1234.';
GO

-- restore the certificate --> RESTAURAMOS EL CERTIFICADO
CREATE CERTIFICATE SQL_encriptar_BBMDBCert
FROM FILE = 'C:\BBMBACKCERT\SQL_encriptar_BBMDBCert.cert'
WITH PRIVATE KEY (FILE = 'C:\BBMBACKCERT\SQL_encriptar_BBMDBCert.key',
DECRYPTION BY PASSWORD = 'Abcd1234.');
GO

-- Commands completed successfully.
```

Restauramos la **base de datos** y el **log**:

```
RESTORE DATABASE BBM_ASPACE
FROM DISK = 'c:\BBMBACKCERT\BBM_ASPACE.bak'
WITH NORECOVERY,
MOVE 'BBM_ASPACE_PRINCIPAL' TO 'c:\BBMBACKCERT\BBM_ASPACE_Data.mdf',
MOVE 'BBM_ASPACE_SECONDARY' TO 'c:\BBMBACKCERT\BBM_ASPACE_Data.ndf',
MOVE 'BBM_ASPACE_log' TO 'c:\BBMBACKCERT\BBM_ASPACE_Log.ldf',
REPLACE, STATS = 10;
GO
/*A VECES DICE QUE NO TIENES PERMISO. VAMOS A LA CARPETA Y LE DAMOS
CONTROL TOTAL SOBRE ELLA*/
```



```
Messages
22 percent processed.
44 percent processed.
67 percent processed.
89 percent processed.
100 percent processed.
Processed 456 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_PRINCIPAL' on file 1.
Processed 112 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_SECONDARY' on file 1.
Processed 2 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_log' on file 1.
RESTORE DATABASE successfully processed 570 pages in 0.229 seconds (19.422 MB/sec).
```

```
/*LANZAMOS EL RESTORE DEL LOG*/
```

```
RESTORE LOG BBM_ASPACE
FROM DISK = 'c:\BBMBACKCERT\BBM_ASPACE\BBM_ASPACELogDB.log';
GO
```

Messages

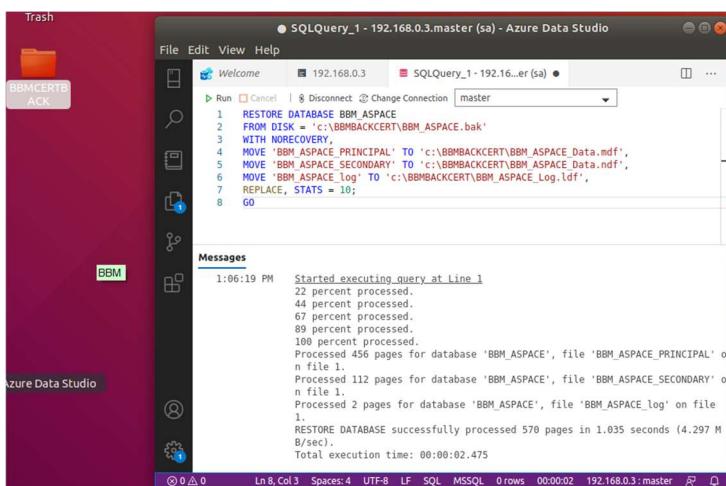
```
Processed 0 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_PRINCIPAL' on file 1.
Processed 0 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_SECONDARY' on file 1.
Processed 4 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_log' on file 1.
RESTORE LOG successfully processed 4 pages in 0.003 seconds (8.300 MB/sec).
```

Hacemos una comprobación de una tabla:

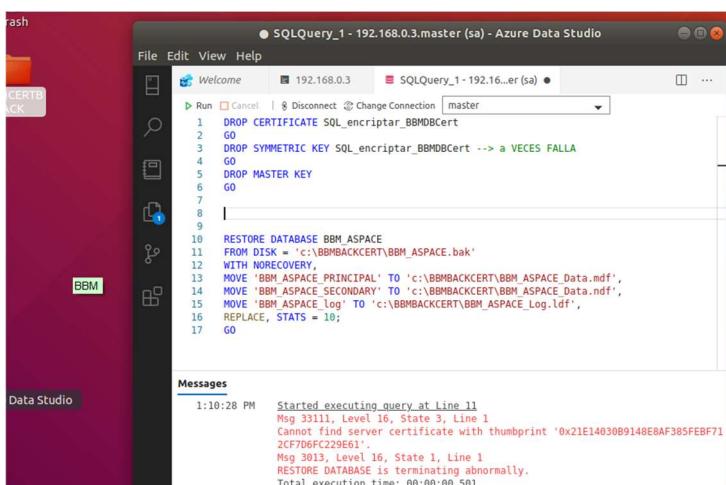
```
USE BBM_ASPACE
GO
SELECT * FROM trial.BBM_Usuario;
GO
```

	Usuario_ID	Nombre_1	Nombre_2	Apellido_1	Apellido_2	DNI	Otros_Detalles
1	NO CUMPLE	RICHI	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
2	USERGPO12	PACO		GONZALEZ	PEREZ	111110125	
3	USERTEST	NOMBRETEST	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
4	USERVNA013	ANA	MARIA	VAZQUEZ	NUNEZ	11111013N	

Comprobamos desde cliente Ubuntu. Nos conectamos a la instancia y probamos recuperación con certificado. Vemos que funciona:



Comprobamos SIN CERTIFICADO y vemos que falla:



Encriptación de Base de datos TDE (Transparent Data Encryption)

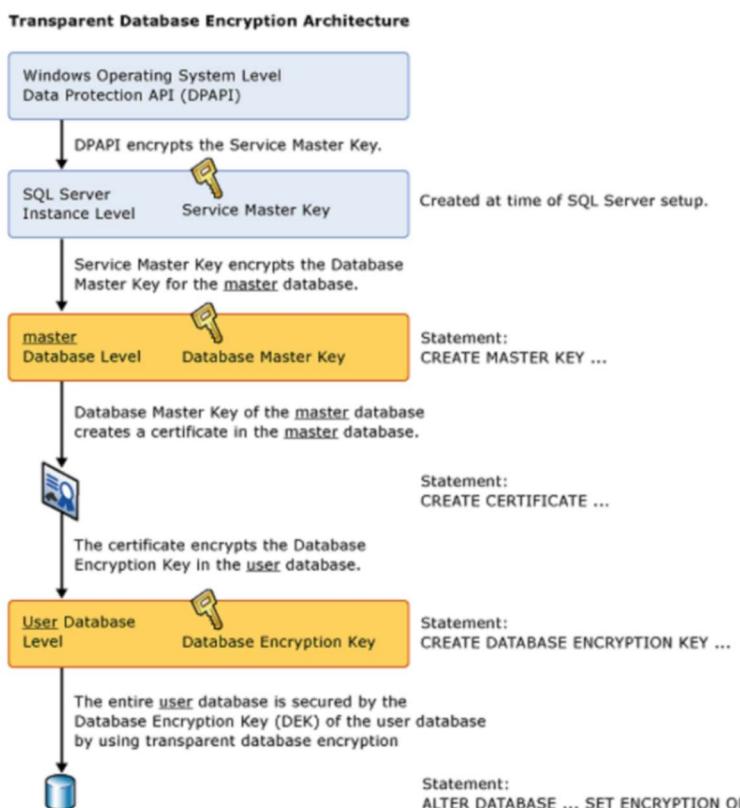
El Cifrado de datos transparente (TDE) cifra los archivos de datos de SQL Server, Azure SQL Database y Azure Synapse Analytics, lo que se conoce como cifrado de datos en reposo.

Para ayudar a proteger una base de datos, se pueden tomar precauciones como las siguientes:

- Diseñar un sistema seguro
- Cifrar los recursos confidenciales
- Crear un firewall en torno a los servidores de bases de datos

Aún así, alguien malintencionado robando medios físicos como unidades o cintas de copia de seguridad podría restaurar la base de datos o conectarse a ella y examinar sus datos. Para ello, una solución sería cifrar los datos confidenciales en la base de datos y usar un certificado para proteger las claves con las que esos datos se cifran. Esta solución impide que alguien que carezca de las claves use los datos. Este tipo de protección se debe planear de antemano.

TDE realiza el cifrado y descifrado de E/S en tiempo real de los archivos de datos y de registro. Este cifrado usa una clave de cifrado de la base de datos (DEK). El registro de arranque de la base de datos almacena la clave para que esté disponible durante la recuperación. La DEK es una clave simétrica. Está protegida por un certificado que la base de datos maestra del servidor almacena, o por una clave asimétrica que un módulo EKM protege.



TDE protege los datos en reposo, que son los archivos de datos y de registro. Permite cumplir muchas leyes, normativas y directrices establecidas en diversos sectores. Esto permite a los desarrolladores de software cifrar datos con algoritmos de cifrado AES y 3DES sin cambiar las aplicaciones existentes.

Cuando TDE se usa con SQL Database V12, SQL Database crea automáticamente el certificado de nivel de servidor almacenado en la base de datos maestra.

Para mover una base de datos de TDE en SQL Database, no es necesario descifrarla para la operación de traslado.

Después de proteger una base de datos, se puede restaurar con el certificado correcto que corresponda.

Después de habilitar TDE, hay que hacer una copia de seguridad del certificado y su clave privada asociada. Si el certificado dejara de estar disponible, o si la base de datos restaura o conecta en otro servidor, necesitará copias de seguridad del certificado y de la clave privada o, de lo contrario, no podrá abrir la base de datos.

Hay que conservar el certificado de cifrado, aunque TDE esté deshabilitado en la base de datos. Si bien la base de datos no está cifrada, puede que algunas partes del registro de transacciones sigan protegidas.

También puede necesitar el certificado en algunas operaciones hasta que haga una copia de seguridad completa de la base de datos.

Un certificado que ha excedido su fecha de expiración se puede seguir usando para cifrar y descifrar datos con TDE.

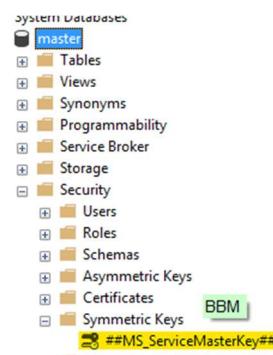
Para habilitar TDE según las buenas prácticas de Microsoft, seguimos estos pasos:

Creación de una clave maestra:

```
USE [master];
GO

drop master key
go

-- Create the database master key
-- to encrypt the certificate
CREATE MASTER KEY
    ENCRYPTION BY PASSWORD = 'Abcd1234.';
GO
```



Creación/obtención de un certificado protegido por la clave maestra.

```
-- Create the certificate we're going to use for TDE
CREATE CERTIFICATE TDEBBMASPACE
    WITH SUBJECT = 'TDE Cert for Project';
GO

-- CERTIFICATES T-SQL

SELECT TOP 1 *
FROM sys.certificates
ORDER BY name DESC
GO
```



name	certificate_id	principal_id	pvt_key_encryption_type	pvt_key_encryption_type_desc	is_active_for_begin_dialog	issuer_name	cert_serial_number	sid
TDEBBMASPACE	268	1	MK	ENCRYPTED_BY_MASTER_KEY	1	TDE Cert for Project	22e8f8e76e881b9b4d0d6c434c0bf7d	0x0106000000000000

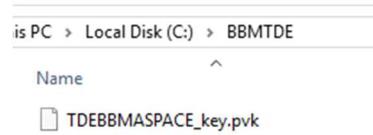
Le hacemos un *backup* al certificado para en caso de perderlo, recuperarlo fácil

```
-- Back up the certificate and its private key
-- Remember the password!
```

```

BACKUP CERTIFICATE TDEBBMSPACE
TO FILE = 'C:\BBMTDE\TDEBBMSPACE.cer'
WITH PRIVATE KEY (
FILE = 'C:\BBMTDE\TDEBBMSPACE_key.pvk',
ENCRYPTION BY PASSWORD = 'Abcd1234.'
);
GO
-- Look at Folder C:\BBMTDE

```



Creamos una clave de cifrado de base de datos y la protegemos con el certificado.

```

-- Create the DEK (DATABASE ENCRYPTION KEY) so we can turn on encryption
USE BBM_ASPACE;
GO

CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE TDEBBMSPACE;
GO

-- INFORMATION

SELECT *
FROM sys.dm_database_encryption_keys
GO

```

Results	Messages								
database_id	encryption_state	create_date	regenerate_date	modify_date	set_date	opened_date	key_algorithm	key_length	encryptor_thumbprint
1	1	2021-05-04 16:32:55.907	2021-05-04 16:32:55.907	2021-05-04 16:32:55.907	1900-01-01 00:00:00.000	2021-05-04 16:32:55.907	AES	256	0xB33423C40B359BC92C5979EB4

Configure la base de datos para que use el cifrado.

```

-- Exit out of the database. If we have an active
-- connection, encryption won't complete.
USE [master];
GO

-- Turn on TDE
-- T-SQL OR SSMS

ALTER DATABASE [BBM_ASPACE] SET ENCRYPTION ON;
GO

SELECT DB_Name(database_id) AS 'Database', encryption_state
FROM sys.dm_database_encryption_keys;
GO

```

	Database	encryption_state
1	tempdb	3
2	BBM_ASPACE	3

NOTA: `sys.dm_database_encryption_keys` devuelve información sobre el estado de cifrado de una base de datos y sus claves de cifrado de la base de datos asociadas

Columna	Tipo	Descripción
database_id	integer	Identificador de la base de datos. Indica si la base de datos está cifrada o no.
	0	Ninguna clave de cifrado de la base de datos, sin cifrado
	1	Sin cifrar
	2	Cifrado en curso
encryption_state	integer	3 Cifrado 4 Cambio de clave en curso 5 Descifrado en curso 6 Cambio de protección en curso (El certificado o clave asimétrica que cifra la clave de cifrado de la base de datos se está cambiando).

Hacemos un backup de la base de datos.

```
BACKUP DATABASE [BBM_ASPACE]
TO DISK = 'C:\BBMTDE\BACKBBMASPACE_Full.bak';
GO

[!] Messages
Processed 456 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_PRINCIPAL' on file 1.
Processed 112 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_SECONDARY' on file 1.
Processed 2 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_log' on file 1.
BACKUP DATABASE successfully processed 570 pages in 0.565 seconds (7.868 MB/sec).

BACKUP LOG [BBM_ASPACE]
TO DISK = 'C:\BBMTDE\BACKBBMASPACE_log.bak'
WITH NORECOVERY
GO

[!] Messages
Processed 53 pages for database 'BBM_ASPACE', file 'BBM_ASPACE_log' on file 1.
BACKUP LOG successfully processed 53 pages in 0.145 seconds (2.818 MB/sec).
```

Encryption vs Hashing

Ya hemos visto lo que es la encriptación, pero ¿Es el único método para ocultar nuestros datos? En este apartado dejaré las diferencias entre **Hashing** y **Encriptación**.



El hash es la práctica de utilizar un algoritmo para asignar datos de cualquier tamaño a una longitud fija (**hash**).

Mientras que el **cifrado es una función bidireccional**, el **hash es una función unidireccional**. Si bien es técnicamente posible hacer un hash inverso de algo, la potencia informática requerida lo hace inviable. El hash es unidireccional.

Ahora, mientras que el **cifrado está destinado a proteger los datos en tránsito**, el **hash está destinado a verificar que un archivo o un dato no se haya alterado**, que sea auténtico. En otras palabras, sirve como una suma de control.

Así es como funciona, **cada algoritmo de hashing genera una longitud fija**. Entonces, por ejemplo, puede escuchar sobre SHA-256, eso significa que el algoritmo generará un valor hash de 256 bits, generalmente representado por una cadena hexadecimal de 64 caracteres.

Cada valor **hash es único**. Si dos archivos diferentes producen el mismo valor hash único, esto se denomina **colisión** y hace que el algoritmo sea esencialmente inútil.

Un ejemplo de hash pudo ser: Si se desea firmar digitalmente un software y hacer que esté disponible para su descarga en un sitio web. Para hacer esto, se va a crear un hash del script o ejecutable que se está firmando, luego de agregar la firma digital también se hará. Después de esto, todo está encriptado para que pueda descargarse.

Cuando un cliente descarga el software, su navegador descifrará el archivo y luego inspeccionará los dos valores hash únicos. A continuación, el navegador ejecutará la misma función hash, utilizando el mismo algoritmo, y volverá a hacer hash tanto en el archivo como en la firma. Si el navegador produce el mismo valor hash, entonces sabe que tanto la firma como el archivo son auténticos, no han sido alterados.

Si no es así, el navegador emite una advertencia.



Algunos de los algoritmos hash más comunes que se utilizan en la actualidad son:



MD4: MD4 es un algoritmo hash que se autodesprecia, creado en 1990, incluso su creador, Ronald Rivest, admite que tiene problemas de seguridad. Sin embargo, el algoritmo hash de 128 bits tuvo un impacto, su influencia se puede sentir en algoritmos más recientes como WMD5, WRIPEMD y la familia WHSA.

MD5: MD5 es otro algoritmo de hash creado por Ray Rivest que se sabe que sufre vulnerabilidades. Fue creado en 1992 como sucesor de

MD4. Actualmente MD6 está en proceso, pero a partir de 2009 Rivest lo había eliminado de la consideración del NIST para SHA-3.

SHA: SHA significa *Security Hashing Algorithm* y probablemente sea más conocido como el algoritmo de hash utilizado en la mayoría de las suites de cifrado *SSL / TLS*. Un conjunto de cifrado es una colección de cifrados y algoritmos que se utilizan para conexiones *SSL / TLS*. SHA maneja los aspectos de hash. SHA-1 ahora está en desuso. SHA-2 ahora es obligatorio. A veces se sabe que SHA-2 tiene SHA-256, aunque también están disponibles variantes con longitudes de bits más largas.

RIPEMD: es una familia de algoritmos hash criptográficos con longitudes de 128, 160, 256 y 320 bits. Fue desarrollado bajo el marco del Proyecto Maduro de la UE por Hans Dobbertin y un grupo de académicos en 1996. Sus variantes de 256 y 320 bits en realidad no agregan ninguna seguridad adicional, solo disminuyen el potencial de una colisión. En 2004, se informó una colisión para RIPEMD-128, lo que significa que RIPEMD-160 es el único algoritmo de esta familia que vale la pena (este será un juego de palabras increíble en aproximadamente dos párrafos).

WHIRLPOOL: Diseñado por Victor Rijmen (el co-creador del algoritmo AES que discutimos anteriormente) y Paulo Barreto en 2000. Desde entonces ha sido objeto de dos revisiones. Produce hashes de 512 bits que normalmente se representan como números hexadecimales de 128 dígitos.

TIGER: es un algoritmo bastante nuevo que está comenzando a ganar algo de tracción con las redes de intercambio de archivos y los sitios de torrents. Actualmente no hay ataques conocidos que sean efectivos contra su variante completa de 24 asaltos.

Después de estas consideraciones, podemos deducir las siguientes cosas:

HASHING	ENCRYPTION
Los datos convertidos en una cadena ilegible no pueden volver a convertirse en una cadena de caracteres legibles.	Los datos cifrados pueden descifrarse y convertirse en una cadena de caracteres legibles (información en texto plano) con la ayuda de claves criptográficas.
Los caracteres ilegibles tienen una longitud fija.	Los caracteres ilegibles no tienen una longitud fija.
No hay uso de claves en el hashing.	El cifrado se realiza con la ayuda de claves. En el caso del cifrado simétrico, sólo se utilizan las claves públicas. En el cifrado asimétrico, se utilizan tanto las claves públicas como las privadas.

Dejo la transcripción del script de clase en mi proyecto como paso final haciendo una prueba de *hash*:

```
USE [BBM_ASPACE];
GO

-- Basic example
DECLARE @BBM_HASHING NVARCHAR(MAX) = 'Project Hash inclusion'
SELECT HASHBYTES ('SHA2_512', @BBM_HASHING) AS [Hash value];
GO

-- All the hashing algorithms
DECLARE @BBM_HASHING NVARCHAR(MAX) = 'Project Hash inclusion'
```

```
SELECT HASHBYTES ('MD2', @BBM_HASHING) AS [Hash value],
DATALENGTH(HASHBYTES ('MD2', @BBM_HASHING)) AS [Data lenght];
SELECT HASHBYTES ('MD4', @BBM_HASHING) AS [Hash value],
DATALENGTH(HASHBYTES ('MD4', @BBM_HASHING)) AS [Data lenght];
SELECT HASHBYTES ('MD5', @BBM_HASHING) AS [Hash value],
DATALENGTH(HASHBYTES ('MD5', @BBM_HASHING)) AS [Data lenght];
SELECT HASHBYTES ('SHA', @BBM_HASHING) AS [Hash value],
DATALENGTH(HASHBYTES ('SHA', @BBM_HASHING)) AS [Data lenght];
SELECT HASHBYTES ('SHA1', @BBM_HASHING) AS [Hash value],
DATALENGTH(HASHBYTES ('SHA1', @BBM_HASHING)) AS [Data lenght];
SELECT HASHBYTES ('SHA2_256', @BBM_HASHING) AS [Hash value],
DATALENGTH(HASHBYTES ('SHA2_256', @BBM_HASHING)) AS [Data lenght];
SELECT HASHBYTES ('SHA2_512', @BBM_HASHING) AS [Hash value],
DATALENGTH(HASHBYTES ('SHA2_512', @BBM_HASHING)) AS [Data lenght];
GO

-- No salt
DECLARE @BBM_HASHING NVARCHAR(MAX) = 'Project Hash inclusion'
SELECT HASHBYTES ('SHA2_512', @BBM_HASHING) AS [Hash value 1];
SELECT HASHBYTES ('SHA2_512', @BBM_HASHING) AS [Hash value 2];

-- Different data types
SELECT HASHBYTES ('SHA2_512', N'Gin tonic') AS [Hash value 1];
SELECT HASHBYTES ('SHA2_512', 'Gin tonic') AS [Hash value 2];

-- Get hashes for customer invoices
USE [BBM_ASPACE];
GO
SELECT
    trial.BBM_Tratamientos.Tratamiento_ID
    , HASHBYTES ('SHA2_512', (
        SELECT
            trial.BBM_Usuario.Usuario_ID,
            trial.BBM_Tratamientos.Tratamiento_ID, trial.BBM_STANSARDPRICE.Precio
        FROM
            trial.BBM_Usuario CROSS JOIN
                trial.BBM_Tratamientos CROSS JOIN
                    trial.BBM_STANSARDPRICE
        WHERE
            Precio = '10'
        FOR XML AUTO
    ) AS [Invoices hash]
FROM
    trial.BBM_Tratamientos
```

Dynamic Data Masking

El **enmascaramiento dinámico de datos** es una nueva característica de seguridad introducida en SQL Server 2016 que limita el acceso de usuarios no autorizados a los datos sensibles en la capa de base de datos.

Como ejemplo de la necesidad de dicha característica es permitir a los desarrolladores de aplicaciones acceder a los datos de producción para la resolución de problemas y evitar que accedan a los datos sensibles al mismo tiempo, sin afectar a su proceso de resolución de problemas. Otro ejemplo es el empleado del centro de llamadas que accederá a la información del cliente para ayudarle en su solicitud, pero los datos financieros críticos, como el número de cuenta bancaria o el número completo de la tarjeta de crédito, estarán enmascarados para esa persona.

El enmascaramiento dinámico de datos, **también conocido como DDM**, es una función de seguridad muy sencilla que puede construirse por completo utilizando comandos T-SQL. Este método de protección de datos permite determinar datos "*sensibles*", por campo, para configurar la función de enmascaramiento adecuada para ocultarlos de las consultas. Esta función no requiere ningún esfuerzo de codificación por parte de la aplicación, ni cifrar o aplicar ningún cambio a los datos reales almacenados en el disco.

Enmascara los datos sensibles "sobre la marcha" para protegerlos de los usuarios sin privilegios mediante funciones de enmascaramiento integradas o personalizadas, sin impedirles recuperar los datos desenmascarados.

Para implantar el DDM, primero hay que especificar los datos sensibles, el rol para enmascararlos y especificar los usuarios privilegiados designados que tienen acceso a esos datos sensibles. El siguiente paso es seleccionar e implementar una función de enmascaramiento.

Antes de empezar el ejemplo, creo una base de datos pequeña para aislar una de las tablas y hacer pruebas con ella.

```
-- DYNAMIC DATA MASKING (DDM)
USE MASTER

DROP DATABASE IF EXISTS BBM_ASPACE_SECURETEST
GO
CREATE DATABASE BBM_ASPACE_SECURETEST;
GO

USE [BBM_ASPACE_SECURETEST]
GO

-- VOLCAMOS LOS DATOS DE LA TABLA CONTACTO PACIENTE A UNA NUEVA TABLA
CREATE TABLE [dbo].[BBM_Contacto_Paciente_TEST](
    [BBM_Tutor_Tutor_ID] [varchar](10) NOT NULL,
    [BBM_Usuario_Usuario_ID] [varchar](10) NOT NULL,
    [Contacto] [int] NULL,
    [Email] [varchar](255) NULL,
    [Direccion] [varchar](255) NULL,
    [Otros_Detalles] [varchar](255) NULL
)
GO
```

Inserto los datos en la tabla para el test.

```
-- INSERTAMOS DATOS PARA LA PRUEBA
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR01','SONUSR01','981222221','correoejemplo01@mail.com','Calle
Falsa 121','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR02','SONUSR02','981222222','correoejemplo02@mail.com','Calle
Falsa 122','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR03','SONUSR03','981222223','correoejemplo03@mail.com','Calle
Falsa 123','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR04','SONUSR04','981222224','correoejemplo04@mail.com','Calle
Falsa 124','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR05','SONUSR05','981222225','correoejemplo05@mail.com','Calle
Falsa 125','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR06','SONUSR06','981222226','correoejemplo06@mail.com','Calle
Falsa 126','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR07','SONUSR07','981222227','correoejemplo07@mail.com','Calle
Falsa 127','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR08','SONUSR08','981222228','correoejemplo08@mail.com','Calle
Falsa 128','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
INSERT      INTO      [dbo].[BBM_Contacto_Paciente_TEST]      VALUES
('DADUSR09','SONUSR09','981222229','correoejemplo09@mail.com','Calle
Falsa 129','INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER')
GO
```

Comprobamos los datos en la tabla para el test.

/*VEMOS LOS RESULTADOS*/

```
SELECT * FROM dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	981222221	correoejemplo01@mail.com	Calle Falsa 121	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	981222222	correoejemplo02@mail.com	Calle Falsa 122	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	981222223	correoejemplo03@mail.com	Calle Falsa 123	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	981222224	correoejemplo04@mail.com	Calle Falsa 124	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	981222225	correoejemplo05@mail.com	Calle Falsa 125	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	981222226	correoejemplo06@mail.com	Calle Falsa 126	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	981222227	correoejemplo07@mail.com	Calle Falsa 127	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	981222228	correoejemplo08@mail.com	Calle Falsa 128	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	981222229	correoejemplo09@mail.com	Calle Falsa 129	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

Para este ejemplo creamos roles e incluimos dos usuarios:

- Padremiron al que no le voy a dejar ver nada de lo que enmascare
- Psicologo al que al final le voy a dejar ver datos que el paciente no

```
-- Create DataUser to have Select access to dbo.BBM_Contacto_Paciente_TEST table
CREATE ROLE BBMDATAMASKINGPROS
GO
CREATE ROLE BBMDATAMASKINGUSERS
GO

GRANT SELECT ON dbo.BBM_Contacto_Paciente_TEST TO BBMDATAMASKINGPROS;
GO
GRANT SELECT ON dbo.BBM_Contacto_Paciente_TEST TO BBMDATAMASKINGUSERS;
GO

CREATE USER PSICOLOGO WITHOUT LOGIN;
GO
CREATE USER PADREMIRON WITHOUT LOGIN;
GO

ALTER ROLE BBMDATAMASKINGPROS ADD MEMBER PSICOLOGO
GO
ALTER ROLE BBMDATAMASKINGUSERS ADD MEMBER PADREMIRON
GO
```

Creamos un procedimiento almacenado para revisar el estado del DDM.

```
-- Stored procedure to check dynamic data masking status
CREATE OR ALTER PROC BBM_SHOW_STATUS
AS
BEGIN
    SET NOCOUNT ON
    SELECT c.name, tbl.name AS table_name, c.is_masked,
    c.masking_function
    FROM sys.masked_columns AS c
    JOIN sys.tables AS tbl
        ON c.[object_id] = tbl.[object_id]
    WHERE is_masked = 1;
END
GO

EXEC BBM_SHOW_STATUS
GO
```

name	table_name	is_masked	masking_function
------	------------	-----------	------------------

NOTA: se supone que ambos tienen los mismos permisos, pero al final voy a revocar el permiso de masking al psicologo

Hay cuatro tipos principales de funciones de enmascaramiento que pueden configurarse en el Enmascaramiento Dinámico de Datos, que presentaremos brevemente aquí y utilizaremos en la demostración más adelante.

El procedimiento siempre es el mismo en todos los casos:

- Primero hacemos un Alter de la tabla sobre la columna
- Vemos el estado de con el procedimiento almacenado
- Nos impersonamos y comprobamos
- Volvemos a dbo y comprobamos

Default

La máscara por defecto, enmascara los valores completos de la columna especificada. Para especificar una máscara para una columna en particular, tiene que utilizar la cláusula "MASKED WITH". Dentro de la cláusula MASKED WITH, tiene que especificar la FUNCIÓN que desea utilizar para el enmascaramiento. Si desea realizar un enmascaramiento por defecto, utilice la función "default()".

```
--Default dynamic data masking of Email column
ALTER TABLE dbo.BBM_Contacto_Paciente_TEST
ALTER COLUMN Email varchar(255) MASKED WITH (FUNCTION = 'default()');
GO
```

```
EXEC BBM_SHOW_STATUS
GO
```

	name	table_name	is_masked	masking_function
1	Email	BBM_Contacto_Paciente	1	default()

```
-- Execute SELECT as DataUser
EXECUTE AS USER = 'PADREMIRO';
GO
-- View monthly sales
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	98122221	xxxx	Calle Falsa 121	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	98122222	xxxx	Calle Falsa 122	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	98122223	xxxx	Calle Falsa 123	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	98122224	xxxx	Calle Falsa 124	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	98122225	xxxx	Calle Falsa 125	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	98122226	xxxx	Calle Falsa 126	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	98122227	xxxx	Calle Falsa 127	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	98122228	xxxx	Calle Falsa 128	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	98122229	xxxx	Calle Falsa 129	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

```
-- Revert the User back to what user it was before
REVERT;
GO
```

```
PRINT USER
GO
-- DBO
```

```
-- View monthly sales
```

```
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	98122221	correo@ejemplo01@mail.com	Calle Falsa 121	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	98122222	correo@ejemplo02@mail.com	Calle Falsa 122	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	98122223	correo@ejemplo03@mail.com	Calle Falsa 123	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	98122224	correo@ejemplo04@mail.com	Calle Falsa 124	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	98122225	correo@ejemplo05@mail.com	Calle Falsa 125	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	98122226	correo@ejemplo06@mail.com	Calle Falsa 126	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	98122227	correo@ejemplo07@mail.com	Calle Falsa 127	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	98122228	correo@ejemplo08@mail.com	Calle Falsa 128	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	98122229	correo@ejemplo09@mail.com	Calle Falsa 129	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

Partial

¿Qué pasa si queremos mostrar parcialmente la información de la columna dejando alguna parte oculta?

Aquí es donde las máscaras parciales resultan útiles. Para utilizar una máscara parcial, hay que pasar "partial(start characters, mask, end characters" como valor del parámetro de la función de la cláusula MASKED WITH. Es importante mencionar que la máscara parcial sólo es aplicable a las columnas de tipo cadena.

```
-- Partial data masking of Customer names
```

```
ALTER TABLE dbo.BBM_Contacto_Paciente_TEST
ALTER COLUMN [Direccion] ADD MASKED WITH (FUNCTION = 
'partial(1, "xxxxx", 0)')
GO
```

```
EXEC BBM_SHOW_STATUS
GO
```

	name	table_name	is_masked	masking_function
1	Contacto	BBM_Contacto_Paciente_TEST	1	partial(1, "xxxxx", 0)
2	Email	BBM_Contacto_Paciente_TEST	1	default()
3	Direccion	BBM_Contacto_Paciente_TEST	1	partial(1, "xxxxx", 0)

```
-- Execute SELECT as DataUser
EXECUTE AS USER = 'PSICOLOGO';
```

```
-- View monthly sales as DataUser
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	9xxxxx	xxxx	Cxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

```
-- Revert the User back to what user it was before
```

```

REVERT;
GO
PRINT USER
GO
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO

```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	98122221	correoejemplo01@mail.com	Calle Falsa 121	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	98122222	correoejemplo02@mail.com	Calle Falsa 122	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	98122223	correoejemplo03@mail.com	Calle Falsa 123	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	98122224	correoejemplo04@mail.com	Calle Falsa 124	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	98122225	correoejemplo05@mail.com	Calle Falsa 125	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	98122226	correoejemplo06@mail.com	Calle Falsa 126	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	98122227	correoejemplo07@mail.com	Calle Falsa 127	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	98122228	correoejemplo08@mail.com	Calle Falsa 128	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	98122229	correoejemplo09@mail.com	Calle Falsa 129	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

Random

La máscara aleatoria se utiliza para enmascarar las columnas de enteros con valores aleatorios. El rango de los valores aleatorios se especifica mediante la función aleatoria

```
--Random dynamic data masking of TotalPrice column
```

```

ALTER TABLE dbo.BBM_Contacto_Paciente_TEST
ALTER COLUMN [Contacto] integer MASKED WITH (FUNCTION = 'random(1,
5000)')
GO

```

```

EXEC BBM_SHOW_STATUS
GO

```

	name	table_name	is_masked	masking_function
1	Contacto	BBM_Contacto_Paciente_TEST	1	random(1, 5000)
2	Email	BBM_Contacto_Paciente_TEST	1	default()
3	Direccion	BBM_Contacto_Paciente_TEST	1	partial(1, "xxxx", 0)

```

-- Execute SELECT as DataUser
EXECUTE AS USER = 'PADREMIRON';
GO
-- View monthly sales
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO

```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	292	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	2693	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	1389	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	3759	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	657	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	3570	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	444	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	3564	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	4895	xxxx	xxxxxx	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

```
-- Revert the User back to what user it was before
REVERT;
GO
-- DBO
-- View monthly sales
SELECT *
FROM dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	98122221	correoejemplo01@mail.com	Calle Falsa 121	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	98122222	correoejemplo02@mail.com	Calle Falsa 122	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	98122223	correoejemplo03@mail.com	Calle Falsa 123	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	98122224	correoejemplo04@mail.com	Calle Falsa 124	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	98122225	correoejemplo05@mail.com	Calle Falsa 125	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	98122226	correoejemplo06@mail.com	Calle Falsa 126	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	98122227	correoejemplo07@mail.com	Calle Falsa 127	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	98122228	correoejemplo08@mail.com	Calle Falsa 128	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	98122229	correoejemplo09@mail.com	Calle Falsa 129	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

Custom

Creada de manera personalizada

```
--Custom string dynamic data masking of Product column

ALTER TABLE dbo.BBM_Contacto_Paciente_TEST
ALTER COLUMN [Otros_Detalles] ADD MASKED WITH (FUNCTION = 'partial(1,"--",1)')
GO

EXEC BBM_SHOW_STATUS
GO
```

	name	table_name	is_masked	masking_function
1	Contacto	BBM_Contacto_Paciente_TEST	1	random(1, 5000)
2	Direccion	BBM_Contacto_Paciente_TEST	1	partial(1, "xxxx", 0)
3	Otros_Detalles	BBM_Contacto_Paciente_TEST	1	partial(1, "--", 1)

```
-- Execute SELECT as DataUser
EXECUTE AS USER = 'PSICOLOGO';
GO
-- View monthly sales
SELECT *
FROM dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	292	xxxx	xxxxxx	I--R
2	DADUSR02	SONUSR02	2693	xxxx	xxxxxx	I--R
3	DADUSR03	SONUSR03	1389	xxxx	xxxxxx	I--R
4	DADUSR04	SONUSR04	3759	xxxx	xxxxxx	I--R
5	DADUSR05	SONUSR05	657	xxxx	xxxxxx	I--R
6	DADUSR06	SONUSR06	3570	xxxx	xxxxxx	I--R
7	DADUSR07	SONUSR07	444	xxxx	xxxxxx	I--R
8	DADUSR08	SONUSR08	3564	xxxx	xxxxxx	I--R
9	DADUSR09	SONUSR09	4895	xxxx	xxxxxx	I--R

```
-- Revert the User back to what user it was before
REVERT;
GO
PRINT USER
GO

-- dbo

-- View monthly sales
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO
```

Results Messages

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	292	correoejemplo01@mail.com	Cxxxxx	I-R
2	DADUSR02	SONUSR02	2693	correoejemplo02@mail.com	Cxxxxx	I-R
3	DADUSR03	SONUSR03	1389	correoejemplo03@mail.com	Cxxxxx	I-R
4	DADUSR04	SONUSR04	3759	correoejemplo04@mail.com	Cxxxxx	I-R
5	DADUSR05	SONUSR05	657	correoejemplo05@mail.com	Cxxxxx	I-R
6	DADUSR06	SONUSR06	3570	correoejemplo06@mail.com	Cxxxxx	I-R
7	DADUSR07	SONUSR07	444	correoejemplo07@mail.com	Cxxxxx	I-R
8	DADUSR08	SONUSR08	3564	correoejemplo08@mail.com	Cxxxxx	I-R
9	DADUSR09	SONUSR09	4895	correoejemplo09@mail.com	Cxxxxx	I-R

Pruebas de Desenmascaramiento

Vamos a probar a darle a UNMASK al rol de **PROFESIONALES**, y probar a ver qué ven los usuarios

```
-- ¿Cómo puedo hacer que el Usuario vea? --> Con Grant Unmask
GRANT UNMASK TO BBMDATAMASKINGPROS
GO

-- Execute SELECT as DataUser
EXECUTE AS USER = 'PSICOLOGO';
GO
-- View monthly sales
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	98122221	correoejemplo01@mail.com	Calle Falsa 121	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
2	DADUSR02	SONUSR02	98122222	correoejemplo02@mail.com	Calle Falsa 122	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
3	DADUSR03	SONUSR03	98122223	correoejemplo03@mail.com	Calle Falsa 123	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
4	DADUSR04	SONUSR04	98122224	correoejemplo04@mail.com	Calle Falsa 124	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
5	DADUSR05	SONUSR05	98122225	correoejemplo05@mail.com	Calle Falsa 125	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
6	DADUSR06	SONUSR06	98122226	correoejemplo06@mail.com	Calle Falsa 126	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
7	DADUSR07	SONUSR07	98122227	correoejemplo07@mail.com	Calle Falsa 127	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
8	DADUSR08	SONUSR08	98122228	correoejemplo08@mail.com	Calle Falsa 128	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER
9	DADUSR09	SONUSR09	98122229	correoejemplo09@mail.com	Calle Falsa 129	INFORMACIÓN SENSIBLE QUE NO SE PUEDE VER

```
-- Revert the User back to what user it was before
REVERT;
GO

/*PROBAMOS CON EL USUARIO*/
EXECUTE AS USER = 'PADREMIRON';
GO
-- View monthly sales
SELECT *
```

```
from dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	136	xxxx	Coooox	I--R
2	DADUSR02	SONUSR02	3135	xxxx	Coooox	I--R
3	DADUSR03	SONUSR03	3365	xxxx	Coooox	I--R
4	DADUSR04	SONUSR04	3598	xxxx	Coooox	I--R
5	DADUSR05	SONUSR05	2055	xxxx	Coooox	I--R
6	DADUSR06	SONUSR06	840	xxxx	Coooox	I--R
7	DADUSR07	SONUSR07	692	xxxx	Coooox	I--R
8	DADUSR08	SONUSR08	4686	xxxx	Coooox	I--R
9	DADUSR09	SONUSR09	68	xxxx	Coooox	I--R

```
REVERT;
GO
```

Probamos a revocar sobre la columna **email** y vemos que con los usuarios podríamos verlo

```
-- Dropping a Dynamic Data Mask --> Con esto puedo borrar máscara
```

```
ALTER TABLE dbo.BBM_Contacto_Paciente_TEST
ALTER COLUMN Email DROP MASKED;
GO
```

```
EXEC BBM_SHOW_STATUS
GO
```

	name	table_name	is_masked	masking_function
1	Contacto	BBM_Contacto_Paciente_TEST	1	random(1, 5000)
2	Email	BBM_Contacto_Paciente_TEST	1	default()
3	Direccion	BBM_Contacto_Paciente_TEST	1	partial(1, 'xxxxx', 0)

```
-- Execute SELECT as DataUser
EXECUTE AS USER = 'PADREMIRON';
GO
-- View monthly sales
SELECT *
from dbo.BBM_Contacto_Paciente_TEST
GO
```

	BBM_Tutor_Tutor_ID	BBM_Usuario_Usuario_ID	Contacto	Email	Direccion	Otros_Detalles
1	DADUSR01	SONUSR01	136	correoempl01@mail.com	Coooox	I--R
2	DADUSR02	SONUSR02	3135	correoempl02@mail.com	Coooox	I--R
3	DADUSR03	SONUSR03	3365	correoempl03@mail.com	Coooox	I--R
4	DADUSR04	SONUSR04	3598	correoempl04@mail.com	Coooox	I--R
5	DADUSR05	SONUSR05	2055	correoempl05@mail.com	Coooox	I--R
6	DADUSR06	SONUSR06	840	correoempl06@mail.com	Coooox	I--R
7	DADUSR07	SONUSR07	692	correoempl07@mail.com	Coooox	I--R
8	DADUSR08	SONUSR08	4686	correoempl08@mail.com	Coooox	I--R
9	DADUSR09	SONUSR09	68	correoempl09@mail.com	Coooox	I--R

Row Level Security

Permite utilizar la pertenencia a un grupo o el contexto de ejecución para controlar el acceso a las filas de una tabla de base de datos. **RLS simplifica el diseño y la codificación de la seguridad** de la aplicación. **RLS permite implementar restricciones en el acceso a las filas de datos.**

La lógica de la restricción de acceso está ubicada en el **nivel de base de datos** en lugar de estar alejado de los datos en otro nivel de aplicación. El sistema de base de datos aplica las restricciones de acceso cada vez que se intenta acceder a los datos desde cualquier nivel. Esto hace que el sistema de seguridad resulte más sólido y confiable al reducir el área expuesta del sistema de seguridad.

RLS admite dos tipos de predicados de seguridad:

- Los **predicados de filtro** filtran en modo silencioso las filas disponibles para leer operaciones (**SELECT**, **UPDATE** y **DELETE**). Se aplican al leer los datos desde la tabla base y afectan a todas las operaciones **get**: **SELECT**, **DELETE** y **UPDATE**. Los usuarios no se pueden seleccionar o eliminar las filas filtradas. El usuario no puede actualizar las filas filtradas. Pero, es posible actualizar las filas de tal manera que se filtren después. Los predicados de bloqueo afectan a todas las operaciones de escritura.

El acceso a los datos de nivel de fila de una tabla está restringido por un predicado de seguridad que se define como una función con valores de tabla insertada. Luego, la función se invoca y una directiva de seguridad la aplica. En los predicados de filtro, la aplicación es consciente de las filas filtradas del conjunto de resultados. Si se filtran todas las filas, se devolverá un conjunto nulo. En el caso de los predicados de bloqueo, las operaciones que infrinjan el predicado generarán un error.

- Los **predicados de bloqueo** bloquean explícitamente las operaciones de escritura (**AFTER INSERT**, **AFTER UPDATE**, **BEFORE UPDATE**, **BEFORE DELETE**) que infringen el predicado. **AFTER INSERT** y **AFTER UPDATE** pueden impedir que los usuarios actualicen las filas con valores que infrinjan el predicado.

Los predicados **BEFORE UPDATE** pueden impedir que los usuarios actualicen las filas que actualmente infrinjan el predicado. Sin embargo, los predicados **BEFORE DELETE** pueden bloquear las operaciones de eliminación.

Los predicados de filtro y de bloqueo y las directivas de seguridad tienen el siguiente comportamiento:

- Puede **definir una función de predicado** que se combine con otra tabla o invoque una función. Si la directiva de seguridad se crea con **SCHEMABINDING = ON** (el valor predeterminado), entonces se puede acceder a la función o combinación desde la consulta, y funciona como se espera sin comprobaciones de permisos adicionales. Si la se crea con **SCHEMABINDING = OFF**, entonces los usuarios necesitarán los permisos **SELECT** en estas funciones y tablas adicionales para consultar la tabla de destino. Si la función de predicado invoca una función escalar de CLR, se necesita además el permiso **EXECUTE**.
- Puede **emitir una consulta a una tabla que tenga un predicado de seguridad definido pero deshabilitado**. Todas las filas que se han filtrado o bloqueado no se ven afectadas.

- Si el usuario **dbo**, un miembro del rol **db_owner** o el propietario de la tabla consulta una tabla que tiene una directiva de seguridad definida y habilitada, las filas se filtran o bloquean según indique la directiva de seguridad.
- Los intentos de **modificar el esquema** de una tabla enlazada por una directiva de seguridad enlazada a un esquema producirán un error. Sin embargo, se pueden modificar las columnas a las que el predicado no hace referencia.
- Los intentos de **agregar un predicado** a una tabla que ya tiene uno definido para la operación especificada producen un error. Esto ocurrirá tanto si el predicado está habilitado como si no.
- Los intentos de **modificar una función**, que se usa como predicado en una tabla dentro de una directiva de seguridad enlazada a un esquema, producen un error.

Definir varias directivas de seguridad activas que contienen predicados no superpuestos, será correcto.

- Los **predicados de filtro** tienen el siguiente comportamiento: **Definir una directiva de seguridad** que filtre las filas de una tabla. La aplicación no es consciente de las filas que se han filtrado para las operaciones **SELECT**, **UPDATE** y **DELETE**. Incluidas las situaciones en las que todas las filas se filtran. La aplicación puede aplicar **INSERT** a las filas, aunque se filtren durante cualquier otra operación.
- Los predicados de bloqueo tienen el siguiente comportamiento: Los **predicados de bloqueo** para **UPDATE** se dividen en operaciones independientes para **BEFORE** y **AFTER**. En consecuencia, no puede, por ejemplo, bloquear a los usuarios para que no actualicen una fila con un valor mayor que el actual. Si se requiere este tipo de lógica, debe usar desencadenadores con las tablas intermedias **DELETED** e **INSERTED** para hacer referencia a los valores antiguos y nuevos juntos.

El optimizador no comprobará un predicado de bloqueo **AFTER UPDATE** si no se ha cambiado ninguna de las columnas usadas por la función de predicado. Por ejemplo: alguien no debería poder cambiar un salario para que sea mayor de 100 000. Alguien podría cambiar la dirección de un empleado cuyo salario ya es superior a 100 000, siempre y cuando las columnas a las que se hace referencia en el predicado no hayan cambiado.

No se han realizado cambios en las API masivas, incluida **BULK INSERT**. Esto significa que los predicados de bloqueo **AFTER INSERT** se aplicarán a las operaciones de inserción masivas como si fueran operaciones de inserción normales. Casos de uso (estos son ejemplos de diseño de cómo se puede usar RLS):

Los predicados de filtro RLS son funcionalmente equivalentes a anexar una cláusula **WHERE**. En términos más formales, RLS presenta control de acceso basado en predicado. Ofrece una evaluación flexible, centralizada y basada en predicados. El predicado puede basarse en metadatos o en cualquier otro criterio que el administrador determine según corresponda. El predicado se usa como un criterio para determinar si el usuario tiene el acceso adecuado a los datos según los atributos del usuario. El control de acceso basado en etiquetas se puede implementar mediante el control de acceso basado en predicados.

En cuanto a los permisos, se puede hacer lo siguiente:

- Crear, modificar o quitar directivas de seguridad necesita el permiso **ALTER ANY SECURITY POLICY**. Crear o quitar una directiva de seguridad necesita el permiso **ALTER** en el esquema.

Además, son necesarios los siguientes permisos para cada predicado que se agrega:

- Los permisos **SELECT** y **REFERENCES** de la función que se usa como predicado.
- El permiso **REFERENCES** de la tabla de destino que se enlaza a la directiva.
- El permiso **REFERENCES** de cada columna desde la tabla de destino que se usa como argumento.

Las directivas de seguridad se aplican a todos los usuarios, incluidos los usuarios **dbo** de la base de datos. Los usuarios **dbo** pueden modificar o quitar directivas de seguridad, sin embargo, se pueden auditar los cambios en las directivas de seguridad. Si los usuarios con privilegios elevados, como **sysadmin** o **db_owner**, necesitan ver todas las filas para solucionar problemas o validar los datos, la directiva de seguridad debe estar escrita de modo que lo permita.

Si se crea una directiva de seguridad con **SCHEMABINDING = OFF**, los usuarios deben tener el permiso de **SELECT** o **EXECUTE** en la función de predicado y cualquier función, vista o tabla adicional que se use dentro de la función de predicado para consultar la tabla de destino. Si se crea una directiva de seguridad con **SCHEMABINDING = ON** (el valor predeterminado), entonces estas comprobaciones de permiso se omiten cuando los usuarios consultan la tabla de destino.

Como Best Practises, se recomienda crear un esquema independiente para los objetos RLS: función de predicado y directivas de seguridad. Esto ayuda a separar los permisos que son necesarios en estos objetos especiales de las tablas de destino. Puede ser necesario separar adicionalmente las distintas directivas y funciones de predicado en las bases de datos multinquilino, pero no por norma en cada caso.

El permiso **ALTER ANY SECURITY POLICY** está destinado a los usuarios con privilegios elevados (como un administrador de directivas de seguridad). El administrador de directivas de seguridad no necesita el permiso **SELECT** en las tablas que protege.

Evitar conversiones de tipos en funciones de predicado para evitar posibles errores en tiempo de ejecución.

Evitar la recursividad en funciones de predicado siempre que sea posible para evitar la degradación del rendimiento. El optimizador de consultas intentará detectar recursividades directas, pero no garantiza encontrar las indirectas. Una recursividad indirecta es cuando una segunda función llama a la función de predicado.

Evitar el uso de combinaciones de tablas de forma excesiva en funciones de predicado para maximizar el rendimiento.

Evitar la lógica del predicado que dependa de opciones **SET** específicas de la sesión. Las funciones de predicado cuya lógica depende de determinadas opciones **SET** específicas de la sesión pueden perder información si los usuarios pueden ejecutar consultas arbitrarias. En general, las funciones de predicado deben cumplir las reglas siguientes:

Las funciones de predicado no deben convertir implícitamente cadenas de caracteres en date, smalldatetime, datetime, datetime2 o datetimeoffset o viceversa, ya que estas conversiones se ven afectadas por las opciones **SET DATEFORMAT** (*Transact-SQL*) y **SET LANGUAGE** (*Transact-SQL*). En su lugar, use la función **CONVERT** y especifique explícitamente el parámetro de estilo.

Las funciones de predicado no deben depender del valor del primer día de la semana, ya que este valor se ve afectado por la opción **SET DATEFIRST** (*Transact-SQL*).

Las funciones de predicado no deben depender de expresiones aritméticas o de agregación que devuelvan NULL en caso de error (como desbordamiento o división por cero), ya que este comportamiento se ve afectado por las opciones **SET ANSI_WARNINGS** (*Transact-SQL*), **SET NUMERIC_ROUNDABORT** (*Transact-SQL*) y **SET ARITHABORT** (*Transact-SQL*).

Las funciones de predicado no deben comparar cadenas concatenadas con NULL, ya que este comportamiento se ve afectado por la opción **SET CONCAT_NULL_YIELDS_NULL** (*Transact-SQL*).

Antes de meternos a describir ejemplos de Row Level security, hay que entender una cosa que está dentro de los scripts que son las **funciones**:

Funciones

Al igual que las funciones de los lenguajes de programación, las **funciones de SQL Server** son rutinas que aceptan parámetros, realizan una acción, y devuelven el resultado de esa acción como un valor. El valor devuelto puede ser un valor escalar único o un conjunto de resultados. Hay varios tipos de funciones:

- Funciones **definidas por el usuario**: Permiten una programación modular. Se puede crear la función una vez, almacenarla en la base de datos y llamarla desde el programa tantas veces como se deseé. Las funciones definidas por el usuario se pueden modificar, independientemente del código de origen del programa. Características:

Permiten una ejecución más rápida. Al igual que los procedimientos almacenados, las funciones definidas por el usuario Transact-SQL reducen el costo de compilación del código Transact-SQL almacenando los planes en la caché y reutilizándolos para ejecuciones repetidas. Esto significa que no es necesario volver a analizar y optimizar la función definida por el usuario con cada uso, lo que permite obtener tiempos de ejecución mucho más rápidos.

Las funciones CLR ofrecen una ventaja de rendimiento importante sobre las funciones Transact-SQL para tareas de cálculo, manipulación de cadenas y lógica empresarial. Transact-SQL se adecuan mejor a la lógica intensiva del acceso a datos.

Pueden reducir el tráfico de red. Una operación que filtra datos basándose en restricciones complejas que no se puede expresar en una sola expresión escalar se puede expresar como una función. La función se puede invocar luego en la cláusula WHERE para reducir el número de filas que se envían al cliente.

NOTA: Las funciones definidas por el usuario de Transact-SQL en consultas solo se pueden ejecutar en un único subproceso (plan de ejecución en serie). Por tanto, el uso de UDF impide el procesamiento de consultas en paralelo.

Los tipos de funciones de usuario son las siguientes:

✚ **Función escalar:** devuelve un único valor de datos del tipo definido en la cláusula **RETURNS**. En una función escalar insertada, el valor escalar es el resultado de una sola instrucción. Para una función escalar de varias instrucciones, el cuerpo de la función puede contener una serie de instrucciones de **Transact-SQL** que devuelven el único valor. El tipo devuelto puede ser de cualquier tipo de datos **excepto text, ntext, image, cursor y timestamp**.

✚ **Funciones con valores de tabla:** devuelven un tipo de datos table. Las funciones insertadas con valores de tabla no tienen cuerpo; la tabla es el conjunto de resultados de una sola instrucción **SELECT**.

- **Funciones del sistema:** *SQL Server* proporciona numerosas funciones del sistema que se pueden usar para realizar diversas operaciones. No se pueden modificar

\$PARTITION	ERROR_PROCEDURE
@@ERROR	ERROR_SEVERITY
@@IDENTITY	ERROR_STATE
@@PACK_RECEIVED	FORMATMESSAGE
@@ROWCOUNT	GET_FILESTREAM_TRANSACTION_CONTEXT
@@TRANCOUNT	GETANSINULL
BINARY_CHECKSUM	HOST_ID
CHECKSUM	HOST_NAME
COMPRESS	ISNULL
CONNECTIONPROPERTY	ISNUMERIC
CONTEXT_INFO	MIN_ACTIVE_ROWVERSION
CURRENT_REQUEST_ID	NEWID
CURRENT_TRANSACTION_ID	NEWSEQUENTIALID
DECOMPRESS	ROWCOUNT_BIG
ERROR_LINE	SESSION_CONTEXT
ERROR_MESSAGE	SESSION_ID
ERROR_NUMBER	XACT_STATE

Los errores de *Transact-SQL* que producen la cancelación de una instrucción y continúan con la siguiente instrucción del módulo (como desencadenadores o procedimientos almacenados) se tratan de forma distinta dentro de una función. En las funciones, estos errores hacen que se detenga la ejecución de la función. Esto hace que se cancele la función que invocó la instrucción.

Las instrucciones de un bloque **BEGIN...END** no pueden producir efectos secundarios. Los efectos secundarios de una función son cambios definitivos del estado de un recurso que está fuera del ámbito de la función, como una modificación de una tabla de base de datos. Los únicos cambios que pueden realizar las instrucciones de la función son cambios en objetos locales de la función, como cursores o variables locales. En una función no se pueden llevar a cabo algunas acciones como, por ejemplo, modificar tablas de base de datos, realizar operaciones en cursores no locales de la función, enviar correo electrónico, intentar modificar un catálogo o generar un conjunto de resultados que se devuelve al usuario.

NOTA: Si una instrucción **CREATE FUNCTION** genera efectos secundarios sobre recursos que no existen en el momento que se emite la instrucción **CREATE FUNCTION**, SQL Server ejecuta la instrucción. Sin embargo, SQL Server no ejecuta la función cuando ésta se invoca.

El número de veces que se ejecuta realmente una función especificada en una consulta puede variar entre los planes de ejecución generados por el optimizador. Un ejemplo es una función invocada por una subconsulta en una cláusula **WHERE**. El número de veces que se ejecuta la subconsulta y su función puede variar con diferentes rutas de acceso seleccionadas por el optimizador.

Las instrucciones válidas en una función son:

- Las instrucciones **DECLARE** se pueden usar para definir variables y cursores de datos locales de la función.
- La asignación de valores a objetos locales de la función, como el uso de **SET** para asignar valores a variables locales escalares y de tabla.
- Las operaciones de cursores que hacen referencia a cursores locales que están declarados, abiertos, cerrados y no asignados en la función. No se admiten las instrucciones **FETCH** que devuelven datos al cliente. Solo se permiten las instrucciones **FETCH** que asignan valores a variables locales mediante la cláusula **INTO**.
- Instrucciones de control de flujo excepto instrucciones **TRY...CATCH**.
- Instrucciones **SELECT** que contienen listas de selección con expresiones que asignan valores a las variables locales para la función.
- Instrucciones **UPDATE**, **INSERT** y **DELETE** que modifican las variables de tabla locales de la función.
- Instrucciones **EXECUTE** que llaman a un procedimiento almacenado extendido.

Las funciones también pueden ser enlazadas a esquemas. **CREATE FUNCTION** admite una cláusula **SCHEMABINDING** que enlaza la función con el esquema de cualquier objeto al que haga referencia, como tablas, vistas y otras funciones definidas por el usuario. Se producen errores al intentar modificar o quitar objetos a los que hace referencia una función enlazada con un esquema. Para poder especificar **SCHEMABINDING** en **CREATE FUNCTION** se deben cumplir estas condiciones:

- Todas las vistas y las funciones definidas por el usuario a las que hace referencia la función deben estar enlazadas con un esquema.
- Todos los objetos a los que hace referencia la función deben encontrarse en la misma base de datos que la función. Se debe hacer referencia a los objetos mediante nombres de una o dos partes.
- Se debe disponer de permisos **REFERENCES** en todos los objetos (tablas, vistas y funciones definidas por el usuario) a los que hace referencia la función.

Se puede usar **ALTER FUNCTION** para quitar el enlace con el esquema. La instrucción **ALTER FUNCTION** debe volver a definir la función sin especificar **WITH SCHEMABINDING**.

A la hora de especificar parámetros, una función definida por el usuario tiene de cero a varios parámetros de entrada y devuelve un valor escalar o una tabla. Una función puede tener un máximo de **1024 parámetros de entrada**. Cuando un parámetro de la función tiene un valor predeterminado, debe especificarse la palabra clave **DEFAULT** al llamar a la función para poder obtener el valor predeterminado. Este

comportamiento es diferente del de los parámetros con valores predeterminados de procedimientos almacenados definidos por el usuario, para los cuales omitir el parámetro implica especificar el valor predeterminado. Las funciones definidas por el usuario no admiten parámetros de salida.

¿Cuáles son las funciones de base de datos SQL?

- Funciones de agregado. Realizan un cálculo en un conjunto de valores y devuelven un valor único. Se pueden usar en la lista de selección o en la cláusula **HAVING** de una instrucción **SELECT**. Se puede usar una agregación en combinación con la cláusula **GROUP BY** para calcular la agregación en las categorías de filas. Usar la cláusula **OVER** para calcular la agregación en un intervalo de valor específico. La cláusula **OVER** no puede seguir las agregaciones **GROUPING** o **GROUPING_ID**.

Todas las funciones de agregación son **deterministas**; es decir, siempre devuelven el mismo resultado cuando se ejecutan con los mismos valores de entrada.

- Funciones analíticas. Calculan un valor agregado basándose en un grupo de filas. A diferencia de las funciones de agregado, estas funciones pueden devolver varias filas para cada grupo. Se puede usar funciones analíticas para calcular medias móviles, totales acumulados, porcentajes o resultados de N valores superiores dentro de un grupo.
- Funciones de categoría. Devuelven un valor de categoría para cada fila de una partición. Según la función que se utilice, algunas filas pueden recibir el mismo valor que otras. Las funciones de categoría son **no deterministas**.
- Funciones de conjuntos de filas. Devuelven un objeto que se puede usar como referencias de tabla en una instrucción SQL.
- Funciones escalares. Operan sobre un valor y después devuelven otro valor. Las funciones escalares se pueden utilizar donde la expresión sea válida.
- Funciones de texto e imagen. Realizan operaciones sobre los valores de entrada o columnas de texto o imagen, y devuelven información acerca del valor.

Hace un momento, nombraba los términos determinista y no determinista, ¿Qué son? Las funciones son **deterministas** cuando devuelven siempre el mismo resultado cada vez que se llaman con un conjunto específico de valores de entrada. Las funciones son **no deterministas** cuando es posible que devuelvan distintos resultados cada vez que se llaman con un mismo conjunto específico de valores de entrada.

Las funciones que toman una entrada de cadena de caracteres y devuelven una salida de cadena de caracteres utilizan la intercalación de la cadena de entrada para la salida.

Las funciones que toman entradas que no son de caracteres y devuelven una cadena de caracteres utilizan la intercalación predeterminada de la base de datos actual para la salida.

Las funciones que toman varias entradas de cadena de caracteres y devuelven una cadena de caracteres utilizan las reglas de prioridad de intercalación para establecer la intercalación de la cadena de salida.

El siguiente script lo voy a usar para demostrar la creación de una función para que si un cliente pregunte por unos precios estándar, éstos ya estén integrados en una consulta.

Primero creo una tabla en la que quiero que figuren unos precios estándar. Para ello tengo los tipos simple, media y completa a los que van asociados unos precios, y se incluyen en un nombre significativo, que bien podría hacer referencia a las tablas.

```
USE BBM_ASPACE
GO
DROP TABLE IF EXISTS trial.BBM_STANDARDPRICE
GO

CREATE TABLE trial.BBM_STANSARDPRICE
(
Identificador varchar (255),
Tipo varchar (255),
Precio int)
GO
trial.BBM_STANSARDPRICE
```

Inserto unos valores para hacer la demostración.

```
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('COMIDA','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('COMIDA','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('COMIDA','SIMPLE','5')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('HORARIO','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('HORARIO','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('HORARIO','SIMPLE','5')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('ACTIVIDAD','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('ACTIVIDAD','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('ACTIVIDAD','SIMPLE','5')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('SESION','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('SESION','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('SESION','SIMPLE','5')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('TERAPIA','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('TERAPIA','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('TERAPIA','SIMPLE','5')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('DOMAN','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('DOMAN','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE] VALUES('DOMAN','SIMPLE','5')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('MEDICACION','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('MEDICACION','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('MEDICACION','SIMPLE','5')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('TRATAMIENTOS','COMPLETA','15')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('TRATAMIENTOS','MEDIA','10')
INSERT INTO [trial].[BBM_STANSARDPRICE]
VALUES('TRATAMIENTOS','SIMPLE','5')
```

```
SELECT * FROM trial.BBM_STANSARDPRICE
```

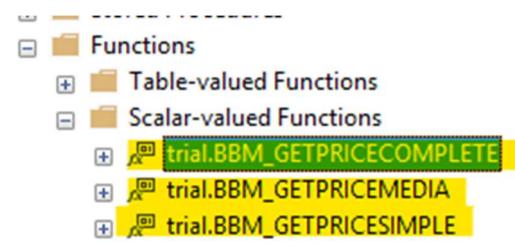
	Identificador	Tipo	Precio
1	COMIDA	COMPLETA	15
2	COMIDA	MEDIA	10
3	COMIDA	SIMPLE	5
4	HORARIO	COMPLETA	15
5	HORARIO	MEDIA	10
6	HORARIO	SIMPLE	5
7	ACTIVIDAD	COMPLETA	15
8	ACTIVIDAD	MEDIA	10
9	ACTIVIDAD	SIMPLE	5
10	SESION	COMPLETA	15
11	SESION	MEDIA	10
12	SESION	SIMPLE	5
13	TERAPIA	COMPLETA	15
14	TERAPIA	MEDIA	10
15	TERAPIA	SIMPLE	5

Compruebo la existencia de las funciones que quiero crear.

```
/*CHECK THE FUNCTION EXISTENCE*/
DROP FUNCTION IF EXISTS trial.BBM_GETPRICESIMPLE
DROP FUNCTION IF EXISTS trial.BBM_GETPRICEMEDIA
DROP FUNCTION IF EXISTS trial.BBM_GETPRICECOMPLETE
```

Creo la función con una variable @GROSSPRICE (precio bruto), y otra @finalprice (precio neto) al que se le aplica el IVA (21%). En el `BEGIN ... END` le indico que el precio neto es la suma de todos los precios donde el precio es ‘valor del Tipo’. Cuando le indico que haga `RETURN`, le pido que me sume el precio final + el precio bruto con el IVA aplicado.

```
/*CREATE FUNCTION FOR STANDARD PRICE SIMPLE PLAN*/
CREATE FUNCTION trial.BBM_GETPRICESIMPLE
(
@GROSSPRICE INT,
@finalprice int
)
RETURNS int
AS
--returns the monthlybill
BEGIN
    SET @finalprice = (SELECT SUM (p.Precio)
    from trial.BBM_STANSARDPRICE p
    where p.Tipo = 'SIMPLE');
    SET @GROSSPRICE = @finalprice;
    RETURN @finalprice + (@GROSSPRICE * 0.21);
END;
```



```

/*CREATE FUNCTION FOR STANDARD PRICE MEDIA PLAN*/
CREATE FUNCTION trial.BBM_GETPRICEMEDIA
(
@GROSSPRICE INT,
@finalprice int
)
RETURNS int
AS
--returns the monthlybill
BEGIN
    SET @finalprice = (SELECT SUM (p.Precio)
    from trial.BBM_STANSARDPRICE p
    where p.Tipo = 'MEDIA');
    SET @GROSSPRICE = @finalprice;
    RETURN @finalprice + (@GROSSPRICE * 0.21);
END;

/*CREATE FUNCTION FOR STANDARD PRICE COMPLETA PLAN*/
CREATE FUNCTION trial.BBM_GETPRICECOMPLETE
(
@GROSSPRICE INT,
@finalprice int
)
RETURNS int
AS
--returns the monthlybill
BEGIN
    SET @finalprice = (SELECT SUM (p.Precio)
    from trial.BBM_STANSARDPRICE p
    where p.Tipo = 'COMPLETA');
    SET @GROSSPRICE = @finalprice;
    RETURN @finalprice + (@GROSSPRICE * 0.21);
END;

```

Como final, ejecuto las funciones para comprobar que funcionan:

```

/*RESULTS*/
PRINT trial.BBM_GETPRICESIMPLE (' ', ' ')
48
PRINT trial.BBM_GETPRICEMEDIA (' ', ' ')
96
PRINT trial.BBM_GETPRICECOMPLETE (' ', ' ')
145

```

Ahora que hemos visto las funciones y he descrito lo que es RLS, voy a dejar descrito un [script demostrando la función de RLS en mi base](#):

Nos ubicamos en la base de datos para crear dos tablas (debido a un BUG tengo que crear ambas y no puedo aplicar `SELECT INTO`).

```
USE BBM_ASPACE
```

```
/*CREAMOS TABLA PARA SUPUESTO, DONDE SE VA A HACER ASIGNACIÓN DE
PACIENTES DIARIOS POR PROFESIONAL Y AUXILIAR*/
```

Creamos la tabla de **profesionales** y de **auxiliares**. Insertamos valores duplicados para la demostración (más adelante, trato de resolver esto con un *trigger*):

```
/*TABLA PROS*/
DROP TABLE IF EXISTS trial.BBM_USERDATASSIGNEDPROS

CREATE TABLE trial.BBM_USERDATASSIGNEDPROS
(
    Usuario varchar (20) NOT NULL,
    Nombre_Usuario varchar(10) NULL,
    Pro_asignado Varchar(30) NULL,
    Aux_asignado varchar(30)
)
GO
trial.BBM_USERDATASSIGNEDPROS

/*INSERTAMOS VALORES A LA TABLA*/

INSERT INTO trial.BBM_USERDATASSIGNEDPROS VALUES
('USER01','Pepe','PSICOLOGO','AUX01'),
('USER02','Paco','LOGOPEDA','AUX02'),
('USER03','Lola','ASISTENTE_SOCIAL','AUX03'),
('USER04','Diana','TERAPEUTA_OCPACIONAL','AUX04'),
('USER05','Jose','FISIOTERAPEUTA','AUX05'),
('USER06','Luis','PSICOLOGO','AUX05'),
('USER07','Julia','LOGOPEDA','AUX04'),
('USER08','Cristina','ASISTENTE_SOCIAL','AUX03'),
('USER08','Toribio','TERAPEUTA_OCPACIONAL','AUX02'),
('USER00','Brigida','FISIOTERAPEUTA','AUX01');
GO
--(10 rows affected)
```

```
SELECT * FROM trial.BBM_USERDATASSIGNEDPROS
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER01	Pepe	PSICOLOGO	AUX01
2	USER02	Paco	LOGOPEDA	AUX02
3	USER03	Lola	ASISTENTE_SOCIAL	AUX03
4	USER04	Diana	TERAPEUTA_OCPACIONAL	AUX04
5	USER05	Jose	FISIOTERAPEUTA	AUX05
6	USER06	Luis	PSICOLOGO	AUX05
7	USER07	Julia	LOGOPEDA	AUX04
8	USER08	Cristina	ASISTENTE_SOCIAL	AUX03
9	USER08	Toribio	TERAPEUTA_OCPACIONAL	AUX02
10	USER00	Brigida	FISIOTERAPEUTA	AUX01

```
/*POR UN BUG CON LOS PERMISOS TENGO QUE DUPLICAR LA TABLA EN VEZ DE
SELECT * INTO*/
DROP TABLE IF EXISTS trial.BBM_USERDATASSIGNEDAUX
```

```
CREATE TABLE trial.BBM_USERDATASSIGNEDAUX
(
    Usuario varchar (20) NOT NULL,
    Nombre_Usuario varchar(10) NULL,
    Pro_asignado Varchar(30) NULL,
    Aux_asignado varchar(30)
)
GO
trial.BBM_USERDATASSIGNEDAUX
```

```

/*INSERTAMOS VALORES A LA TABLA*/

INSERT INTO trial.BBM_USERDATASSIGNEDAUX VALUES
    ('USER01','Pepe','PSICOLOGO','AUX01'),
    ('USER02','Paco','LOGOPEDA','AUX02'),
    ('USER03','Lola','ASISTENTE_SOCIAL','AUX03'),
    ('USER04','Diana','TERAPEUTA_OCUPACIONAL','AUX04'),
    ('USER05','Jose','FISIOTERAPEUTA','AUX05'),
    ('USER06','Luis','PSICOLOGO','AUX05'),
    ('USER07','Julia','LOGOPEDA','AUX04'),
    ('USER08','Cristina','ASISTENTE_SOCIAL','AUX03'),
    ('USER08','Toribio','TERAPEUTA_OCUPACIONAL','AUX02'),
    ('USER00','Brigida','FISIOTERAPEUTA','AUX01');

```

GO

--(10 rows affected)

```
SELECT * FROM trial.BBM_USERDATASSIGNEDAUX
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER01	Pepe	PSICOLOGO	AUX01
2	USER02	Paco	LOGOPEDA	AUX02
3	USER03	Lola	ASISTENTE_SOCIAL	AUX03
4	USER04	Diana	TERAPEUTA_OCUPACIONAL	AUX04
5	USER05	Jose	FISIOTERAPEUTA	AUX05
6	USER06	Luis	PSICOLOGO	AUX05
7	USER07	Julia	LOGOPEDA	AUX04
8	USER08	Cristina	ASISTENTE_SOCIAL	AUX03
9	USER08	Torbio	TERAPEUTA_OCUPACIONAL	AUX02
10	USER00	Brigida	FISIOTERAPEUTA	AUX01

Creamos los usuarios y los añadimos a un rol que creamos

```
/*CREAMOS LOS PROS*/
```

```

CREATE USER PSICOLOGO WITHOUT LOGIN;
CREATE USER LOGOPEDA WITHOUT LOGIN;
CREATE USER ASISTENTE_SOCIAL WITHOUT LOGIN;
CREATE USER TERAPEUTA_OCUPACIONAL WITHOUT LOGIN;
CREATE USER FISIOTERAPEUTA WITHOUT LOGIN;

```

```
/*CREAMOS LOS AUXILIARES*/
```

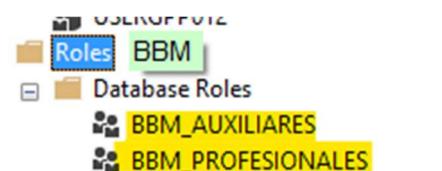
```

CREATE USER AUX01 WITHOUT LOGIN;
CREATE USER AUX02 WITHOUT LOGIN;
CREATE USER AUX03 WITHOUT LOGIN;
CREATE USER AUX04 WITHOUT LOGIN;
CREATE USER AUX05 WITHOUT LOGIN;

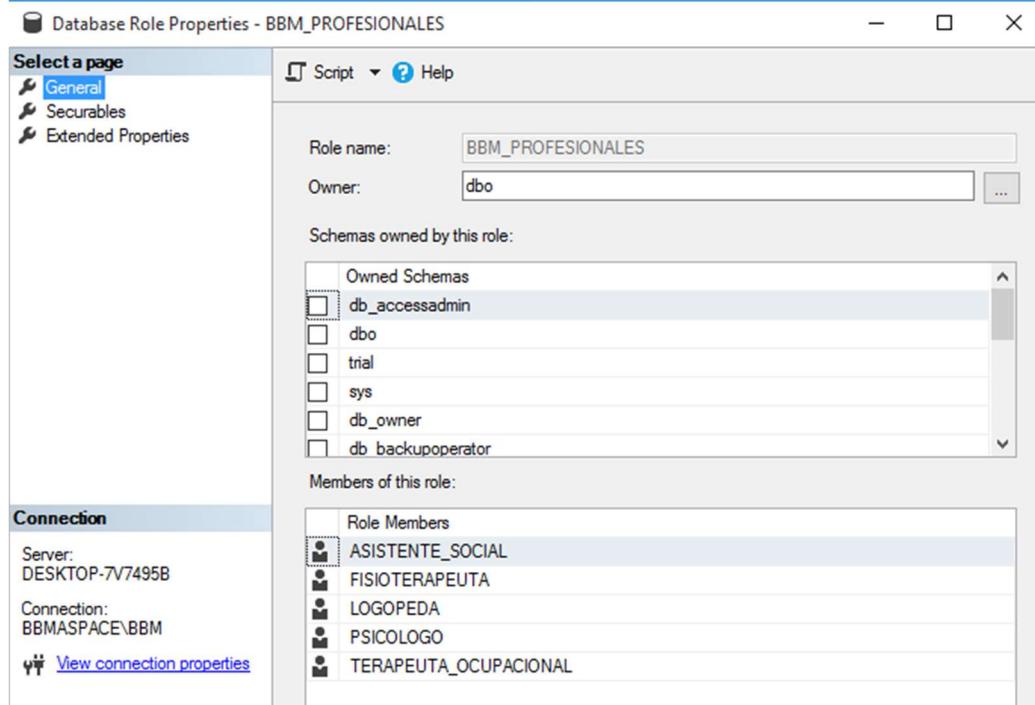
```

```
/*CREAMOS EL ROL DE LOS PROS*/
CREATE ROLE BBM_PROFESIONALES
```

```
/*CREAMOS EL ROL DE LOS AUXILIARES*/
CREATE ROLE BBM_AUXILIARES
```



```
/*AÑADIMOS USUARIOS AL ROL DE LOS PROS*/
ALTER ROLE BBM_PROFESIONALES ADD MEMBER PSICOLOGO
ALTER ROLE BBM_PROFESIONALES ADD MEMBER LOGOPEDA
ALTER ROLE BBM_PROFESIONALES ADD MEMBER ASISTENTE_SOCIAL
ALTER ROLE BBM_PROFESIONALES ADD MEMBER TERAPEUTA_OCUPACIONAL
ALTER ROLE BBM_PROFESIONALES ADD MEMBER FISIOTERAPEUTA
```



```
/*AÑADIMOS USUARIOS AL ROL DE LOS AUXILIARES*/
ALTER ROLE BBM_AUXILIARES ADD MEMBER AUX01
ALTER ROLE BBM_AUXILIARES ADD MEMBER AUX02
ALTER ROLE BBM_AUXILIARES ADD MEMBER AUX03
ALTER ROLE BBM_AUXILIARES ADD MEMBER AUX04
ALTER ROLE BBM_AUXILIARES ADD MEMBER AUX05
```

Database Role Properties - BBM_AUXILIARES

Select a page:

Role name: BBM_AUXILIARES

Owner: dbo

Schemas owned by this role:

Owned Schemas
<input type="checkbox"/> db_accessadmin
<input type="checkbox"/> dbo
<input type="checkbox"/> trial
<input type="checkbox"/> sys
<input type="checkbox"/> db_owner
<input type="checkbox"/> db_backupoperator

Members of this role:

Role Members
AUX01
AUX02
AUX03
AUX04
AUX05

Connection:

Server: DESKTOP-7V7495B

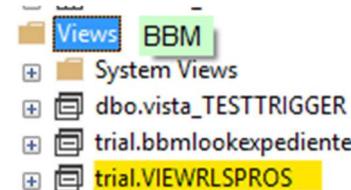
Connection: BBMASPACE\BBM

View connection properties

Como lo que intento con este proyecto es que haya una persona que asigne pacientes a profesionales y auxiliares, lo que hago es genera una vista para cada uno de los roles. Primero con los profesionales para probar:

```
USE BBM_ASPACE
GO

DROP VIEW IF EXISTS trial.VIEWRLSPROS
```



```
CREATE VIEW trial.VIEWRLSPROS WITH SCHEMABINDING AS
SELECT Usuario, Nombre_Usuario, Pro_asignado
FROM trial.BBM_USERDATASSIGNEDPROS
```

```
PRINT USER
SELECT * FROM trial.VIEWRLSPROS
```

	Usuario	Nombre_Usuario	Pro_asignado
1	USER01	Pepe	PSICOLOGO
2	USER02	Paco	LOGOPEDA
3	USER03	Lola	ASISTENTE_SOCIAL
4	USER04	Diana	TERAPEUTA_OCUPACIONAL
5	USER05	Jose	FISIOTERAPEUTA
6	USER06	Luis	PSICOLOGO
7	USER07	Julia	LOGOPEDA
8	USER08	Cristina	ASISTENTE_SOCIAL
9	USER08	Torbio	TERAPEUTA_OCUPACIONAL
10	USER00	Brigida	FISIOTERAPEUTA

```
/*OTORGUE PERMISOS AL ROL*/
GRANT SELECT ON [trial].[VIEWRLSPROS] TO [BBM_PROFESIONALES]
GO
```

Database Role Properties - BBM_PROFESIONALES

Select a page: General, Securables, Extended Properties

Script Help

Database role name: BBM_PROFESIONALES

Securables:

Schema	Name	Type
trial	VIEWRLSPROS	View

Search...

```

PRINT USER
EXECUTE AS USER = 'LOGOPEDA'
PRINT USER
--LOGOPEDA
SELECT * FROM trial.VIEWRLSPROS

```

	Usuario	Nombre_Usuario	Pro_asignado
1	USER01	Pepe	PSICOLOGO
2	USER02	Paco	LOGOPEDA
3	USER03	Lola	ASISTENTE_SOCIAL
4	USER04	Diana	TERAPEUTA_OCPACIONAL
5	USER05	Jose	FISIOTERAPEUTA
6	USER06	Luis	PSICOLOGO
7	USER07	Julia	LOGOPEDA
8	USER08	Cristina	ASISTENTE_SOCIAL
9	USER08	Toribio	TERAPEUTA_OCPACIONAL
10	USER00	Brigida	FISIOTERAPEUTA

```

REVERT
PRINT USER

```

En vista de que funciona, creo una función para que me devuelva en la consulta el usuario que está conectado haciendo la consulta. Creamos una política de seguridad sobre la vista para que me devuelva sólo el parámetro que le estoy pidiendo:

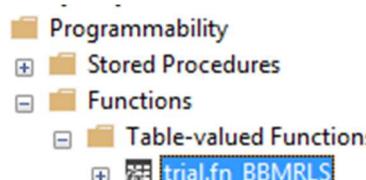
```

USUARIO*/
DROP Function IF Exists trial.fn_BBMRLS
GO
CREATE OR ALTER FUNCTION trial.fn_BBMRLS
(@FILTERCOL sysname) --> busca en sysname
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN SELECT 1 as fn_BBMELSECUREDATA
-- filter out records based on database user name
where @FILTERCOL = user_name(); --> Devuelve el usuario conectado
GO

/*CREAMOS POLITICA DE SEGURIDAD DE PROFESIONALES*/
DROP SECURITY POLICY IF EXISTS trial.BBMFILTROASIGNACIONESPROS

CREATE SECURITY POLICY trial.BBMFILTROASIGNACIONESPROS
ADD FILTER PREDICATE trial.fn_BBMRLS(Pro_asignado) --> add filter
predicate + schema.nombre_funcion + (campo que toma como parametro)
ON trial.VIEWRLSPROS --> Sobre la tabla
WITH (STATE = ON); --> Con la política activada para que comience a
funcionar y
GO
/*DDB -> SECURITY -> SECURITY POLICIES*/

```



☰ Security Policies
BBMFILTROASIGNACIONESPROS

Comprobamos

```
/*HAGO UNA PRUEBA CON UN PROFESIONAL*/
-- PRIMERO PROFESIONALES
PRINT USER
EXECUTE AS USER = 'LOGOPEDA'
```

```
PRINT USER
--LOGOPEDA
```

```
SELECT * FROM trial.VIEWRLSPROS
```

	Usuario	Nombre_Usuario	Pro_asignado
1	USER02	Paco	LOGOPEDA
2	USER07	Julia	LOGOPEDA

```
REVERT
```

```
PRINT USER
--dbo
```

```
/*HAGO UNA RÉPLICA CON LOS AUXILIARES*/
USE BBM_ASPACE
GO
DROP VIEW IF EXISTS trial.VIEWRLSAUX
GO
CREATE VIEW trial.VIEWRLSAUX WITH
SCHEMABINDING AS
SELECT Usuario, Nombre_Usuario, Aux_asignado
FROM trial.BBM_USERDATASSIGNEDAUX
```

- ☰ Views
 - + System Views
 - + dbo.vista_TESTTRIGGER
 - + trial.bbmlookexpediente
 - + trial.VIEWRLSAUX

```
PRINT USER
```

```
SELECT * FROM trial.VIEWRLSAUX
```

	Usuario	Nombre_Usuario	Aux_asignado
1	USER01	Pepe	AUX01
2	USER02	Paco	AUX02
3	USER03	Lola	AUX03
4	USER04	Diana	AUX04
5	USER05	Jose	AUX05
6	USER06	Luis	AUX05
7	USER07	Julia	AUX04
8	USER08	Cristina	AUX03
9	USER08	Toribio	AUX02
10	USER00	Brigida	AUX01

```
/*OTORGO PERMISOS AL ROL*/
```

```
GRANT SELECT ON [trial].[VIEWRLSAUX] TO [BBM_AUXILIARES]
GO
```

Database Role Properties - BBM_AUXILIARES

Select a page

- General
- Securables**
- Extended Properties

Script ▾ Help

Database role name: BBM_AUXILIARES

Securables:

	Schema	Name	Type
<input checked="" type="checkbox"/>	trial	VIEWRLSAUX	View

Search...

```

PRINT USER
EXECUTE AS USER = 'AUX04'
PRINT USER
--AUX04

SELECT * FROM trial.VIEWRLSAUX

```

	Usuario	Nombre_Usuario	Aux_asignado
1	USER01	Pepe	AUX01
2	USER02	Paco	AUX02
3	USER03	Lola	AUX03
4	USER04	Diana	AUX04
5	USER05	Jose	AUX05
6	USER06	Luis	AUX05
7	USER07	Julia	AUX04
8	USER08	Cristina	AUX03
9	USER08	Toribio	AUX02
10	USER00	Brigida	AUX01

```

REVERT

PRINT USER

```

La causa de que haya una segunda tabla cuando las dos hacen lo mismos es que una tabla sólo puede tener una política de seguridad, por tanto, hago una réplica en auxiliares.

```

/*CREAMOS UNA FUNCIÓN QUE BUSQUE EN SYSNAME Y ME DEVUELVA EL NOMBRE DE
USUARIO*/
DROP Function IF Exists trial.fn_BBMRLSAUX
GO
CREATE OR ALTER FUNCTION trial.fn_BBMRLSAUX
(@FILTERCOL sysname) --> busca en sysname
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN SELECT 1 as fn_BBMLSECUREDATA
-- filter out records based on database user name
where @FILTERCOL = user_name(); --> Devuelve el usuario conectado
GO

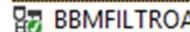
/*CREAMOS POLITICA DE SEGURIDAD DE AUXILIAR*/
DROP SECURITY POLICY IF EXISTS trial.BBMFILTROASIGNACIONESAUX

CREATE SECURITY POLICY trial.BBMFILTROASIGNACIONESAUX

```



```

ADD FILTER PREDICATE trial.fn_BBMRLSAUX(Aux_asignado) --> add filter
predicate + schema.nombre_funcion + (campo que toma como parametro)
ON trial.VIEWRLSAUX --> Sobre la tabla
WITH (STATE = ON); --> Con la politica activada para que comience a
funcionar y
GO
/*DDB -> SECURITY -> SECURITY POLICIES*/
 Security Policies
 BBMFILTROASIGNACIONESAUX

/*HAGO UNA PRUEBA CON UN AUXILIAR*/
-- PRIMERO AUXILIAR
PRINT USER

EXECUTE AS USER = 'AUX04'

PRINT USER
--AUX04

SELECT * FROM trial.VIEWRLSAUX


|   | Usuario | Nombre_Usuario | Aux_asignado |
|---|---------|----------------|--------------|
| 1 | USER04  | Diana          | AUX04        |
| 2 | USER07  | Julia          | AUX04        |


REVERT

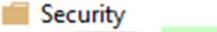
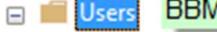
PRINT USER
--dbo

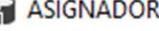
```

Para la parte de **insertar**, creo un usuario asignador al cual le concedo permisos se SELECT, UPDATE e INSERT. Después meto un trigger

```

/*PARA DARLE VERACIDAD VAMOS A DAR PERMISOS DE ACTUALIZACIÓN SOLAMENTE
SOBRE UNA TABLA, Y A UN NUEVO USUARIO ADMINISTRATIVO, QUE ES EL QUE
ASIGNA LOS PACIENTES*/

--CREO EL USUARIO
CREATE USER ASIGNADOR WITHOUT LOGIN;




-- Granting UPDATE and INSERT access A LOS PROS
GRANT SELECT ON [trial].[BBM_USERDATASSIGNEDPROS] TO [ASIGNADOR]
GO
GRANT UPDATE, INSERT ON trial.BBM_USERDATASSIGNEDPROS TO ASIGNADOR;
GO

```

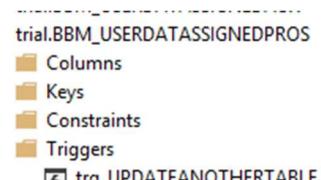
Trigger recuperación primera evaluación

Lo que se trata en este trigger es que si yo actualizo la tabla de profesionales, también se actualice la tabla de auxiliares. Y si yo inserto una nueva fila en la tabla de profesionales, que también se inserte en la de auxiliares:

```
/*PROBAMOS CON INSERTAR*/
```

```
/*CON UN TRIGGER, HACEMOS QUE LOS DATOS QUE SE INSERTAN EN LA TABLA DE PROFESIONALES, SE REPLICAN EN LA DE AUXILIARES*/
DROP TRIGGER IF EXISTS trial.trg_UPDATEANOTHERTABLE
GO

CREATE OR ALTER TRIGGER
trial.trg_UPDATEANOTHERTABLE ON
[trial].[BBM_USERDATASSIGNEDPROS]
AFTER UPDATE, INSERT
AS
BEGIN
    --SET NOCOUNT ON agregado para evitar conjuntos de resultados adicionales
    -- interferir con las instrucciones SELECT.
    SET NOCOUNT ON;
```



```
-- OBTENER LOS VALORES DE LOS CAMPOS
DECLARE @R int, @U VARCHAR(20), @NU VARCHAR(10), @PA VARCHAR(30),
@AA VARCHAR(30)
SELECT @R = new.ROW#           FROM inserted new;
SELECT @U = new.Usuario        FROM inserted new;
SELECT @NU = new.Nombre_Usuario FROM inserted new;
SELECT @PA = new.Pro_asignado   FROM inserted new;
SELECT @AA = new.Aux_asignado   FROM inserted new;

/*ACTUALIZAR PROS -> ACTUALIZA AUX*/
IF (UPDATE(Usuario) OR UPDATE(Nombre_Usuario) OR
UPDATE(Pro_asignado) OR UPDATE(Aux_asignado)) /*COGE ESTOS DATOS DE LA
TABLA TEST*/
BEGIN
    INSERT INTO trial.BBM_USERDATASSIGNEDAUX (Usuario,
Nombre_Usuario, Pro_asignado, Aux_asignado) VALUES (@U, @NU, @PA, @AA);
    DELETE FROM trial.BBM_USERDATASSIGNEDAUX WHERE ROW# = @R;
    PRINT 'THE EMPLOYEE HAS BEEN UPDATED/INSERTED
SUCCESSFULLY'
END
END
```

Demostramos el funcionamiento.

```
/*DEMOSTRAMOS EL FUNCIONAMIENTO/

PRINT USER
GO
-- dbo

SELECT * FROM trial.BBM_USERDATASSIGNEDPROS
GO
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER01	Pepe	PSICOLOGO	AUX01
2	USER02	Paco	LOGOPEDA	AUX02
3	USER03	Lola	ASISTENTE_SOCIAL	AUX03
4	USER04	Diana	TERAPEUTA_OCUPACIONAL	AUX04
5	USER05	Jose	FISIOTERAPEUTA	AUX05
6	USER06	Luis	PSICOLOGO	AUX05
7	USER07	Julia	LOGOPEDA	AUX04
8	USER08	Cristina	ASISTENTE_SOCIAL	AUX03
9	USER08	Torbio	TERAPEUTA_OCUPACIONAL	AUX02
10	USER00	Brigida	FISIOTERAPEUTA	AUX01

```
-- UPDATE USUARIO QUE SE ASIGNA
EXECUTE AS USER = 'ASIGNADOR';
GO
PRINT USER
```

```
SELECT * FROM trial.BBM_USERDATASSIGNEDPROS
WHERE Nombre_Usuario = 'Brigida'
GO
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER00	Brigida	FISIOTERAPEUTA	AUX01

```

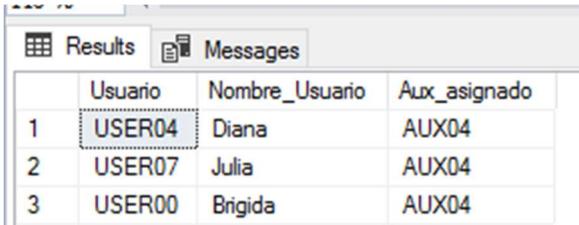
/*PROBAMOS A ACTUALIZAR UNA FILA, PORQUE HA CAMBIADO UN PACIENTE DE
AUXILIAR*/
UPDATE trial.BBM_USERDATASSIGNEDPROS
SET Aux_asignado = 'AUX04'
WHERE Nombre_Usuario = 'Brigida'
GO
-- (1 row affected)

REVERT
PRINT USER

EXECUTE AS USER = 'AUX04';

PRINT USER
--AUX04

/*TRATAMOS DE VER LA ACTUALIZACIÓN EN LA TABLA DE AUXILIARES PARA VER
QUE EL TRIGGER SE DISPARA*/

SELECT * FROM trial.VIEWRLSAUX
GO

REVERT

PRINT USER
--dbo

```

A mí me interesa que se **bloqueen los predicados** y que la actualización la haga solo el trigger. Como se verá más adelante, la *policy* bloquea todo. Altero la *security policy* y demuestro:

```

/*BLOCK PREDICATES*/

/*PARA ESTA PRUEBA, A MÍ ME INTERESA QUE SOBRE LA TABLA AUXILIARES NADIE
ACTUALICE NADA, Y TAN SOLO LA ACTUALIZACIÓN LA HAGA EL TRIGGER*/

ALTER SECURITY POLICY trial.BBMFILTROASIGNACIONESAUX
ADD BLOCK PREDICATE trial.fn_BBMRLSAUX (Aux_asignado)
ON trial.BBM_USERDATASSIGNEDAUX AFTER UPDATE,
ADD BLOCK PREDICATE trial.fn_BBMRLSAUX(Aux_asignado)
ON trial.BBM_USERDATASSIGNEDAUX AFTER INSERT;
GO
/*CON ESTO NO SE DEBERÍA DE PODER HACER NADA EN ESA TABLA EN CUANTO A
INSERCIÓNES */

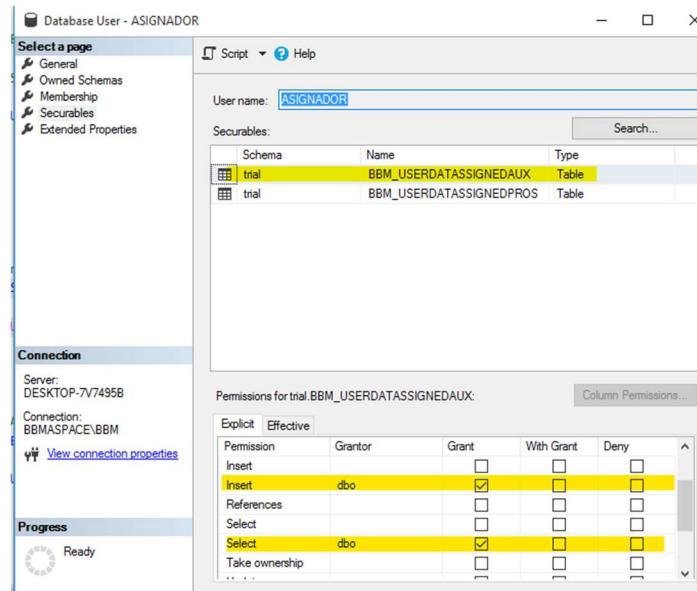
/*DEMOSTRAMOS EL FUNCIONAMIENTO*/
PRINT USER
GO
-- dbo

```

```
SELECT * FROM trial.BBM_USERDATASSIGNEDAUX
GO
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER01	Pepe	PSICOLOGO	AUX01
2	USER02	Paco	LOGOPEDA	AUX02
3	USER03	Lola	ASISTENTE_SOCIAL	AUX03
4	USER04	Diana	TERAPEUTA_OCPACIONAL	AUX04
5	USER05	Jose	FISIOTERAPEUTA	AUX05
6	USER06	Luis	PSICOLOGO	AUX05
7	USER07	Julia	LOGOPEDA	AUX04
8	USER08	Cristina	ASISTENTE_SOCIAL	AUX03
9	USER08	Toribio	TERAPEUTA_OCPACIONAL	AUX02
10	USER00	Brigida	FISIOTERAPEUTA	AUX01
11	USER00	Brigida	FISIOTERAPEUTA	AUX04

```
-- Granting UPDATE and INSERT access A LOS AUX
GRANT SELECT ON [trial].[BBM_USERDATASSIGNEDAUX] TO [ASIGNADOR]
GO
GRANT UPDATE, INSERT ON trial.BBM_USERDATASSIGNEDAUX TO ASIGNADOR;
GO
```



```
-- UPDATE USUARIO QUE SE ASIGNA
EXECUTE AS USER = 'ASIGNADOR';
GO
PRINT USER
```

```
SELECT * FROM trial.BBM_USERDATASSIGNEDAUX
GO
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER01	Pepe	PSICOLOGO	AUX01
2	USER02	Paco	LOGOPEDA	AUX02
3	USER03	Lola	ASISTENTE_SOCIAL	AUX03
4	USER04	Diana	TERAPEUTA_OCPACIONAL	AUX04
5	USER05	Jose	FISIOTERAPEUTA	AUX05
6	USER06	Luis	PSICOLOGO	AUX05
7	USER07	Julia	LOGOPEDA	AUX04
8	USER08	Cristina	ASISTENTE_SOCIAL	AUX03
9	USER08	Toribio	TERAPEUTA_OCPACIONAL	AUX02
10	USER00	Brigida	FISIOTERAPEUTA	AUX01
11	USER00	Brigida	FISIOTERAPEUTA	AUX04

```
/*PROBAMOS A ACTUALIZAR UNA FILA, PORQUE HA CAMBIADO UN PACIENTE DE
AUXILIAR*/
UPDATE trial.BBM_USERDATASSIGNEDAUX
SET Aux_asignado = 'AUX05'
WHERE Nombre_Usuario = 'Brigida';
GO
```

Messages
Msg 33504, Level 16, State 1, Line 416
The attempted operation failed because the target object 'BBM_ASPACE.trial.BBM_USERDATASSIGNEDAUX' has a block predicate that conflicts with this operation. If t
The statement has been terminated.

```
REVERT
```

```
PRINT USER
```

```
/*TRATAMOS DE VER LA ACTUALIZACIÓN EN LA TABLA DE AUXILIARES PARA VER
QUE NO HAY ACTUALIZACIÓN*/
```

```
SELECT * FROM trial.BBM_USERDATASSIGNEDAUX
WHERE Aux_asignado = 'AUX05'
GO
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER05	Jose	FISIOTERAPEUTA	AUX05
2	USER06	Luis	PSICOLOGO	AUX05

```
/*INSERTAMOS EN LA TABLA PROS Y VEMOS LOS EFECTOS*/
```

```
EXECUTE AS USER = 'ASIGNADOR';
GO
PRINT USER
```

```
SELECT * FROM trial.BBM_USERDATASSIGNEDAUX
WHERE Aux_asignado = 'AUX05'
GO
```

	Usuario	Nombre_Usuario	Pro_asignado	Aux_asignado
1	USER05	Jose	FISIOTERAPEUTA	AUX05
2	USER06	Luis	PSICOLOGO	AUX05

```
/*PROBAMOS A INSERTAR UNA FILA FICTICIA*/
```

```
INSERT INTO trial.BBM_USERDATASSIGNEDPROS VALUES
('TEST','DEMOSTRAR','INSERT','trigger')
GO
```

Msg 33504, Level 16, State 1, Procedure trg_UPDATEANOTHERTABLE, Line 17 [Batch Start Line 451]
The attempted operation failed because the target object 'BBM_ASPACE.trial.BBM_USERDATASSIGNEDAUX' has a block predicate that conflicts with this operation. If t
The statement has been terminated.

```
--VEMOS QUE FALLA TAMBIÉN EL TRIGGER
```

```
REVERT
PRINT USER
```

Always Encrypted

Always Encrypted es una nueva característica de seguridad que se introdujo en SQL Server 2016. *Always Encrypted* es una tecnología para garantizar que los datos almacenados en una base de datos permanezcan encriptados en todo momento durante el procesamiento de consultas de SQL Server. *Always Encrypted* permite a los clientes cifrar datos confidenciales, como números de tarjetas de crédito y números de identificación nacional, dentro de la aplicación del cliente y nunca revelar la clave de cifrado al motor de la base de datos.

Como resultado, *Always Encrypted* proporciona separación entre quienes poseen los datos y quienes los administran. Incluso los administradores de bases de datos, administradores de sistemas y administradores de la nube no pueden acceder a los datos. *Always Encrypted* es un cifrado del lado del cliente y hace que el cifrado sea transparente para la aplicación. Los datos se cifran de forma transparente dentro de un controlador de cliente y el cliente administra la clave de cifrado. La clave se puede almacenar en el almacén de certificados de Windows o en Azure Key Vault.

Always Encrypted admite dos tipos de cifrado: **aleatorio** y **determinista**:

El cifrado **aleatorio** utiliza un método que cifra los datos de una manera menos predecible. Genera un valor cifrado diferente para el mismo texto sin formato cada vez. Es más seguro, pero evita búsquedas de igualdad, agrupación, indexación y uniones en columnas cifradas.

El cifrado **determinista** siempre genera el mismo valor cifrado para cualquier valor de texto sin formato determinado. El uso de cifrado determinista permite realizar búsquedas de igualdad, agrupar, filtrar por igualdad y unir tablas en función del valor cifrado. Sin embargo, también puede permitir que usuarios no autorizados adivinen información sobre valores cifrados mediante el uso de patrones en la columna cifrada.

Con *Always Encrypted*, la aplicación cliente maneja el cifrado y descifrado de datos real fuera de SQL Server. Para cifrar y descifrar, la aplicación de datos debe utilizar un controlador *Always Encrypted Enabled* que interactúe con SQL Server.

Para implementar *Always Encrypted* en una columna, es necesario generar una **Clave de cifrado de columna (CEK)** y una **Clave maestra de columna (CMK)**. El CEK se utiliza para proteger / cifrar los datos de la columna, mientras que el CMK se utiliza para almacenar y proteger / cifrar el CEK. El CEK se almacena en el motor de base de datos de SQL Server. Para la Clave maestra de columna, el motor de base de datos solo almacena metadatos que apuntan a la ubicación de la clave. La clave maestra real se almacena en un almacén de claves externo confiable.

Los escenarios típicos en los que se utiliza son:

Cliente y datos locales → Un cliente tiene una aplicación cliente y SQL Server , que se ejecutan localmente, en su ubicación de la empresa. El cliente desea contratar a un proveedor externo para administrar SQL Server. Para proteger la información confidencial almacenada en SQL Server, el cliente utiliza *Always Encrypted* para garantizar la separación de obligaciones entre los administradores de base de datos y los de aplicación. El cliente almacena valores de texto no cifrado de claves de *Always Encrypted* en un almacén de claves de confianza al que puede acceder la aplicación

cliente. Los administradores de SQL Server no tienen acceso a las claves y, por lo tanto, no pueden descifrar la información confidencial almacenada en SQL Server.

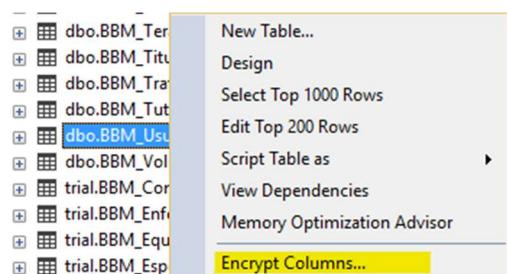
Cliente local con datos de Azure → Un cliente tiene una aplicación cliente local en su ubicación de la empresa. La aplicación trabaja sobre la información confidencial almacenada en una base de datos hospedada en Azure (*SQL Database* o *SQL Server* que se ejecutan en una máquina virtual en *Microsoft Azure*). El cliente usa *Always Encrypted* y almacena las claves de *Always Encrypted* en un almacén de claves de confianza hospedado en local, para garantizar que los administradores de nube de Microsoft no tengan acceso a la información confidencial.

Cliente y datos en Azure → Un cliente tiene una aplicación cliente hospedada en *Microsoft Azure* (por ejemplo, en un rol de trabajo o un rol web) que trabaja sobre la información confidencial almacenada en una base de datos hospedada en Azure (*SQL Database* o *SQL Server* se ejecutan en una máquina virtual en *Microsoft Azure*). Aunque *Always Encrypted* no proporciona un aislamiento completo de los datos ante los administradores de la nube, dado que tanto los datos como las claves están expuestos a los administradores de la nube de la plataforma que hospeda el nivel de cliente, el cliente se sigue beneficiando de la reducción del área expuesta a ataques de seguridad (los datos siempre se cifran en la base de datos).

Always Encrypted es una tecnología de cifrado del lado del cliente en la que el controlador del cliente de SQL Server hace el trabajo. Básicamente, trasladamos el cifrado / descifrado fuera del servidor SQL, moviendo las claves y el cifrado / descifrado de datos al nivel de la aplicación. *Always Encrypted* se basa en una aplicación del lado del cliente para cifrar y descifrar automáticamente los datos confidenciales.

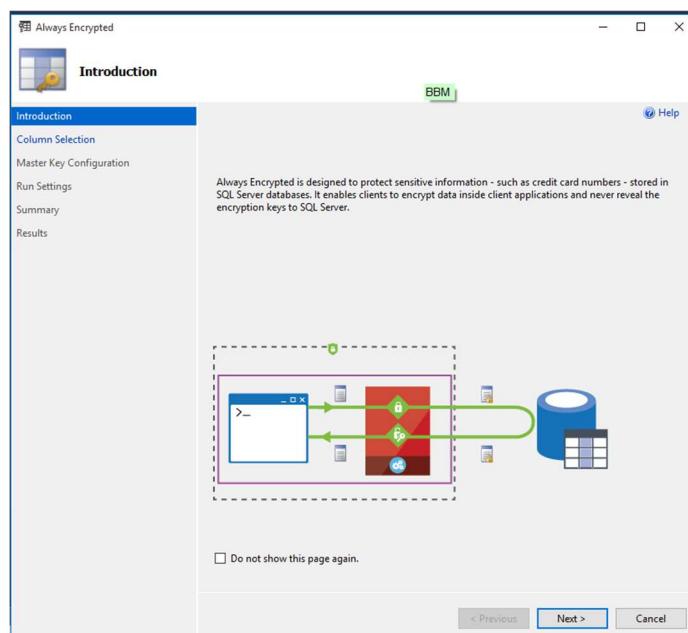


Esto implica un controlador especial que cifra los datos en columnas confidenciales antes de pasarlo al motor de base de datos y reescribe automáticamente las consultas para que se conserve la semántica. Cuando la aplicación cliente recupera los datos cifrados de la base de datos, el mismo controlador los descifra de forma transparente y devuelve el texto sin formato a la aplicación cliente. En consecuencia, *SQL Server* nunca ve la información confidencial en texto sin formato. El servidor tampoco tiene acceso a la clave. Las claves se gestionan completamente en el lado del cliente.

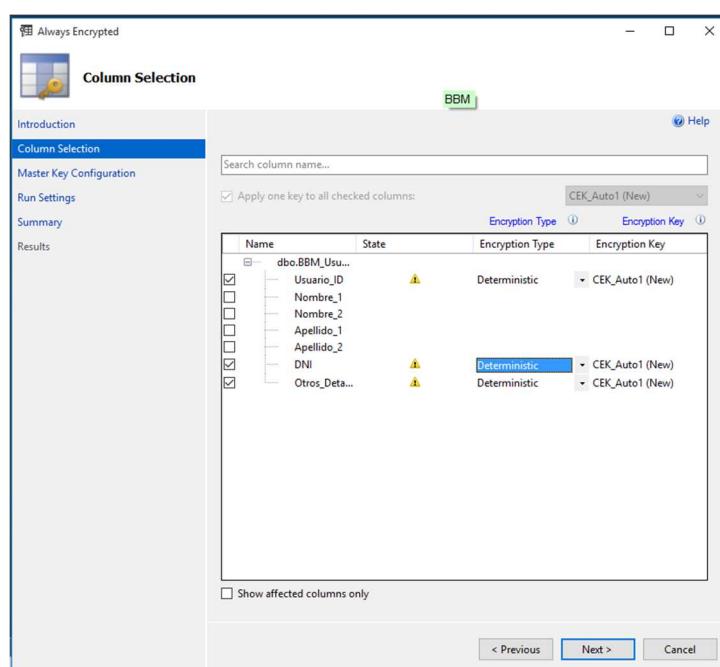


Estos son los pasos para encriptar datos en una tabla. Primero, nos conectamos a la base de datos usando SSMS. Hacemos clic con el botón derecho en la tabla de destino, que contiene datos confidenciales, y seleccionamos **cifrar columnas (Encrypt Columns)**. En este escenario, estamos encriptando columnas para la tabla **trial.BBM_Usuario**.

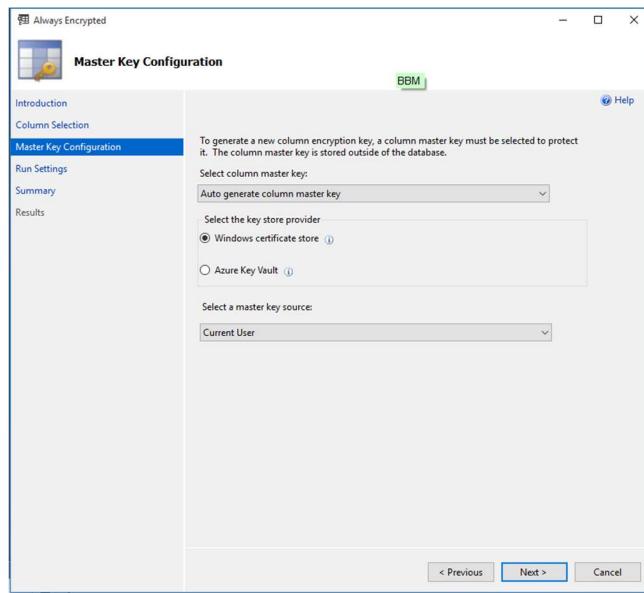
Esta acción inicia el asistente *Always Encrypted*. La primera ventana es informativa:



Una vez que se hace clic en el botón **Siguiente**, se procederá al paso de selección de columna. En este escenario, estamos seleccionando las columnas **Usuario_ID**, **DNI** y **Otros_Detalles** para cifrarlas.

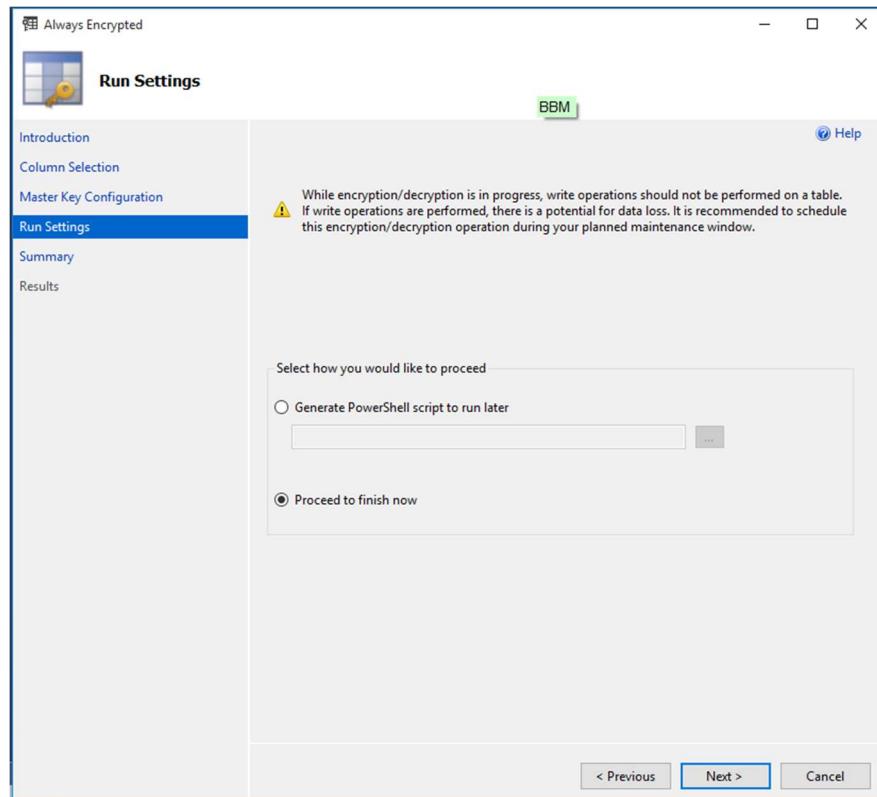


Una vez que hace clic en Siguiente, el asistente pasa al paso de configuración de la clave maestra

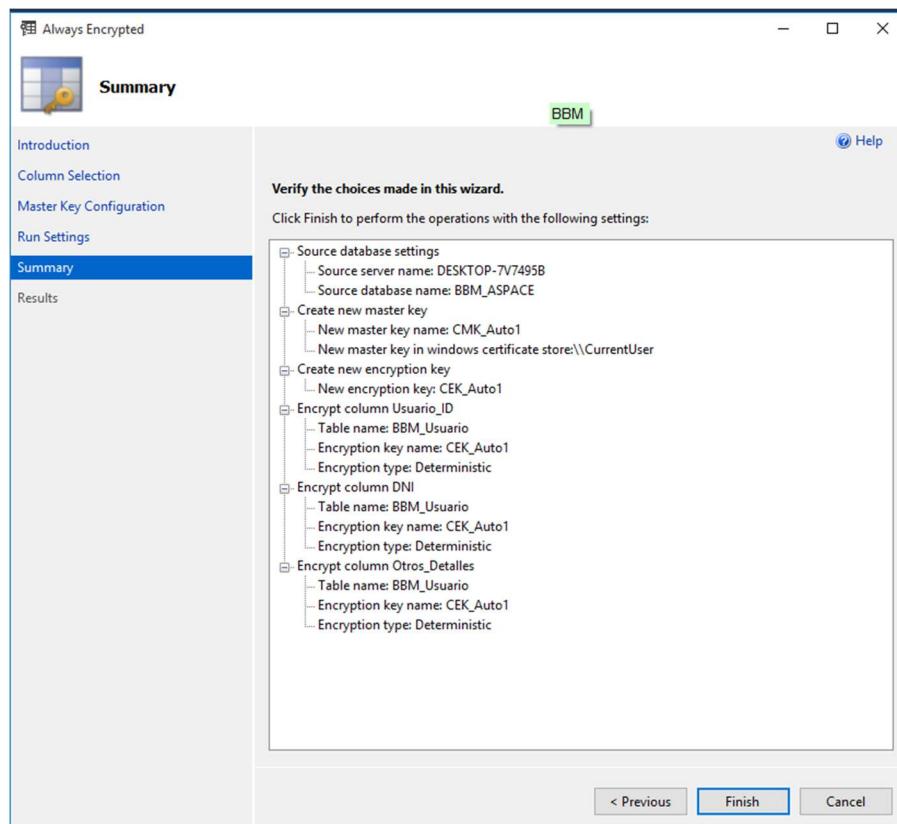


La opción aquí incluye una decisión sobre si se quiere generar automáticamente la CMK o usar una existente. También se decide dónde almacenar la CMK, ya sea localmente en la **Tienda de certificados de Windows** o en **Azure Key Vault**. Para esta demostración, estoy seleccionando el **almacén de certificados de Windows para la máquina local**.

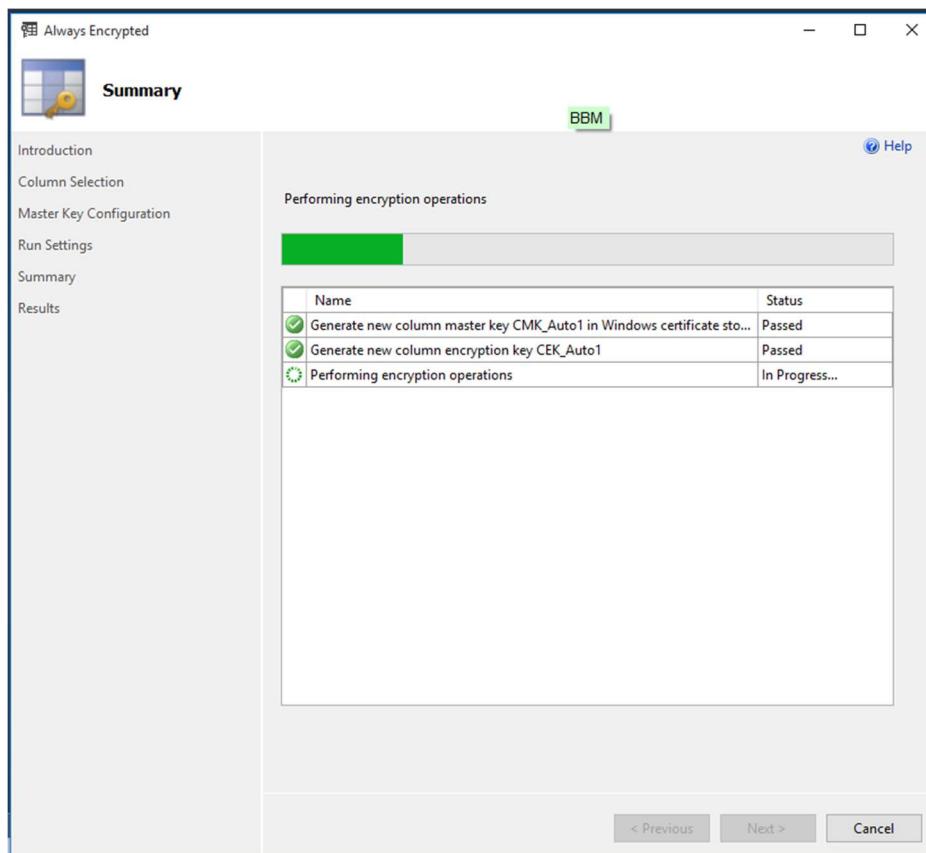
Una vez que hacemos clic en Siguiente, se nos presenta el cuadro de diálogo Ejecutar configuración. Podemos ejecutar nuestros pasos de configuración inmediatamente o generar un script de PowerShell para ejecutarlo más tarde.



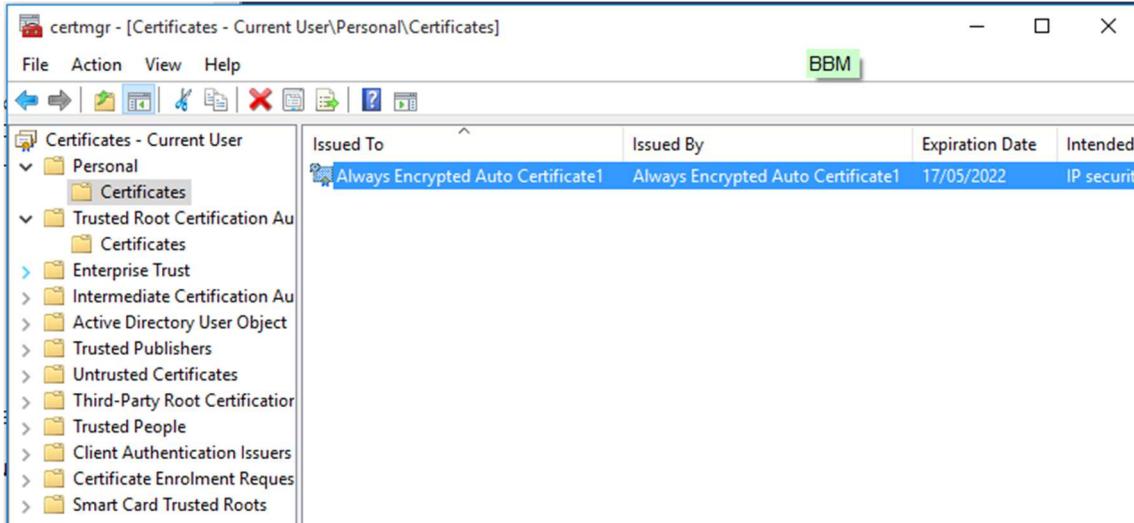
En la página Resumen, solo revisamos las selecciones. Haga clic en Finalizar



Empieza una pantalla de carga:



Cuando **finalizamos**, se selecciona el almacén de certificados de Windows para nuestro certificado. Para ello pulsamos en el teclado **Win + R** y escribimos **certmgr.msc**. En la ventana que se abre, buscamos el **certificado automático siempre cifrado en certificados personales**, como se muestra aquí:



También se pueden encontrar las **Claves de cifrado de columna** y **Claves maestras de columna** en **Base de datos** en la dirección **Base de datos>Tabla>Security>Always Encrypted**.



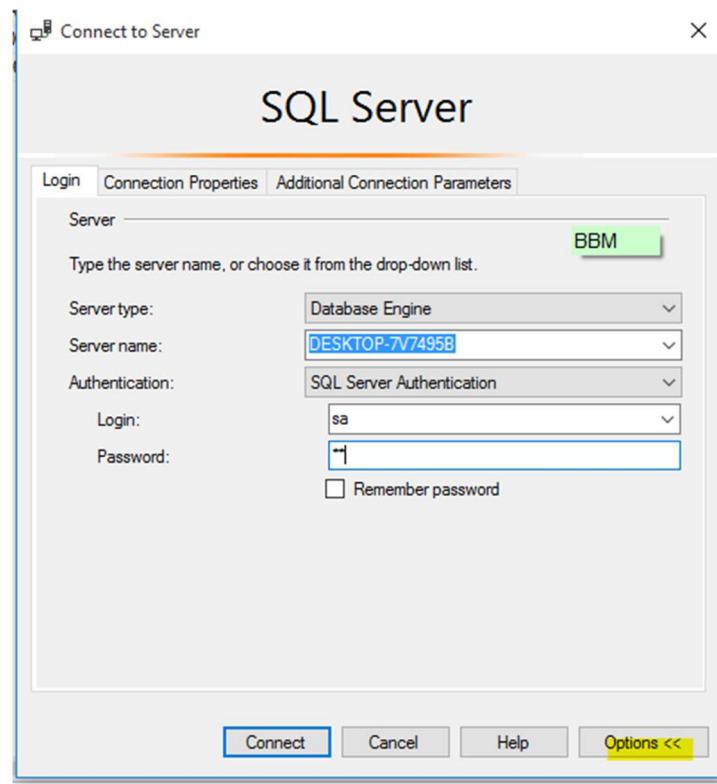
Ahora ejecutamos una instrucción **SELECT** para listar los datos de la tabla.

```
SQLQuery1.sql - D...MASPACE(BBM (56)) * 
SELECT * FROM trial.BBM_Usuario
```

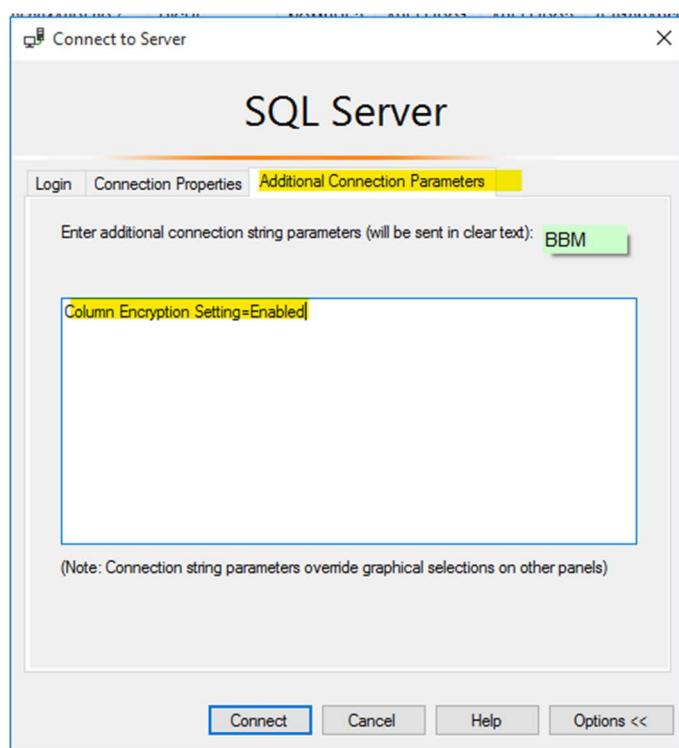
Usuario_ID	Nombre_1	Nombre_2	Apellido_1	Apellido_2	DNI	Otros_Detalles
1	PACO	GONZALEZ	PEREZ		0x015A7482620419E1211B4A2CB837C0323944D8D3AEE0A...	0x0163C2EB6CD482EA0BDD4E478E
2	RICHI	NOMBRE2	APPELLIDO1	APPELLIDO2	0x0198AB607BE29FC4CD9D9090F58983C84A87FC0FBA17A...	0x01D894C86CE94DF66AC6C04B8F
3	NOMBRETEST	NOMBRE2	APPELLIDO1	APPELLIDO2	0x01E7E1034BB0B0740D26B947C68C8C477CAA696E70AC54...	0x01D894C86CE94DF66AC6C04B8F
4	ANA	VAZQUEZ	NUNEZ		0x01DA05329F14F59CF375DE818DE285E6E52A26CB54CF6C6...	0x0163C2EB6CD482EA0BDD4E478E

Todos los valores están encriptados para las columnas que seleccionamos.

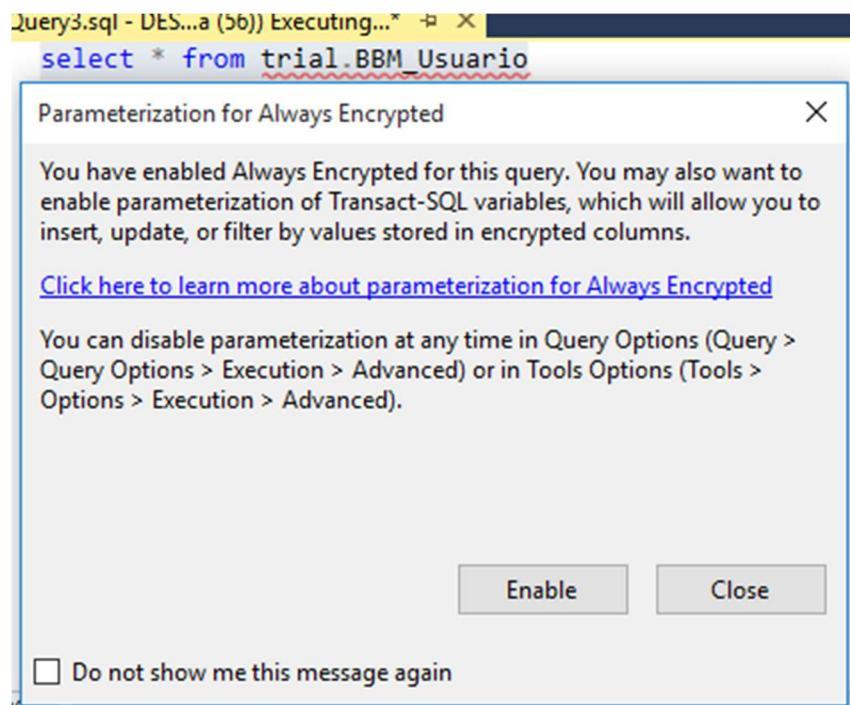
Tenemos todos los requisitos previos en su lugar y el certificado, por lo que se supone que se deben ver los datos sin problemas. Para hacer esto, se necesita especificar un parámetro de cadena de conexión especial. Para SSMS, solo es necesario abrir el cuadro de diálogo de conexión y hacer clic en el botón Opciones.



Elegimos la pestaña **Additional Connection Parameters**. Aquí ingresamos la cadena **Column Encryption Setting = Enabled**.



Después de escribir la configuración, hacemos clic en **Conectar** y volvemos a ejecutar la instrucción de selección. Para ejecutar la consulta sale este mensaje (le damos a **Enable**)



El resultado es este:

The screenshot shows the SQL Server Management Studio interface with the query "select * from trial.BBM_Usuario" executed. The results tab displays the following data:

	Usuario_ID	Nombre_1	Nombre_2	Apellido_1	Apellido_2	DNI	Otros_Detalles
1	USERGPP012	PACO		GONZALEZ	PEREZ	11111012S	
2	NO CUMPLE	RICHI	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
3	USERTEST	NOMBRETEST	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
4	USERVNA013	ANA	MARIA	VAZQUEZ	NUNEZ	11111013N	

Auditoría

Una auditoría es la combinación de varios elementos en un solo paquete para un grupo específico de acciones del servidor o acciones de la base de datos. Los componentes de la auditoría de SQL Server se combinan para producir un resultado que se denomina auditoría, al igual que una definición de informe combinada con gráficos y elementos de datos produce un informe. Utiliza eventos extendidos para ayudar a crear una auditoría y se realiza a nivel de instancia de SQL Server pudiendo tener varias en una misma.

Existen varios niveles de auditoría para SQL Server, según los requisitos gubernamentales o de estándares para su instalación. SQL Server Audit proporciona las herramientas y los procesos que debe tener para habilitar, almacenar y ver auditorías en varios objetos de servidor y base de datos.

Se pueden registrar los grupos de acciones de auditoría del servidor por instancia y los grupos de acciones de auditoría de la base de datos o las acciones de auditoría de la base de datos por base de datos. El evento de auditoría ocurrirá cada vez que se encuentre la acción auditable.

Cuando se define una auditoría, hay que especificar la ubicación para la salida de los resultados. Este es el destino de la auditoría. La auditoría se crea en un estado deshabilitado y no audita automáticamente ninguna acción. Una vez habilitada la auditoría, el destino de la auditoría recibe datos de la auditoría.

La especificación de auditoría del servidor recopila muchos grupos de acciones a nivel de servidor generados por la función Eventos extendidos. Puede incluir grupos de acciones de auditoría en una especificación de auditoría de servidor. Los grupos de acciones de auditoría son grupos de acciones predefinidos, que son eventos atómicos que ocurren en el motor de base de datos. Estas acciones se envían a la auditoría, que las registra en el destino.

Los eventos de auditoría son las acciones atómicas que puede auditar el motor de SQL Server. Los grupos de acciones de auditoría son grupos de acciones predefinidos. Ambos están en el ámbito de la base de datos de SQL Server. Estas acciones se envían a la auditoría, que las registra en el destino. No incluya objetos del ámbito del servidor, como las vistas del sistema, en una especificación de auditoría de la base de datos del usuario.

Los resultados de una auditoría se envían a un destino, que puede ser un archivo, el registro de eventos de seguridad de Windows o el registro de eventos de la aplicación de Windows. Los registros deben revisarse y archivarse periódicamente para asegurarse de que el destino tenga suficiente espacio para escribir registros adicionales.

Cuando se está guardando información de auditoría en un archivo, para ayudar a evitar la manipulación, puede restringir el acceso a la ubicación del archivo de las siguientes maneras:

- La cuenta de servicio de SQL Server debe tener permisos de lectura y escritura.
- Los administradores de auditoría suelen requerir permisos de lectura y escritura. Esto supone que los administradores de auditoría son cuentas de Windows para la administración de archivos de auditoría, tales como: copiarlos en diferentes recursos compartidos, hacer una copia de seguridad de ellos, etc.
- Los lectores de auditoría autorizados para leer archivos de auditoría deben tener permiso de lectura.
- Incluso cuando el motor de base de datos está escribiendo en un archivo, otros usuarios de Windows pueden leer el archivo de auditoría si tienen permiso. El motor de base de datos no tiene un bloqueo exclusivo que evite las operaciones de lectura.

Microsoft recomienda en sus *best practises*, generar informes de auditoría a partir de una instancia separada de SQL Server, a la que solo tienen acceso los administradores de auditoría o los lectores de auditoría. Al utilizar una instancia separada del Motor de base de datos para la generación de informes, se ayuda a evitar que usuarios no autorizados obtengan acceso al registro de auditoría.

Para auditar, se puede utilizar SQL Server Management Studio o Transact-SQL. Una vez creada y habilitada la auditoría, el objetivo recibirá entradas. Usando el Visor de archivos de registro en SQL Server Management Studio o la función [fn_get_audit_file](#) para leer el archivo de destino.

El proceso general para crear y utilizar una auditoría es el siguiente (uso de primer ejemplo el script de una auditoría sobre server):

Creación de application log

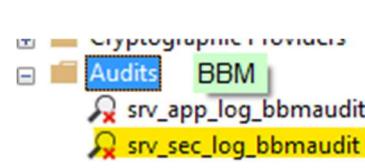
```
use master
go
create server audit [srv_app_log_bbmaudit]
    to application_log
with
    ( queue_delay = 1000,
        on_failure = fail_operation --> SI DA UN ERROR, INTERRUMPE LA
BBM_AUDIT
)
Go
```



NOTA: VER RESULTADO EN GUI SECURITY -> AUDIT

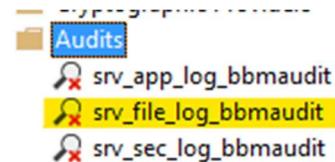
Creación de security log

```
use master
go
create server audit [srv_sec_log_bbmaudit]
    to security_log
with
    ( queue_delay = 1000,
        on_failure = continue --> SI FALLA, CONTINUA EJECUTANDO
)
Go
```



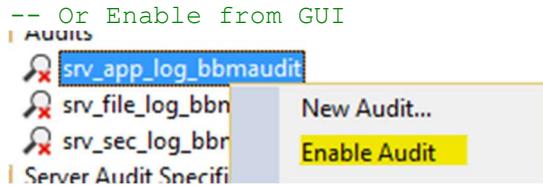
Creamos un **log en un archivo** y controlo lo que audito

```
use master
go
create server audit [srv_file_log_bbmaudit]
to file
(  filepath = 'c:\BBM_AUDIT\' 
   ,maxsize = 0 mb
   ,max_rollover_files = 2147483647
   ,reserve_disk_space = off
)
with
( queue_delay = 1000,
  on_failure = continue
)
go
```



Habilitamos en la opción **enable on**:

```
ALTER SERVER AUDIT srv_file_log_bbmaudit WITH (STATE = ON)
GO
```



Creación de especificación de auditoría de SERVIDOR para: Filelog_Audits

```
use master
go
create server audit specification [srv_instance_log_bbmaudit]
for server audit [srv_file_log_bbmaudit]
  add (server_state_change_group),
  add(backup_restore_group),
  add (dbcc_group)
with (state = on)
go
```

Provocamos uno de los eventos indicados en la especificación de auditoría del servidor.

```
use master
go
backup database BBM_ASPACE
  to disk = 'c:\BBM_AUDIT\BBM_ASPACE.bak'
  with init;
go
```

Miramos la base de datos actual (AUDIT ACTION TYPE)

```
DBCC CHECKDB ;
GO
```

```
DBCC results for 'BBM_ASPACE'.
Warning: NO_INDEX option of checkdb being used. Checks on non-system indexes will be skipped.
Service Broker Msg 9675, State 1: Message Types analyzed: 14.
Service Broker Msg 9676, State 1: Service Contracts analyzed: 6.
Service Broker Msg 9667, State 1: Services analyzed: 3.
Service Broker Msg 9668, State 1: Service Queues analyzed: 3.
Service Broker Msg 9669, State 1: Conversation Endpoints analyzed: 0.
Service Broker Msg 9674, State 1: Conversation Groups analyzed: 0.
Service Broker Msg 9670, State 1: Remote Service Bindings analyzed: 0.
Service Broker Msg 9605, State 1: Conversation Priorities analyzed: 0.
DBCC results for 'sys.sysrscols'.
There are 1485 rows in 17 pages for object "sys.sysrscols".
DBCC results for 'sys.sysrowsets'.
There are 219 rows in 3 pages for object "sys.sysrowsets".
DBCC results for 'sys.sysclones'.
There are 0 rows in 0 pages for object "sys.sysclones".
DBCC results for 'sys.sysallocunits'.
There are 251 rows in 3 pages for object "sys.sysallocunits".
DBCC results for 'sys.sysfiles1'.
There are 3 rows in 1 pages for object "sys.sysfiles1".
DBCC results for 'sys.sysseobjvalues'.
There are 0 rows in 0 pages for object "sys.sysseobjvalues".
DBCC results for 'sys.syspriorities'.
There are 0 rows in 0 pages for object "sys.syspriorities".
DBCC results for 'sys.sysdatabases'.
```

Con el comando que tenemos a continuación podríamos comprobar la base de datos sin nonclustered indexes (AUDIT ACTION TYPE)

```
DBCC CHECKDB (BBM_ASPACE, NOINDEX);
GO
```

Para ver los registros de una auditoría con salida a un archivo: DONDE MIRO LA BBM_AUDIT

```
SELECT *
FROM sys.fn_get_audit_file ('C:\BBM_AUDIT\*', ,default, default);
GO
```

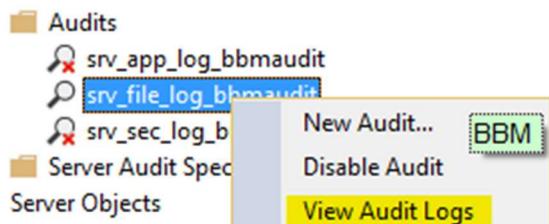
event_time	sequence_number	action_id	succeeded	permission_bitmask	is_column_permission	session_id	server_principal_id	database_principal_id	target_server_principal_id	target_database_principal_id
2021-05-24 14:00:49.7210712	1	AUSC	1	0x00	0	53	266	0	0	0
2021-05-24 14:21:28.6612870	1	BA	1	0x00	0	53	266	1	0	0
2021-05-24 14:21:40.5517375	1	DBCC	1	0x00	0	53	266	1	0	0
2021-05-24 14:21:40.5517375	1	DBCC	1	0x00	0	53	266	1	0	0
2021-05-24 14:21:53.6768725	1	DBCC	1	0x00	0	53	266	1	0	0
2021-05-24 14:21:53.6768725	1	DBCC	1	0x00	0	53	266	1	0	0
2021-05-24 14:23:04.0686487	1	BA	1	0x00	0	53	266	1	0	0

```
-- Por ejemplo el del Backup de mi base de datos: BBM_ASPACE
use master
go
backup database BBM_ASPACE
to disk = 'c:\BBM_AUDIT\BBM_ASPACE.bak'
with init;
go
```

Para ver los registros de una auditoría con salida a un archivo:

```
SELECT *
FROM sys.fn_get_audit_file ('C:\BBM_AUDIT\*', ,default, default);
GO
```

En caso de hacerlo en GUI, haríamos clic de botón derecho>View Audit Logs



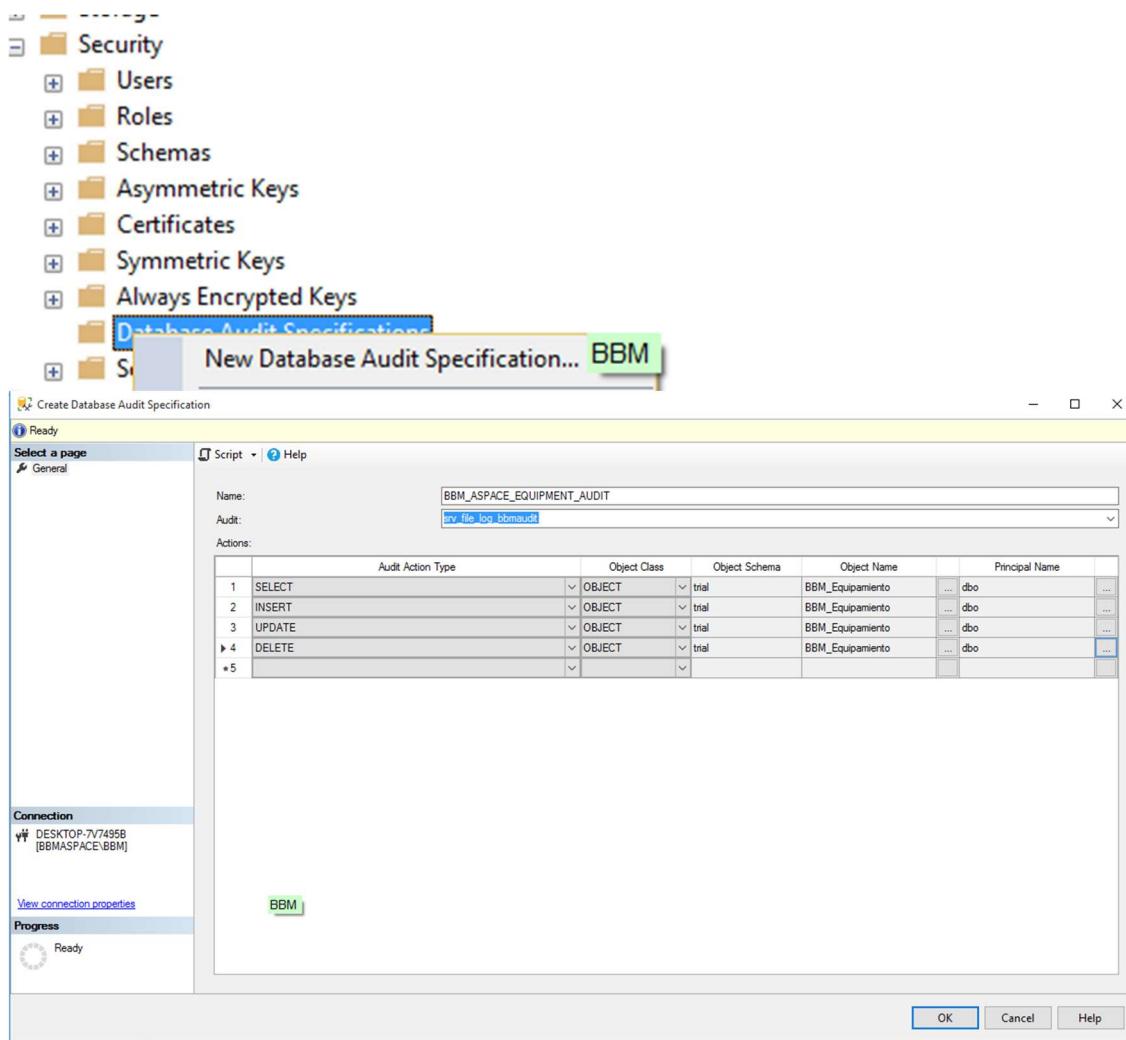
Para la creación de especificación a nivel de bases de datos

```
USE BBM_ASPACE
GO

SELECT * FROM trial.BBM_Equipamiento
GO
```

ID_Equipamiento	Nombre_Equipo	CreateDate	ModifiedDate
1	Electro	2021-03-12 22:11:21.113	2021-03-12 22:12:46.747

Hacemos clic de botón derecho en Security>Database Audit Specification...



En Script sería algo así

```
CREATE DATABASE AUDIT SPECIFICATION [BBM_ASPACE_EQUIPMENT_AUDIT]
FOR SERVER AUDIT [srv_file_log_bbmaudit]
```

```

ADD (SELECT ON OBJECT::[trial].[BBM_Equipamiento] BY [dbo]),
ADD (INSERT ON OBJECT::[trial].[BBM_Equipamiento] BY [dbo]),
ADD (UPDATE ON OBJECT::[trial].[BBM_Equipamiento] BY [dbo]),
ADD (DELETE ON OBJECT::[trial].[BBM_Equipamiento] BY [dbo])
GO

```

```

ALTER DATABASE AUDIT SPECIFICATION [BBM_ASPACE_EQUIPMENT_AUDIT]
WITH (STATE = ON)
GO

```

Probamos realizando una consulta

```

USE BBM_ASPACE
GO

```

```

SELECT * FROM trial.BBM_Equipamiento
GO

```

Realizamos una inserción

```

INSERT INTO [trial].[BBM_Equipamiento]
values ('TEST','2021-05-24 22:12:52.114','2021-05-25 22:12:52.114')
GO
-- (1 row affected)

```

Para ver los registros de una auditoría con salida a un archivo:

```

SELECT *
    FROM sys.fn_get_audit_file
    ('C:\BBM_AUDIT\*.sqlaudit', default, default);
GO

```

	event_time	sequence_number	action_id	succeeded	permission_bitmask	is_column_permission	session_id	server_principal_id	database_principal_id	target_server_principal_id	target_database_principal_id
1	2021-05-24 14:00:49.7210712	1	AUSC	1	0x00	0	53	266	0	0	0
2	2021-05-24 14:21:28.6612870	1	BA	1	0x00	0	53	266	1	0	0
3	2021-05-24 14:21:40.5517375	1	DBCC	1	0x00	0	53	266	1	0	0
4	2021-05-24 14:21:40.5517375	1	DBCC	1	0x00	0	53	266	1	0	0
5	2021-05-24 14:21:53.6768725	1	DBCC	1	0x00	0	53	266	1	0	0
6	2021-05-24 14:21:53.6768725	1	DBCC	1	0x00	0	53	266	1	0	0
7	2021-05-24 14:23:04.0686487	1	BA	1	0x00	0	53	266	1	0	0
8	2021-05-24 14:31:34.0434039	1	SL	1	0x0001	1	53	266	1	0	0
9	2021-05-24 14:34:51.1884795	1	SL	1	0x0001	1	57	266	1	0	0
10	2021-05-24 14:35:39.2577923	1	IN	1	0x0008	0	53	266	1	0	0

```

ALTER DATABASE AUDIT SPECIFICATION [BBM_ASPACE_EQUIPMENT_AUDIT]
WITH (STATE = OFF)
GO

```

```

DROP DATABASE AUDIT SPECIFICATION [BBM_ASPACE_EQUIPMENT_AUDIT]
GO

```

NOTA: En el caso de una falla durante el inicio de la auditoría, el servidor no se iniciará. En este caso, el servidor se puede iniciar usando la opción -f en la línea de comando.

LEGISLACIÓN

La Ley Orgánica de Protección de Datos y Garantía de Derechos Digitales (LOPDGDD) entró en vigor el 6 de diciembre de 2018, sustituyendo a la antigua *Ley Orgánica 15/1999 de Protección de Datos de Carácter Personal*. El objetivo de la LOPDGDD es adaptar la legislación española a la normativa europea, definida por el Reglamento General de Protección de Datos (RGPD), vigente desde el **25 de mayo de 2018**.

Por tanto, si hablamos de protección de datos en España, la norma de referencia es la LOPDGDD. Esta ley establece los requisitos y obligaciones en materia de protección de datos en empresas sobre cómo proceder con la información personal, así como los derechos que asisten a usuarios y consumidores.

La finalidad de la LOPDGDD es proteger la intimidad, privacidad e integridad del individuo, en cumplimiento con el artículo 18.4 de la Constitución Española. Del mismo modo, regula las obligaciones del individuo en todo proceso de transferencia de datos para garantizar la seguridad del intercambio.

Se consideran datos personales aquella información en texto, imagen o audio que permita la identificación de una persona. Existen datos que se consideran de poco riesgo, como el nombre o el correo electrónico, mientras que otros son considerados de riesgo más elevado, por ejemplo, datos sensibles relacionados con la religión o la salud personal.

No se tratan como datos personales aquellos que no permiten identificar a una persona. Por ejemplo, manuales de maquinaria, previsiones meteorológicas o datos que han pasado a ser anónimos, es decir, ya no se pueden relacionar con ningún individuo. En este caso, la normativa a cumplir es el Reglamento de libre circulación de datos no personales.

Asimismo, otra de sus principales finalidades es **establecer un marco legislativo para la protección de datos personales en Internet**. En este sentido, incorpora puntos muy a tener en cuenta, como el derecho al olvido o a la portabilidad, además de cambios en la obtención del consentimiento para recoger y usar la información personal.

En lo que nos atañe, **¿Qué pasos a seguir propone Microsoft para adaptarse a él?** Microsoft ha señalado su compromiso con el cumplimiento del RGPD y su apoyo a los clientes en el proceso de adaptación al mismo, que se extenderá por todo el entorno de las tecnologías y con ello por el entorno de Microsoft SQL Server. Por tanto, Microsoft recomienda a las empresas empezar su proceso de adaptación al Reglamento centrándose en cuatro aspectos principales:

- **Detectar.** Determinar qué información personal está siendo gestionada y dónde reside la misma, identificando qué servidores o bases de datos contienen información personal o qué filas o columnas pueden marcarse como contenedoras de la misma. SQL Server dispone de varias herramientas para descubrir los datos, como la tabla de sistema sys.columns, índices Full Text, Profiler o xevents.
- **Administrar.** Supervisar cómo se puede acceder a esa información personal y cómo esta es procesada y utilizada, asegurándose de que los permisos otorgados a las personas que acceden a los datos son los mínimos necesarios para la realización de su cometido. Este punto se puede acometer con SQL Server controlando permisos con **SQL Server Authentication**, enmascarando datos con **Dynamic Data Masking** o filtrando los datos que puede ver un usuario en una tabla con **Row-Level Security**.

- **Proteger.** Establecer controles de seguridad para prevenir, detectar y reaccionar a las debilidades e incumplimientos en la protección de datos. Esto requiere diferentes métodos para diferentes tipos de información y escenarios. Para proteger el dato, SQL Server dispone de varios mecanismos de encriptación a nivel físico y lógico, como encriptación de conexiones, **Transparent Data Encryption**, **Always Encrypted**. También podemos controlar quien y cuando está accediendo a los datos mediante SQL Server Audit.
- **Monitorizar e Informar.** Guardar auditorías de todas las operaciones relacionadas con el manejo de información personal, gestionar las solicitudes de información y notificar cuando se produzca un incumplimiento del reglamento. Así como realizar un seguimiento de estos procesos y procedimientos para garantizar que se mantienen actualizados.

Por último, en SQL Server podemos controlar el historial de cambios o accesos a una tabla con **System-Versioned temporal tables** y también con SQL Server Audit, y reportar esos fallos mediante **SQL Alerts** y **DB Mail** o paneles gráficos en tiempo real con **Power BI**.

Detectando y clasificando datos personales con SQL Server



El primer paso para cumplir con el GDPR es evaluar si este aplica a la organización y de ser así, qué datos son los que deben ajustarse para cumplirlo. Para realizar este análisis, es necesario conocer qué datos se tiene y donde están ubicados. Tener un esquema de clasificación de datos puede ayudar a responder a solicitudes de acceso a datos personales de una forma más rápida y controlada.

SQL Server cuenta con diversas herramientas que pueden ayudar a descubrir y clasificar estos datos personales:

- Se pueden ejecutar **consultas a tablas de sistema y metadatos** para identificar de una forma rápida si una columna tiene datos personales o no.
- Si se tienen campos de texto libre, como el típico “Observaciones”, se pueden utilizar **búsquedas de tipo Full-Text** para encontrar datos de carácter personal.
- Utilizar la característica **Propiedades Extendidas** para etiquetar y añadir una descripción a los campos o tablas que contengan datos personales. A partir de la versión 17.5 de SQL Server Management Studio se ha añadido una funcionalidad que nos ayudará a realizar esta clasificación de una forma muy fácil.

Detectar: Data Discovery and Classification & Vulnerability and Assesment

Lo ideal para poder detectar que columnas pueden contener información personal es tener una buena documentación de la base de datos para saber de una manera rápida las tablas que están implicadas, lamentablemente estamos en el mundo real y el 90% de las veces esta documentación no existe o está obsoleta, por lo que aquí viene SQL Server y su tabla **sys.columns** para ayudarnos.

La **tabla sys.columns** es una tabla de sistema que alberga información de todas las columnas de los objetos de una Base de datos, como pueden ser tablas o vistas.

Por ejemplo, para poder saber todas las columnas de una base de datos que contengan la palabra “nombre” se puede ejecutar la siguiente consulta:

```
SELECT object_name(object_id) AS TableName,
       Name AS ColumnName
  FROM sys.columns
 WHERE name LIKE '%nombre%'
```

TableName	ColumnName
1 BBM_Tutor	Nombre_1
2 BBM_Tutor	Nombre_2
3 BBM_Usuario	Nombre_1
4 BBM_Usuario	Nombre_2
5 BBM_Expediente	BBM_Enfermedad_Diagnosticada_Nombre_Enfermedad
6 BBM_Medicacion	Nombre_Medicamento
7 BBM_Enfermedad_Diagnosticada	Nombre_Comun_Enfermedad
8 BBM_Enfermedad_Diagnosticada	Nombre_Enfermedad
9 BBM_Usuario	Nombre_1
10 BBM_Usuario	Nombre_2
11 BBM_Emppleado	Nombre_1
12 BBM_Emppleado	Nombre_2
13 BBM_Enfermedad_Diagnosticada	Nombre_Comun_Enfermedad
14 BBM_Enfermedad_Diagnosticada	Nombre_Enfermedad
15 bbmlookexpediente	BBM_Enfermedad_Diagnosticada_Nombre_Enfermedad
16 BBM_Expediente	BBM_Enfermedad_Diagnosticada_Nombre_Enfermedad
17 BBM_Equipamiento	Nombre_Equipo
18 BBM_Medicacion	Nombre_Medicamento

STORED PROCEDURE + TEMPORAL TABLES RECUPERACIÓN

Partiendo de esa base, y COMPLETANDO LA TABLA TEMPORAL RECUPERACIÓN DE EXÁMEN Y EL PROCEDIMIENTO ALMACENADO, hem creado un pequeño script que empieza con una tabla temporal a modo de diccionario en la que se pondrán todas las palabras que se quieren buscar (en este caso palabras candidatas a albergar información personal) y luego mediante cursosres buscar esas palabras en todas las bases de datos y las guardarlas en otra tabla temporal que se mostrará al final.

```
USE BBM_ASPACE /*O EN BASE DE DATOS A USAR*/
```

```
GO
```

```
DROP PROCEDURE IF EXISTS sp_BBM_SEARCHENCRYPT
```

GO

```

CREATE      OR      ALTER      PROCEDURE
sp_BBM_SEARCHENCRYPT
AS
BEGIN
DECLARE          @DatabaseName
nvarchar(100)
, @Word nvarchar(50)
, @SQL nvarchar(max)

-- Borra la tabla #Words si existe
IF OBJECT_ID('tempdb.dbo.#Words', 'U') IS NOT NULL
DROP TABLE #Words;

-- Borra la tabla #DiscoverGDPR si existe
IF OBJECT_ID('tempdb.dbo.#DiscoverGDPR', 'U') IS NOT NULL
DROP TABLE #DiscoverGDPR;

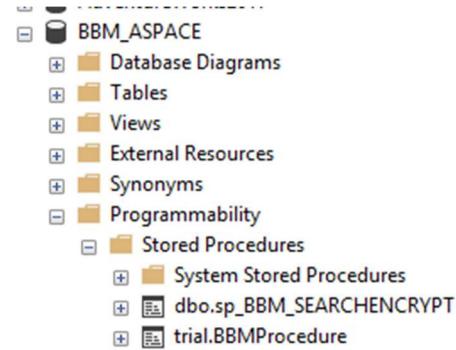
-- Creación de tabla #Words
CREATE TABLE #Words (word nvarchar(50))

-- Creación de tabla #DiscoverGDPR
CREATE TABLE #DiscoverGDPR (DatabaseName nvarchar(100), SchemaName
nvarchar(100), TableName nvarchar(100), ColumnName nvarchar(100))

-- Insertamos palabras a buscar en la tabla #Words
INSERT INTO #Words VALUES
-- Spanish
('Nombre')
,('Apellido')
,('Tel') -- Aquí cogería valores como Telefono y abreviaturas
,('Tfno')
,('Direccion')
,('Poblacion')
,('Ciudad')
,('Pais')
,('Postal') -- Aquí cogería valores como CódigoPostal, DireccionPostal,
DestinoPostal
,('CP')
,('Nac') -- Aquí cogería valores como Nacionalidad, FechaNacimiento,
LugarNacimiento
,('DNI')
,('CIF')
,('NIE')
,('Pasaporte')
,('Identifi')
,('Mail') -- Aquí cogería valores como Mail, Email, Correo Mail
,('Correo') -- Aquí cogería valores como Correo, CorreoElectronico
,('Foto') -- Aquí cogería valores como Foto, Fotografia
,('Banco')
,('Tarjeta')
,('Cuenta')
,('Numero') -- Aquí cogería valores como NumeroCuenta, NumeroTelefono
,('IP')

-- English (Algunos términos de Spanish también son válidos, como Postal
o Identifi)
,('Name')
,('Surname')
,('Phone') -- Aquí cogería valores como Phone, PhoneNumber, Cellphone

```



```
,('Mobile')
,('Cell')
,('Celular')
,('Address')
,('City')
,('Country')
,('ZIP')
,('Code')
,('Birthday')
,('Passport')
,('Photo')
,('Bank')
,('Card')
,('Account')
,('Number')
,('IP')

-- Creamos un cursor con las Bases de Datos en las que queremos buscar
la información
DECLARE db_cursor CURSOR
FOR

    SELECT name
    FROM master.sys.databases
    WHERE name NOT IN ('master', 'model', 'msdb', 'tempdb',
'distribution');

-- Iniciamos cursor db_cursor
OPEN db_cursor

-- Avanzamos cursor db_cursor
FETCH NEXT FROM db_cursor INTO @DatabaseName;

-- Loop db_cursor
WHILE @@FETCH_STATUS = 0
BEGIN

-- Creamos un cursor que recorra la tabla #Words
DECLARE Word_Cursor CURSOR FOR
SELECT * FROM #Words

-- Iniciamos cursor Word_Cursor
OPEN Word_Cursor

-- Avanzamos cursor Word_Cursor
FETCH NEXT FROM Word_Cursor INTO @Word

-- Loop Word_Cursor
WHILE @@FETCH_STATUS = 0
BEGIN

-- Creamos la sentencia
SET @SQL = 'USE ' + @DatabaseName + ';' +
'INSERT INTO #DiscoverGDPR ' +
'SELECT ''' + @DatabaseName + ''' AS [database], ' +
'        SCHEMA_NAME(schema_id) AS [schema], ' +
'        t.name AS table_name, c.name AS column_name ' +
'FROM sys.tables AS t ' +
'INNER JOIN sys.columns c ON t.OBJECT_ID = c.OBJECT_ID
' +
```

```

        'WHERE      c.name    LIKE    ''%'+ @Word + '%'    COLLATE
SQL_Latin1_General_CI_AI'

-- Ejecutamos sentencia
EXEC sp_executesql @SQL

-- Avanzamos cursor Word_Cursor
FETCH NEXT FROM Word_Cursor INTO @Word

END

-- Cerramos y borramos cursor Word_Cursor
CLOSE Word_Cursor
DEALLOCATE Word_Cursor

-- Avanzamos cursor db_cursor
FETCH NEXT FROM db_cursor INTO @DatabaseName;

END

-- Cerramos y borramos cursor db_cursor
CLOSE db_cursor;
DEALLOCATE db_cursor;

-- Mostramos los datos
SELECT *
FROM #DiscoverGDPR
ORDER BY DatabaseName, SchemaName, TableName, ColumnName
END
GO

```

Aquí podemos ver parte de los resultados que devolvería la base de datos BBM_ASPACE:

`EXEC sp_BBM_SEARCHENCRYPT`

	DatabaseName	SchemaName	TableName	ColumnName
139	BBM_ASPACE	dbo	BBM_Asistente_Social	Descripcion_funci...
140	BBM_ASPACE	dbo	BBM_Asistente_Social	Descripcion_funci...
141	BBM_ASPACE	dbo	BBM_Cocina	Descripcion_funci...
142	BBM_ASPACE	dbo	BBM_Cocina	Descripcion_funci...
143	BBM_ASPACE	dbo	BBM_Contacto_Pac...	Email
144	BBM_ASPACE	dbo	BBM_Direccion	Descripcion_funci...
145	BBM_ASPACE	dbo	BBM_Direccion	Descripcion_funci...
146	BBM_ASPACE	dbo	BBM_Epleado	DNI
147	BBM_ASPACE	dbo	BBM_Epleado	Email
148	BBM_ASPACE	dbo	BBM_Epleado	Nombre_1
149	BBM_ASPACE	dbo	BBM_Epleado	Nombre_2
150	BBM_ASPACE	dbo	BBM_Epleado	Telefono
151	BBM_ASPACE	dbo	BBM_Enfermedad_...	Descripcion
152	BBM_ASPACE	dbo	BBM_Enfermedad_...	Descripcion
153	BBM_ASPACE	dbo	BBM_Enfermedad_...	Nombre_Comun_...
154	BBM_ASPACE	dbo	BBM_Enfermedad_...	Nombre_Enferme...
155	BBM_ASPACE	dbo	BBM_Enfermera	Descripcion_funci...

Otra manera de detectar es con la evaluación de vulnerabilidades de SQL (Vulnerability and Assesment) es una herramienta fácil de usar que puede ayudarle a

detectar posibles vulnerabilidades de la base de datos, realizar su seguimiento y corregirlas. Úsela para mejorar la seguridad de la base de datos de manera proactiva. Está disponible en **SQL Server Management Studio (SSMS)** para SQL Server 2012 o versiones posteriores.

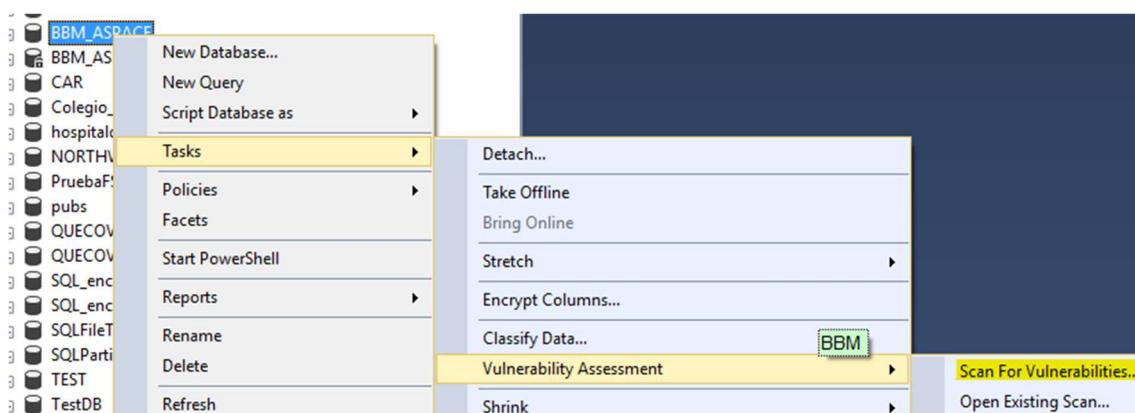
Proporciona visibilidad sobre el estado de seguridad e incluye acciones recomendadas para resolver problemas de seguridad y mejorar la seguridad de la base de datos. En este sentido, puede ayudar a:

- Satisfacer los requisitos de cumplimiento que requieren los informes de examen de base de datos
- Cumplir los estándares de privacidad de los datos
- Supervisar un entorno de base de datos dinámico donde resulta difícil realizar un seguimiento de los cambios

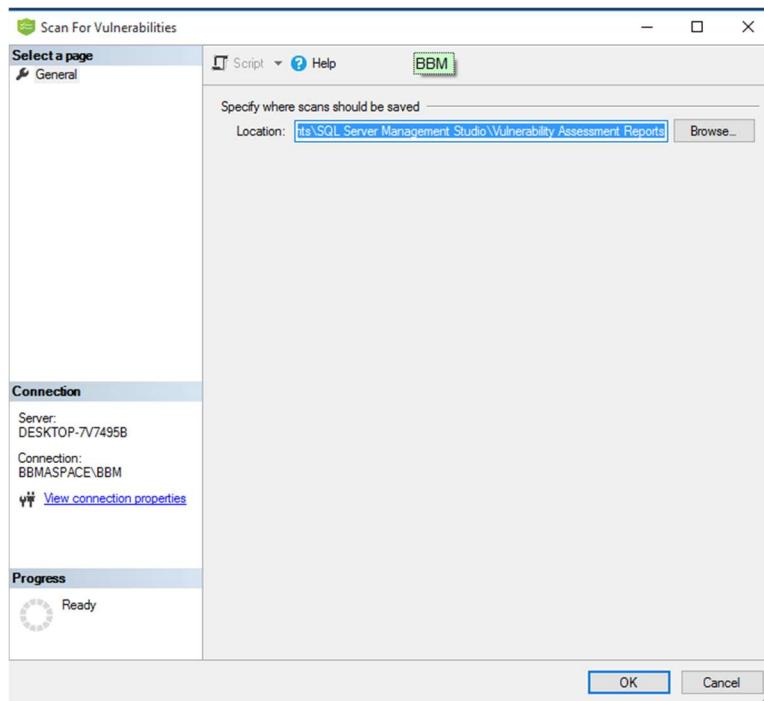
El servicio ejecuta un examen directamente en la base de datos y emplea una base de reglas de conocimientos que marcan las vulnerabilidades de seguridad y resaltan las desviaciones con respecto a los procedimientos recomendados, como errores de configuración, permisos excesivos y datos confidenciales sin protección. Las reglas se basan en procedimientos recomendados de Microsoft y se centran en los problemas de seguridad que presentan los riesgos más grandes para la base de datos y sus valiosos datos. Estas reglas también representan muchos de los requisitos que deben cumplir diversos organismos reguladores para satisfacer los estándares de cumplimiento.

Los resultados del examen incluyen pasos que requieren acción para corregir cada uno de los problemas y proporcionan scripts de solución personalizados donde sea aplicable. Un informe de evaluación se puede personalizar para el entorno y establecer una línea de base aceptable para las configuraciones de permisos, configuraciones de características y configuraciones de bases de datos.

Para ejecutar un examen de vulnerabilidades en la base de datos, hacemos **clic de botón derecho sobre la base de datos>Task>Vulnerability Assesment>Scan For Vulnerabilities**.



A partir de ahí, se abre un **Wizard**. El primer cuadro de diálogo permite especificar la ubicación donde se guardarán los exámenes. La dejo por defecto y le doy a **Aceptar** para examinar la base de datos en busca de vulnerabilidades.



Una vez finalizado el examen, se muestra automáticamente el informe de esta operación en el panel principal de SSMS. El informe presenta información general sobre el estado de seguridad, cuántos problemas se encontraron y sus niveles de gravedad respectivos. Los resultados incluyen advertencias sobre las desviaciones con respecto a los procedimientos recomendados, así como una instantánea de la configuración relacionada con la seguridad, como las entidades de seguridad y los roles de base de datos y sus permisos asociados. El informe de examen también proporciona un mapa de datos confidenciales detectados en la base de datos e incluye recomendaciones de los métodos integrados disponibles para protegerlo.

Vulnerability Assessment Results

DESKTOP-7V7495B: BBM_ASPACE
at 5/23/2021 9:51:20 AM

Total security checks	Total failing checks	Risk Distribution	Learn more
54 ✓	9 ✗	High Risk: 1 (Red bar) Medium Risk: 4 (Orange bar) Low Risk: 4 (Blue bar)	SQL Security Center Best Practices for SQL Security

[Failed \(9\)](#) [Passed \(45\)](#)

ID	Security Check	Category	Risk	Additional Information
VA1245	The dbo information should be consistent between the target DB and master	Surface Area Reduction	High	
VA1281	All memberships for user-defined roles should be intended	Auditing and Logging	Medium	No baseline set
VA1287	Sensitive data columns should be classified	Data Protection	Medium	No baseline set
VA1095	Excessive permissions should not be granted to PUBLIC role	Authentication and Authorization	Medium	
VA1219	Transparent data encryption should be enabled	Data Protection	Medium	
VA1069	Permissions to select from system tables and views should be revoked from non-sysadmins	Authentication and Authorization	Low	No baseline set
VA2031	Minimal set of principals should be granted database-scoped SELECT permission on objects or columns	Authentication and Authorization	Low	No baseline set
VA2032	Minimal set of principals should be granted database-scoped SELECT or EXECUTE permissions on schema	Authentication and Authorization	Low	No baseline set
VA1054	Excessive permissions should not be granted to PUBLIC role on objects or columns	Authentication and Authorization	Low	

Tras la revisión de los resultados, se determinan cuáles de los hallazgos del informe son verdaderos problemas de seguridad en el entorno. Si vamos al final del informe, veremos que el propio SSMS nos da una posible *Query* para solucionar este problema

Vulnerability Assessment Results

DESKTOP-7V7495B: BBM_ASPACE
at 5/23/2021 9:51:20 AM

Total security checks	Total failing checks	High Risk	Medium Risk	Low Risk
54	9	1	4	4

[Learn more](#)
[SQL Security Center](#)
[Best Practices for SQL Security](#)

Failed (9) Passed (45)

ID	Security Check	Category	Risk	Additional Information
VA2032	Minimal set of principals should be granted database-scoped SELECT or EXECUTE permissions on schema	Authentication and Authorization	Low	No baseline set

Approve as Baseline Clear Baseline

Status: Fail (no baseline set) [BBM](#)

Description: Every SQL Server securable has permissions associated with it that can be granted to principals. Permissions can be scoped at the server level (assigned to logins and server roles) or at the database level (assigned to database users and database roles). These rules check that only a minimal set of principals are granted database-scoped SELECT or EXECUTE permissions on schema.

Impact: Developing an application using a least-privileged user account (LUA) approach is an important part of a defensive, in-depth strategy for countering security threats. The LUA approach ensures that users follow the principle of least privilege and always log on with limited user accounts. Administrative tasks are broken out using fixed server roles, and the use of the sysadmin fixed server role is severely restricted. Always follow the principle of least privilege when granting permissions to database users. Grant the minimum permissions necessary to a user or role to accomplish a given task. See [https://msdn.microsoft.com/en-us/library/bb669084\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/bb669084(v=vs.110).aspx).

Rule Query:

```
AND grantee_principal_id NOT IN (DATABASE_PRINCIPAL_ID('guest'), DATABASE_PRINCIPAL_ID('public'))
AND permission_name IN ('SELECT', 'EXECUTE')
AND [state] IN ('G', 'W')
```

[Open in Query Editor Window](#)

Actual Result:

In Baseline	Permission Class	Object	Permission	Principal Type	Principal
<input checked="" type="checkbox"/>	SCHEMA	trial	EXECUTE	DATABASE_ROLE	BBMFisioGestion

Remediation: Revoke permissions from principals where not needed. It is recommended to have at most 1 principal granted a specific permission.

Remediation Script:

```
REVOKE EXECUTE ON SCHEMA::[trial] FROM [BBMFisioGestion]
```

[Open in Query Editor Window](#)

También es posible validar manualmente los errores haciendo clic en el botón **✓ Approve as Baseline**.

Vulnerability Assessment Results

sql2016: WideWorldImporters
at 11/22/2017 1:39:00 PM

Total security checks	Total failing checks	High Risk	Medium Risk	Low Risk	Learn more
54	6	2	3	1	SQL Security Center Best Practices for SQL Security

[Failed \(6\)](#) [Passed \(48\)](#)

ID	Security Check	Category	Risk	Additional Information
VA1281	All memberships for user-defined roles should be intended	Auditing and Logging	Medium	No baseline set
VA1285	Sensitive data columns should be identified	Data Protection	Medium	No baseline set

Approve as Baseline Clear Baseline

Name: VA1281 - All memberships for user-defined roles should be intended

Risk: Medium

Status: ✖ Fail (no baseline set)

Description: User-defined roles are security principals defined by the user to group principals to easily manage permissions. Monitoring these roles is important to avoid having excessive permissions. Create a baseline which defines expected membership for each user-defined role. This rule checks whether all memberships for user-defined roles are as defined in the baseline.

Impact: Keeping track of role memberships is important to avoid granting excessive permissions

Rule Query:

```
SELECT user_name(role_principal_id) as role_name, user_name(member_principal_id) as member_name
FROM sys.database_role_members
WHERE role_principal_id NOT IN (16384,16385,16386,16387,16389,16390,16391,16392,16393)
```

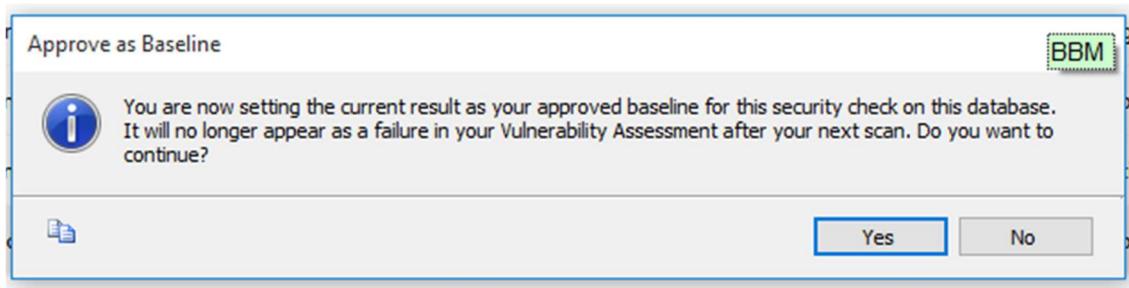
[Open in Query Editor Window](#)

Actual Result:

In Baseline	Role	Member
<input checked="" type="checkbox"/>	app_role	anna
<input checked="" type="checkbox"/>	app_role	app_dev

Remediation: Keep track of role membership and remove unnecessary members from roles to avoid granting excessive permissions or update baseline to comply with new changes

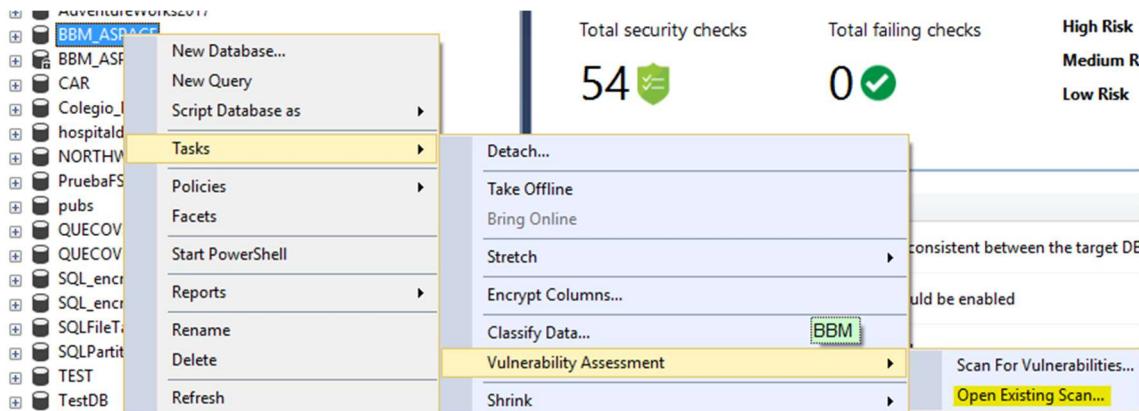
Aparecerá un cuadro de texto al que habrá que darle a **Yes** para aceptar los riesgo que tiene dar como válido un error.



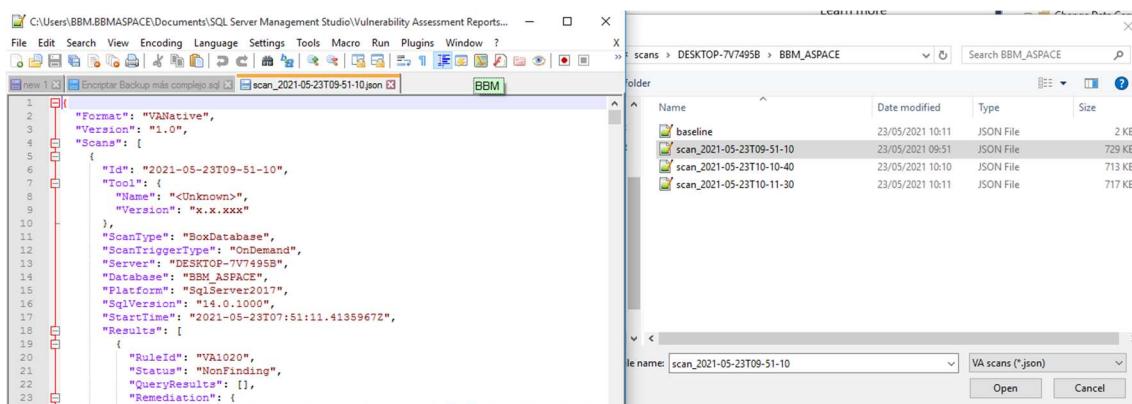
Después de completar la configuración de las **líneas de base de la regla**, se ejecuta un nuevo examen.

Vulnerability Assessment Results				
DESKTOP-7V7495B: BBM_ASPACE				
at 5/23/2021 10:11:32 AM				
Total security checks	Total failing checks			
54	0			
High Risk 0	Learn more			
Medium Risk 0	SQL Security Center			
Low Risk 0	Best Practices for SQL Security			
	BBM			
 Failed (0) Passed (54)				
ID	Security Check	Category	Status	Additional Information
VA1245	The dbo information should be consistent between the target DB and master	Surface Area Reduction	Pass	Per custom baseline
VA1219	Transparent data encryption should be enabled	Data Protection	Pass	Per custom baseline
VA1282	Orphan roles should be removed	Authentication and Authorization	Pass	Per custom baseline
VA1287	Sensitive data columns should be classified	Data Protection	Pass	Per custom baseline
VA2031	Minimal set of principals should be granted database-scoped SELECT permission on objects or columns	Authentication and Authorization	Pass	Per custom baseline
VA2032	Minimal set of principals should be granted database-scoped SELECT or EXECUTE permissions on schema	Authentication and Authorization	Pass	Per custom baseline

Si se quieren ver los escáneres hechos con anterioridad, seguimos la misma ruta que hicimos para el escáner, pero esta vez le damos a **Open Existing Scan...**



Los archivos se guardan en .json en esta ruta C:\Users\BBM.BBMASPACE\Documents\SQL Server Management Studio\Vulnerability Assessment Reports\scans\DESKTOP-7V7495B\BBM_ASPACE. Si los analizamos con el Notepad++ veremos algo así:

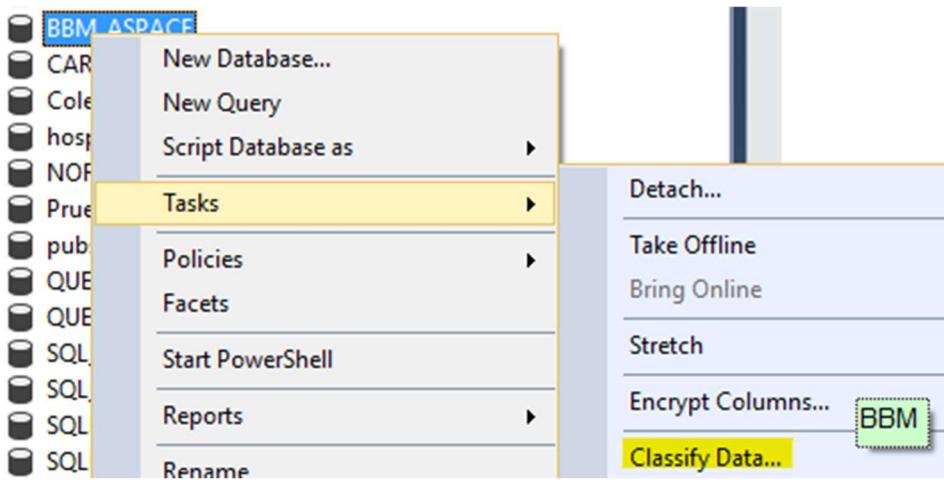


Clasificar

Una vez que tenemos detectadas las columnas que contienen datos de carácter personal, el siguiente paso que debemos tomar es clasificarlas según el tipo de datos que almacenen, para esto contamos con las **Propiedades Extendidas** de las columnas.

Las Propiedades Extendidas están disponibles desde **SQL Server 2008** y es una forma de añadir una descripción o una clasificación a cada columna, para acceder a esas características solo tendremos que pulsar botón derecho sobre una columna y pinchar en propiedades.

Podemos acceder a esta opción dando **click derecho encima de una base de datos->Tasks->Classify Data...**



Nada más abrirlo, y de forma automática ya nos hará una recomendación de clasificación de determinadas columnas.

Schema	Table	Column	Information Type	Sensitivity Label
0 classified columns				

Al pinchar, nos aparecerá la lista de columnas.

The screenshot shows a classification interface with a header bar indicating "7 columns with classification recommendations (click to minimize)". Below this is a table titled "0 classified columns" with columns: Schema, Table, Column, Information Type, and Sensitivity Label. A green box highlights the "Schema" column. At the bottom, there is a button labeled "Accept selected recommendations".

Schema	Table	Column	Information Type	Sensitivity Label
dbo	BBM_Contacto_Paciente	Email	Contact Info	Confidential - GDPR
dbo	BBM_Emppleado	Email	Contact Info	Confidential - GDPR
dbo	BBM_Equipamiento	Codigo_EAN	Credit Card	Confidential
dbo	BBM_Medicacion	Codigo_EAN	Credit Card	Confidential
dbo	BBM_Medicacion	Posologia	Credit Card	Confidential
trial	BBM_Medicacion	Codigo_EAN	Credit Card	Confidential
trial	BBM_Medicacion	Posologia	Credit Card	Confidential

Podemos seleccionar las columnas que queramos y pinchar en “Accept selected recommendations”. Esta acción lo que hará es añadir el Information Type y el Sensibility Label a las **Propiedades Extendidas** de la columna.

The screenshot shows the same classification interface as before, but with several rows selected. A green box highlights the "Schema" column for the first row. The selected rows have a blue background. The "Accept selected recommendations" button is visible at the bottom.

Schema	Table	Column	Information Type	Sensitivity Label
<input checked="" type="checkbox"/>	dbo	BBM_Contacto_Paciente	Contact Info	Confidential - GDPR
<input checked="" type="checkbox"/>	dbo	BBM_Emppleado	Contact Info	Confidential - GDPR
<input checked="" type="checkbox"/>	dbo	BBM_Equipamiento	Credit Card	Confidential
<input checked="" type="checkbox"/>	dbo	BBM_Medicacion	Credit Card	Confidential
<input checked="" type="checkbox"/>	dbo	BBM_Medicacion	Credit Card	Confidential
<input checked="" type="checkbox"/>	trial	BBM_Medicacion	Credit Card	Confidential
<input checked="" type="checkbox"/>	trial	BBM_Medicacion	Credit Card	Confidential

Una vez que acabemos con las columnas sugeridas, nos apoyaremos en la lista que habremos sacado con nuestro script en la fase de detección para clasificar otras columnas, esto se hace dándole al botón **Add Classification**. Nos aparece una columna a la derecha para elegir

The screenshot shows the classification interface with the "Add Classification" dialog box open on the right. The dialog contains fields for Schema (set to "trial"), Table ("BBM_Usuario"), Column ("DNI"), Information Type ("Other"), and Sensitivity Label ("Highly Confidential - GDPR").

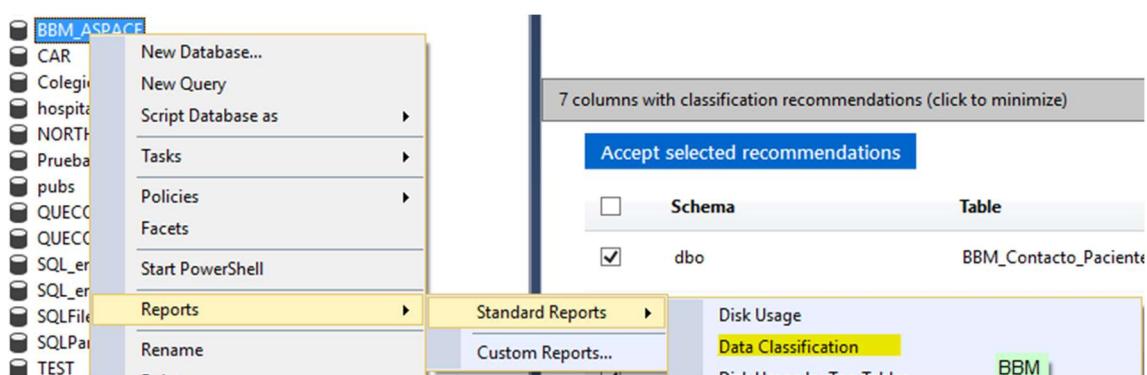
Al acabar, bajamos y le damos al botón Add.

La clasificación se hará eligiendo un Information type y un Sensitivity Label, dentro de cada uno encontraremos los siguientes tipos:

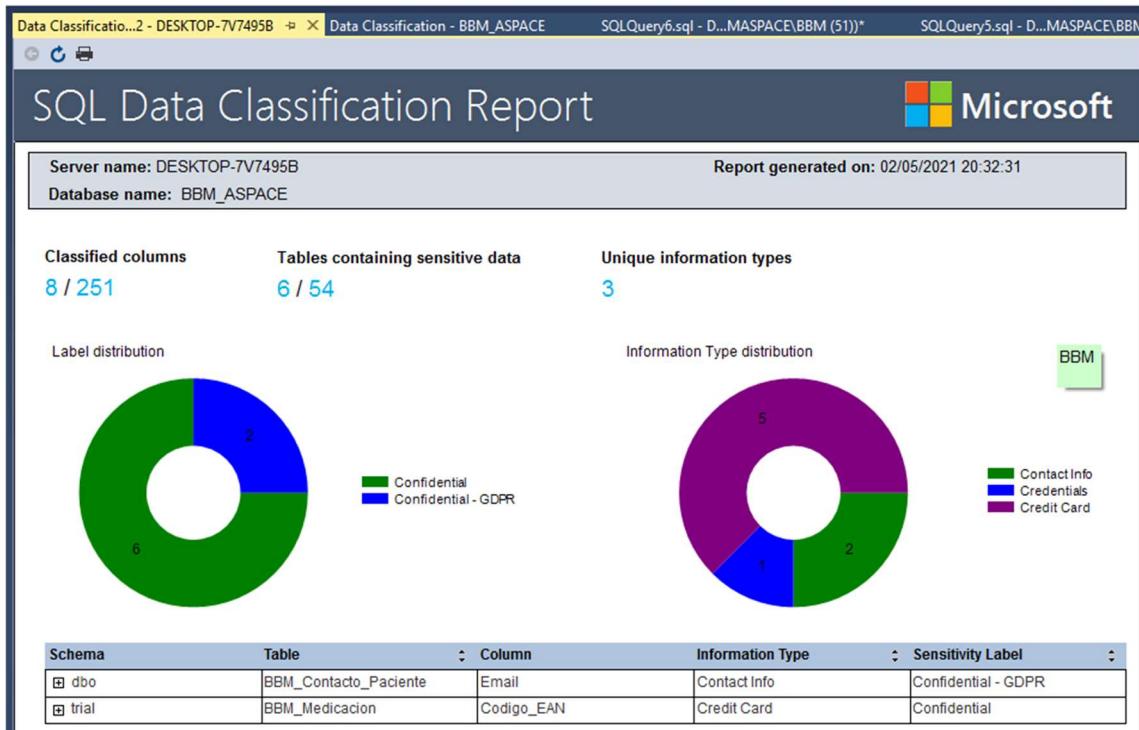
Information Type	Sensitivity Label
Banking	Public
Contact Info	General
Credentials	Confidential
Credit Card	Confidential - GDPR
Date Of Birth	Highly Confidential
Financial	Highly Confidential - GDPR
Health	[n/a]
Name	Sensitivity Label:
National ID	Public
Networking	General
SSN	
Other	
[n/a]	

Ahora que ya están detectados y clasificados los datos personales, se puede lanzar un **reporte** que nos enseñe datos estadísticos de esta clasificación de una forma rápida.

Accedemos a este reporte dando click derecho encima de una base de datos->Reports->Standard Reports->Data Classification



En nuestro caso únicamente veremos en el reporte 8 columnas, que son las que habíamos marcado previamente con la herramienta **Classify Data**.



Ahora que ya tenemos todas nuestras columnas perfectamente clasificadas, es hora de ir al **siguiente punto del GDPR** y ver cómo **Administrar** estos datos.

Administrando los accesos y el uso de los datos con SQL Server



Una vez que hemos detectado y clasificado nuestros datos, entramos en el segundo paso en nuestro camino para cumplir con el GDPR, ahora es el turno de **administrar los accesos a los datos y controlar el uso** que se hace de ellos.

¿Por qué esto es importante para el GDPR?

The GDPR specifically addresses the need for mechanisms which limit access to data, requiring “measures [that] shall ensure that by default personal data are not made accessible without the individual’s intervention to an indefinite number of natural persons.”

GDPR Article 25(2)–“Data protection by design and by default.”

SQL Server dispone de varias herramientas para el control de acceso, pero es el administrador SQL Server el que hace uso de ellas, y en muchas ocasiones por comodidad se otorga permisos innecesarios a determinados usuarios, dando acceso a datos que este usuario no tendría que poder ver o a datos confidenciales.

Para ayudar con la administración de estos accesos y controlar los datos que se pueden ver, SQL Server ofrece lo siguiente:

- Mecanismos de autenticación para asegurar que solo los usuarios con credenciales válidas puedan acceder al servidor de base de datos. SQL Server admite la autenticación SQL Server y la autenticación con seguridad integrada de Windows, y en entorno Azure, para Azure SQL Database y Azure SQL Data Warehouse, el control de acceso por roles de Active Directory de Azure.
- Control de acceso basado en roles, otorgando a los usuarios distintos roles (niveles de acceso) dependiendo la función dentro del equipo.
- Row-Level security para evitar el acceso a filas específicas de una tabla basado en características del usuario que intenta acceder al dato, por ejemplo que en la tabla de empleados solo puedas ver tus datos y no los de otros empleados.
- Utilización de enmascaramiento de datos, por ejemplo para números de cuentas bancarias del tipo XXXX-XXXX-XXXX-1321, para ocultar datos a usuarios que no tengan permisos usando Dynamic Data Masking.
- Verificar los cambios que ocurren en una tabla usando SQL Server Audit o para Azure SQL Database utilizando Azure SQL Server Auditing.

En esta entrada vamos a centrarnos en las características más nuevas, disponibles desde SQL Server 2016, como son Row-Level Security y Dynamic Data Masking.

- Row-Level Security. Explicado en el **APARTADO**
- Enmascaramiento DEFAULT. Explicado en el **APARTADO**
- Enmascaramiento EMAIL. Explicado en el **apartado**
- Enmascaramiento PARTIAL. Explicado en el **APARTADO**
- Enmascaramiento RANDOM. Explicado en el **apartado**

Ahora que ya sabemos como controlar los accesos y la visibilidad del dato, es hora de ir al **siguiente punto del GDPR** y ver como **Proteger** estos datos.

Protegiendo datos en Reposo, en Uso y en Tránsito



No basta solo con asegurar el dato en uso, también tenemos que protegerlo en reposo y en tránsito.

¿Por qué esto es importante para el GDPR?

This relates to the GDPR obligation to take into account “risks that are presented by processing, in particular from accidental or unlawful destruction, loss, alteration, [or] unauthorized disclosure of” that data. In this case, the protection is at the level of the physical device—and prevents the risk of compromising the storage itself, for example via copying the physical data out to another server.

GDPR Article 32(2)—“Security of processing.”

Tres estados en los que se puede encontrar el dato:

- **En reposo**, a nivel físico, en nuestro caso podría ser el fichero .mdf, .ldf o los backups que se realizan de la base de datos.
- **En uso**, a nivel lógico, es decir, los datos que encontramos en las tablas de una base de datos.
- **En tránsito**, cuando nuestros datos viajan en forma de paquetes por la red.

Una vez que ya tenemos claros los tres estados, vamos a ver de qué forma SQL Server nos puede ayudar a protegernos:

- Utiliza conexiones cifradas para evitar que los datos viajen (en tránsito) sin encriptar con **Connection Encryption**
- Asegura los datos personales a través del cifrado en la capa física del almacenamiento (en reposo) usando **Transparent Data Encryption**. → **EXPLICADO AQUÍ**
- Evita que usuarios no autorizados o con alto nivel de privilegios accedan al dato en tránsito, en reposo y en uso con la característica **Always Encrypted**.
- Maximiza la disponibilidad del dato y evita caídas del servicio con **Always On Availability Groups**.
- En Azure SQL Database y Azure SQL Data Warehouse detecta actividades anómalas y riesgos potenciales de seguridad con **SQL Database Threat Detection**.

Cómo hacer una entrada que cubra todas estas funcionalidades sería un poco largo, vamos a centrarnos en dos de ellas, que además se pueden desplegar de una forma muy rápida, Connection Encryption y Transparent Data Encryption.

En este ejemplo probamos a activar la encriptación de conexiones (**Connection Encryption**) para que los datos dejen de viajar en texto plano, ya que por defecto en SQL Server solo los usuarios y contraseñas viajan encriptados.

Antes de esto, podemos hacer una prueba rápida con **Wireshark** para entender como “alguien con malas intenciones” vería nuestros datos. Para esto necesitamos conocer la IP de nuestro servidor SQL Server, y el puerto que utilice SQL Server, por defecto es el 1433.

```
C:\Users\BBM.BBMASPACE>IPCONFIG
Windows IP Configuration

Ethernet adapter Ethernet1:

  Connection-specific DNS Suffix . :
  Link-local IPv6 Address . . . . . : fe80::2c41:7306:4580:97da%4
  IPv4 Address. . . . . : 192.168.0.3 [BBM]
  Subnet Mask . . . . . : 255.255.255.248
  Default Gateway . . . . . : 192.168.0.1

Tunnel adapter isatap.{2E58178F-87EF-44EE-A7EB-8978301A82D5}:

  Media State . . . . . : Media disconnected
  Connection-specific DNS Suffix . :

C:\Users\BBM.BBMASPACE>
```

Abriremos Wireshark y pondremos como filtro la siguiente sentencia para ver todos los datos que están pasando por SQL Server:

`ip.addr == YOUR_IP || tcp.port== YOUR_SQL_PORT`

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.3	192.168.0.2	DNS	77	Standard query 0xa282 A www.wireshark.org
2	4.012057	192.168.0.3	192.168.0.2	DNS	85	Standard query 0x2325 A vortex.data.microsoft.com
4	4.4551124	192.168.0.2	192.168.0.3	DNS	77	Standard query response 0xa282 Server failure A www.wireshark.org
8	5.026944	192.168.0.3	192.168.0.2	DNS	85	Standard query 0x2325 A vortex.data.microsoft.com
10	6.026990	192.168.0.3	192.168.0.2	DNS	85	Standard query 0x2325 A vortex.data.microsoft.com
13	8.042807	192.168.0.3	192.168.0.2	DNS	85	Standard query 0x2325 A vortex.data.microsoft.com
17	12.089759	192.168.0.3	192.168.0.2	DNS	85	Standard query 0x2325 A vortex.data.microsoft.com
18	15.449351	192.168.0.2	192.168.0.3	DNS	85	Standard query response 0x2325 Server failure A vortex.data.microsoft.com

Ahora vamos a lanzar una consulta contra nuestro servidor SQL Server, en nuestro caso vamos a utilizar la tabla **[trial].[Usuario_ID]** de BBM_ASPACE.

```
USE [BBM_ASPACE]
GO

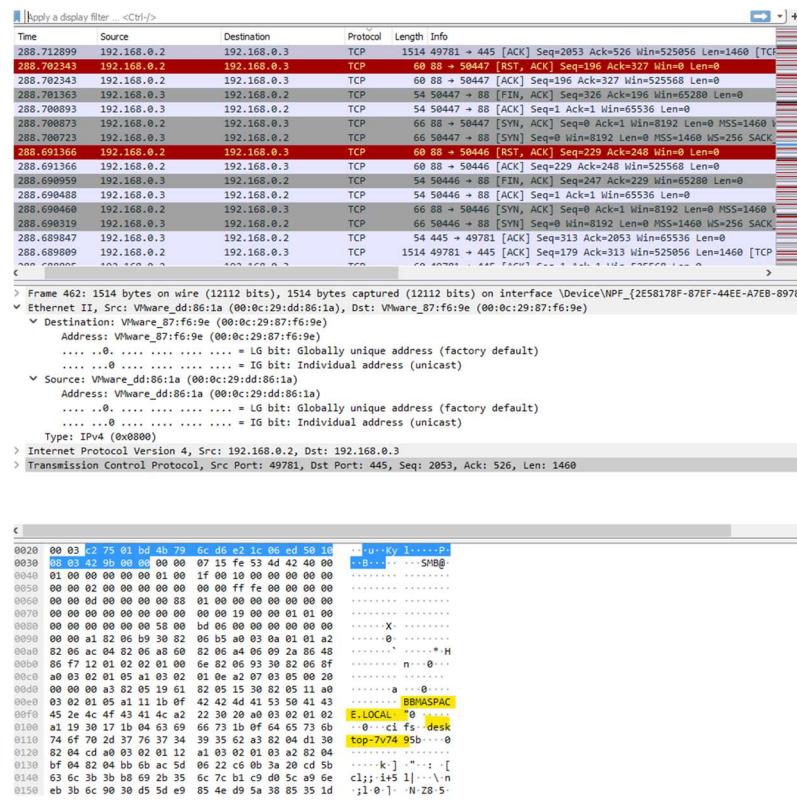
SELECT [Usuario_ID], [Nombre_1], [Nombre_2], [Apellido_1], [Apellido_2],
[DNI], [Otros_Detalles]
FROM [trial].[BBM_Usuario]
GO
```

Y el resultado que nos muestra es el siguiente:

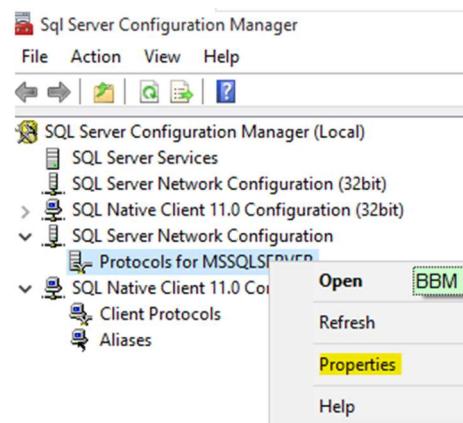
	Results	Messages					
	Usuario_ID	Nombre_1	Nombre_2	Apellido_1	Apellido_2	DNI	Otros_Detalles
1	NO CUMPLE	RICHI	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
2	USERGPP012	PACO		GONZALEZ	PEREZ	111110125	
3	USERTEST	NOMBRETEST	NOMBRE2	APELLIDO1	APELLIDO2	123456789	LOREM
4	USERVNA013	ANA	MARIA	VAZQUEZ	NUNEZ	11111013N	

Nos movemos a Wireshark y buscaremos entre los paquetes capturados, podemos **filtrar por los paquetes que en la columna Protocol ponga TCP**, también nos fijaremos en que la variable **Len** (Longitud) tenga un valor un poco alto.

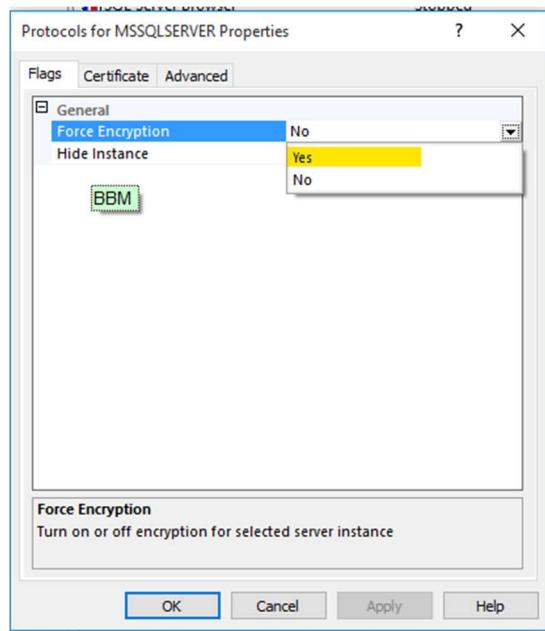
Y una vez tengamos localizado el paquete, solo tendremos que ver su contenido para ver como todos los resultados de nuestra query se muestran sin encriptar:



Ahora para solucionar esto, en nuestro servidor SQL Server abrimos la aplicación **SQL Server Configuration Manager**, y nos dirigimos a las **Propiedades de Protocols for MSSQLSERVER**.



Y cambiamos la opción Force Encryption a Yes.



Ahora reiniciamos el servicio SQL Server y nuestros paquetes ya viajarán encriptados.

Name	State	Start Mode	Log On As	Process ID	Service Type
SQL Server (MSSQLSERVER)	Running	Automatic	NT Service\MSSQLS...	9688	SQL Server
SQL Server Browser	Stopped	Start	... NT AUTHORITY\LO...	0	
SQL Server Agent (MSSQLSERVER)	Stopped	Stop Pause Resume Restart	NT Service\SQLSER...	0	SQL Agent

Para probar esto volvemos a ejecutar la misma consulta, y vemos el resultado que nos aparecerá en Wireshark:

The screenshot shows the Wireshark interface with the following details:

- Panorama View:** Shows a list of 1828 packets captured over "Ethernet1".
- Selected Packet:** The 1514th packet (Frame 1713) is selected. It is a TCP segment from port 49767 to port 1433, Seq: 94490, Ack: 339068, Len: 1460.
- Packet Details:** Shows the structure of the selected TCP segment.
- Hex View:** Displays the raw hex data of the selected packet.
- ASCII View:** Displays the ASCII representation of the selected packet.
- Selected Hex View:** A detailed view of the selected packet's data segment, labeled as "A data segment used in reassembly of a lower-level protocol (tcp.segment_data)".
- Status Bar:** Shows "Packets: 1828 · Displayed: 1828 (100.0%) · Profile: Default".

Monitorizar y Reportar accesos no autorizados a Datos Personales



Este trabajo tiene que ser constante y por eso en este último punto vamos a ver como **Monitorizar e Informar esos posibles accesos no autorizados a los datos personales**.

¿Por qué esto es importante para el GDPR?

The GDPR, as part of the data protection requirement, stipulates a requirement that “Each controller... shall maintain a record of processing activities under its responsibility.”

GDPR Article 30(1)—“Records of processing activities.”

The GDPR has a clear requirement regarding data breaches: “In the case of a personal data breach, the controller shall without undue delay and, where feasible, not later than 72 hours after having become aware of it, notify the personal data breach to the supervisory authority.”

GDPR Article 33(1)—“Notification of a personal data breach to the supervisory authority.”

Para ayudar con la monitorización y el reporte, en SQL Server podemos utilizar varias herramientas:

- **System-Versioned temporal tables**, lo primero destacar que aunque por su nombre lo parezca, **NO ES UNA TABLA TEMPORAL**, es un tipo de tabla de versionado que permite a SQL Server guardar un histórico completo de todos los cambios realizados y una forma rápida de realizar un análisis para saber cómo y cuándo un dato ha cambiado.
- **SQL Server Audit**, para crear auditorías en las capas de servidor y base de datos, que pueden contener diferentes especificaciones para diferentes eventos; creación de logins, copias de seguridad, cambios en una tabla...
- **SQL Alerts and DB Mail**, usando estas dos herramientas, podemos definir nuestras propias alertas para notificar rápidamente una brecha de seguridad, por ejemplo si detectamos que se ha realizado un backup en una localización que no es la usual.

Las herramientas **SQL Server Audit**, **SQL Alerts** y **DB Mail** son viejas conocidas y ya llevan muchas versiones con nosotros, por eso en esta entrada vamos a mostrar como utilizar System-Versioned Temporal Tables que está disponible desde **SQL Server 2016**.

Después de todas estas explicaciones, nuestro proyecto ya será un poco más seguro

ATAQUES

El mundo de la informática es vulnerable de sufrir algún tipo de ataque por terceras personas, con la intención de propagar algún tipo de **malware** o robar información importante de la víctima. Por todo esto, es fundamental tomar las medidas que sean necesarias para mantener a buen recaudo la información.

Dentro de todo esto, las bases de datos son uno de los sistemas que más sufren este tipo de ataques, en gran medida a que es ahí donde en la mayoría de las ocasiones está almacenada la información. Para acceder a ella, los hackers buscan cualquier tipo de vulnerabilidad que no haya sido controlada para acceder al sistema y hacerse con aquello que les sea de interés.

La mayoría de información sensible del mundo está almacenada en sistemas gestores de bases de datos como MySQL, Oracle, **Microsoft SQL Server** entre otros. Toda esa información es la que hace que los hackers centren todo su esfuerzo en poder acceder a esa información por medio de alguna de las muchas vulnerabilidades que nos podemos encontrar referente a estos gestores.

Hasta este momento, gran parte del esfuerzo para mejorar la seguridad de cualquier servicio informático se centraba en asegurar los perímetros de las redes por medio de firewalls, IDS / IPS y antivirus, pero cada vez las organizaciones están poniendo más esfuerzos en la protección de la seguridad de las bases de datos protegiéndolos de intrusiones y cambios no autorizados.

Veamos los tipos de ataques más comunes que tenemos:

DDos

Una "denegación de servicio" (a veces llamada "denegación de servicio distribuida" o DDoS) El ataque ocurre cuando un sistema, en este caso un servidor web, recibe tantas solicitudes a la vez que los recursos del servidor están sobrecargados, el sistema simplemente se bloquea y se apaga. El objetivo y el resultado de un ataque DDoS exitoso son los sitios web en el servidor de destino que no están disponibles para solicitudes de tráfico legítimas.

En muchas ocasiones nos encontramos con entornos de producción donde “demasiada gente” tiene acceso. En muchos casos son accesos de “solo lectura” aunque eso no los hace inofensivos de cara a la estabilidad del sistema. Pero podemos ir más allá, vamos a ver como con un **login**, con rol **public**, sin acceso a ninguna base de datos de usuario, se puede realizar fácilmente un **ataque de denegación de servicio (DoS)** contra SQL Server.

Este ataque se basa en la extenuación de los **workers** de una instancia. Para comprender mejor el funcionamiento del ataque vamos a explicar algunos conceptos a un nivel muy básico:

- **Scheduler:** Tenemos uno por cada core lógico que se le presente a la instancia. Se encarga del acceso en modo cooperativo al core asignado por parte de los workers.
- **Task:** Representa una tarea que hay que realizar. Puede ser desde un login hasta la ejecución de un operador de un plan de ejecución.
- **Worker:** Es la representación lógica de un thread dentro del SQLOS. Son los responsables de ejecutar una tarea en un scheduler.

Con la configuración por defecto (max worker threads = 0) SQL Server crea un número de workers al arrancar y mantendrá un número dinámico de ellos en función de la carga hasta un máximo que se calcula en base a varias variables:

- Sistemas de 32 bits
 - ❖ Cores <= 4 : max worker threads = 256
 - ❖ Cores > 4 : max worker threads = 256 + ((cores - 4) * 8)
- Sistemas de 64 bits
 - ❖ Cores <= 4 : max worker threads = 512
 - ❖ Cores > 4 : max worker threads = 512 + ((cores - 4) * 16)

Es decir que para un equipo con 8 cores de 64 bits el valor de max worker threads será equivalente a $512 + (8-4)*16 = 576$. Podemos obtener el valor calculado para nuestra instancia con la siguiente consulta:

```
USE BBM_ASPACE
SELECT max_workers_count FROM sys.dm_os_sys_info
```

	Results	Messages
		BBM
1	512	

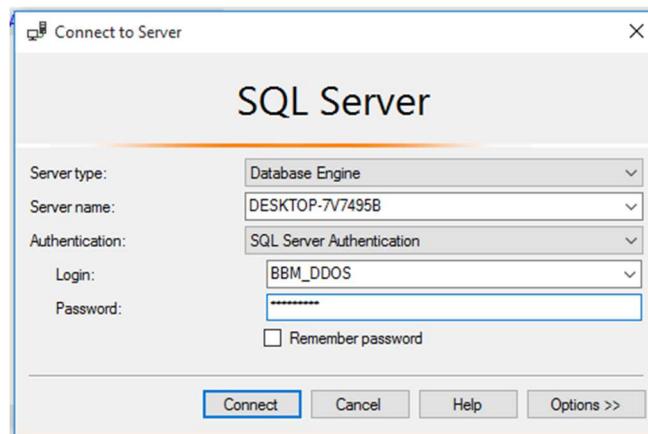
Si decidíramos configurar un número mayor de worker threads debemos tener en cuenta que se necesita una cantidad de memoria importante por cada uno de ellos, por ejemplo en un sistema x64 son casi 2 MB los que necesita cada worker. También comentar que en sistemas migrados desde SQL 2000 nos encontramos en ocasiones el valor 255 configurado tras la migración ya que era el máximo en dicha versión y el proceso de migración no lo modifica.

Desde el momento en que un usuario puede acceder a nuestra instancia, puede comenzar a hacer uso de estos workers que deben ser compartidos para el buen funcionamiento. Incluso el proceso de login depende de la disponibilidad de dichos workers, lo cual hará que la extenuación de éstos produzca que no podamos conectarnos con normalidad a la instancia. El primer paso para esta prueba de concepto será crear un login básico sin ningún permiso adicional:

```
USE [master]
GO

CREATE LOGIN [BBM_DDOS] WITH PASSWORD='Abcd1234.'
GO
```

Con este usuario lo que haremos es una conexión inicial que lanzará una creación de tabla temporal, abrirá una transacción y añadirá una fila a dicha tabla:



```
IF OBJECT_ID('tempdb..##BBMDOS') IS NOT NULL DROP TABLE ##BBMDOS;
CREATE TABLE ##BBMDOS (i INT);

BEGIN TRAN;
INSERT INTO ##BBMDOS (i) VALUES (1)
--(1 row affected)
```

Una vez tengamos esto, lanzaremos desde otra conexión sobre dicha tabla una consulta que, obviamente, quedará bloqueada:

```
SELECT * FROM ##BBMDOS
```

Results		Messages
	i	BBM
1	1	

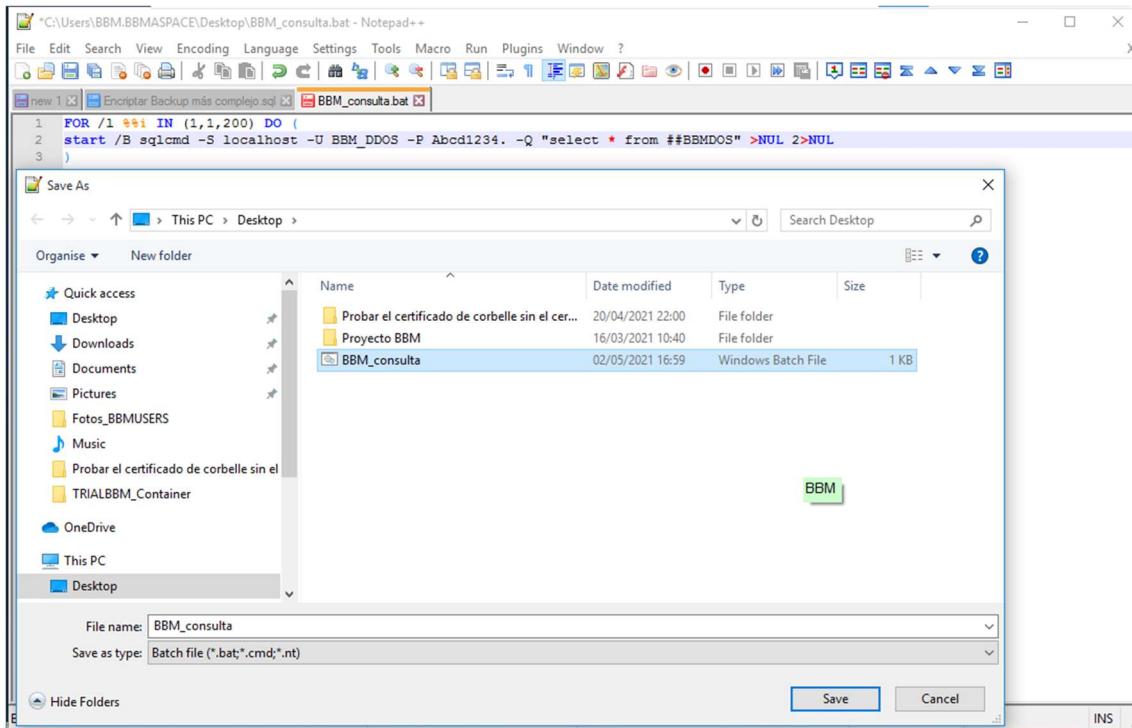
Si lanzamos la siguiente consulta podremos ver el número y estado de cada uno de los workers de nuestra instancia:

```
select COUNT(*), state from sys.dm_os_workers
GROUP BY state
ORDER BY COUNT(*) desc
```

Results		Messages
(No column name)	state	BBM
1	SUSPENDED	
2	RUNNING	
3	INIT	

A continuación lanzaremos por ejemplo 200 consultas sobre la tabla temporal adicionales con algo tan sencillo como un script batch MS-DOS como este:

```
FOR /l %%i IN (1,1,200) DO (
start /B sqlcmd -S localhost -U BBM_DDOS -P Abcd1234. -Q "select * from ##BBMDOS" >NUL 2>NUL
)
```

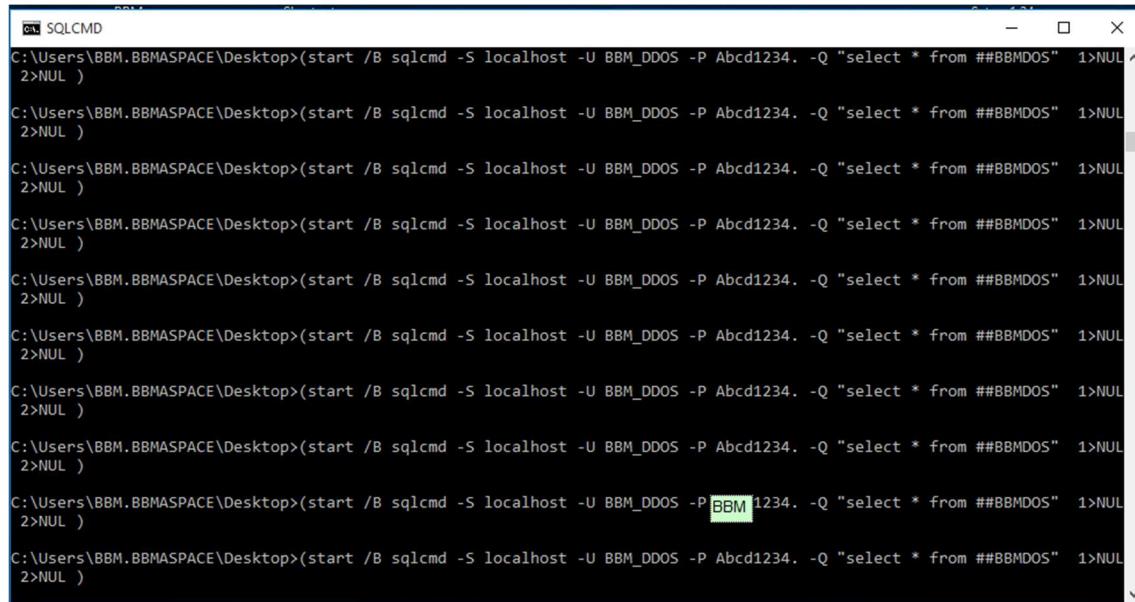
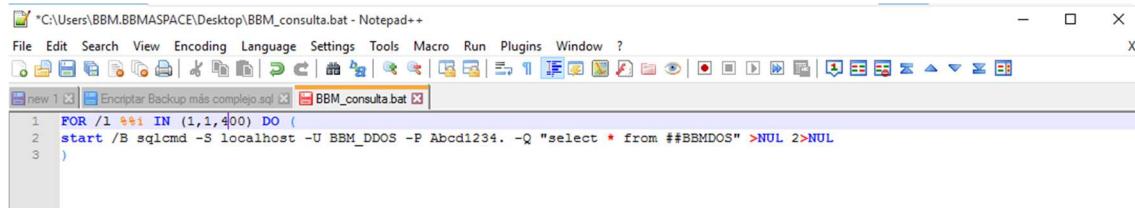


Si volvemos a lanzar la consulta anterior veremos que ahora mismo tenemos más de 200 workers con estado suspended (esperando en este caso a que se libere el bloqueo sobre la tabla):

	(No column name)	state	BBM
1	233	SUSPENDED	
2	10	RUNNING	
3	1	INIT	

Si tenemos en cuenta que en mi entorno de prueba el número máximo de workers era 512, nos bastará con lanzar 400 más para asegurarnos que excedemos el límite:

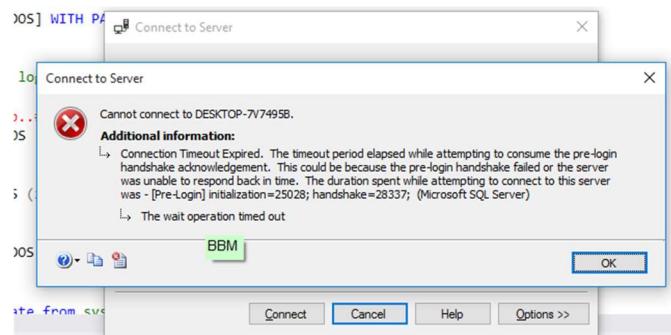
```
FOR /l %%i IN (1,1,400) DO (
start /B sqlcmd -S localhost -U BBM_DDOS -P Abcd1234. -Q "select * from ##BBMDOS" >NUL 2>NUL
)
```



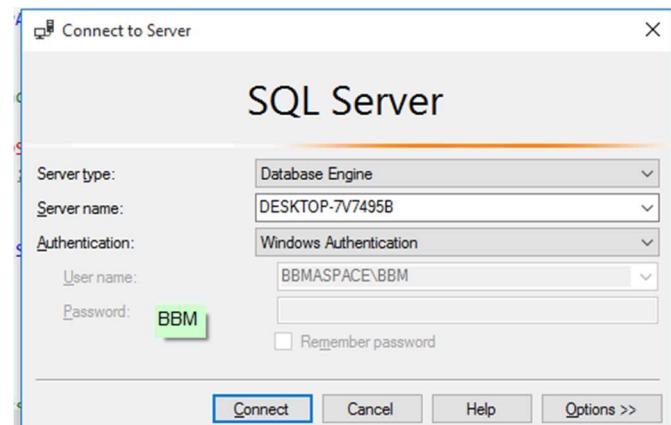
Una vez hayamos lanzado las 600 peticiones comprobaremos que, incluso conexiones ya existentes a la base de datos, no responderán a ningún comando, por sencillo que sea, al no disponer de ningún worker libre:

(No column name)	state	
1	546	SUSPENDED
2	10	RUNNING
3	1	INIT

Si intentamos establecer una nueva conexión contra la instancia, obtendremos un error de login timeout:



Llegados a este punto la única forma de conectar que nos quedaría sería, como administradores y desde la propia máquina (salvo que habilitáramos previamente el acceso remoto a la DAC), conectar mediante la única conexión administrativa disponible. Para ello conectaremos a la instancia del dominio



	(No column name)	state
1	546	SUSPENDED
2	10	RUNNING
3	1	INIT

Una vez conectados obtendremos el estado de los workers con la consulta anterior y vemos que tenemos un número por encima del máximo. La razón es que hay un conjunto de workers extra que pueden ser creados para la DAC y para otros procesos internos:

Una vez que tenemos acceso, el siguiente paso sería mirar si tenemos bloqueos. En nuestro caso como ya sabemos que se trata de un conflicto entre lectores y escritores (select-insert) vamos a buscar aquellos bloqueos concedidos de tipo exclusivo (X) y aquellos de tipo compartido (S) que no están concedidos:

```
SELECT * FROM sys.dm_tran_locks WHERE (request_status='WAIT' AND request_mode='S') OR (request_status='GRANT' AND request_mode='X')
```

resource_type	resource_subtype	resource_database_id	resource_description	resource_associated_entity_id	resource_lock_partition	request_mode	request_type	request_status	request_reference_count	request_lifetime
1 RID	2	3:4:0	115292105226293248	0	X	LOCK	GRANT	0	33554432	
2 KEY	1	fba20ead8244	56294956173824	0	X	LOCK	GRANT	0	33554432	
3 KEY	1	fba20ead8244	56294956173824	0	S	LOCK	WAIT	1	0	
4 KEY	1	fba20ead8244	56294956173824	0	S	LOCK	WAIT	1	0	
5 KEY	1	fba20ead8244	56294956173824	0	S	LOCK	WAIT	1	0	
6 KEY	1	(6e29ed338a70)	281474979463168	0	X	LOCK	GRANT	0	33554432	
7 KEY	1	BBM	(976586af97c)	0	X	LOCK	GRANT	0	33554432	

Podemos ver que la sesión que tiene el bloqueo «cabecera» del problema es la 52 por lo que si la matamos con un **KILL 52** (después de ejecutar el procedimiento almacenado **sp_who2**) permitiremos que el resto ejecuten y, por tanto, se liberen los workers para que el sistema vuelva a responder. Aunque en este caso se trata de un ataque intencionado, la misma situación nos la podemos encontrar si tenemos un bloqueo por un error de aplicación (que deja una transacción abandonada por ejemplo) y comenzamos a tener muchos otros usuarios que van quedando bloqueados por culpa de ello.

```

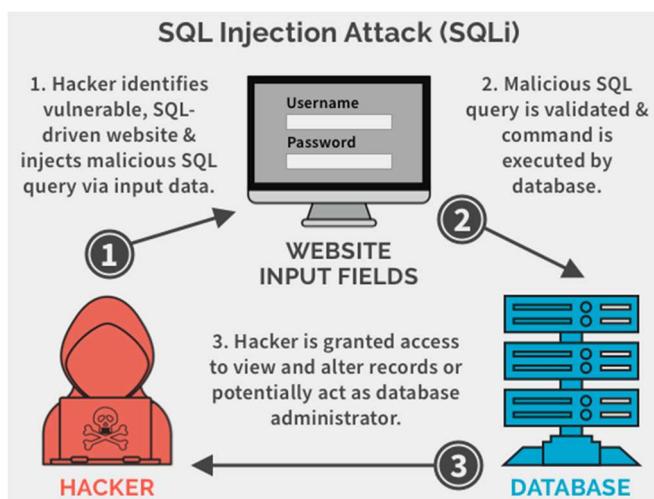
use master -- Todo lo que implica cerrar procesos se hace desde master
go
-- O bien voy al activity monitor y mato el proceso / o utilizo el stored
procedure sp_who2
-- Lo hago y veo los procesos
sp_who2
GO
-- mato el proceso
kill 52
GO

```

Situaciones similares a esta nos las hemos encontrado en producción y en muchos casos los administradores han recurrido, sin ser necesario, a un reinicio del servidor. El reinicio nos solucionaría el problema en este caso pero perderemos información sobre el origen del problema. Si es posible siempre convendría, antes de reiniciar o tomar una medida radical, conectarnos con la DAC y al menos guardar el resultado de algunas DMVs como sys.dm_tran_locks para poder analizar el problema a posteriori.

En conclusión, no debemos permitir acceso directo al SQL Server a ningún usuario a excepción de los administradores. En el caso que, de forma inevitable, tengamos que dar acceso «libre» de lectura a usuarios deberíamos intentar que dicho acceso sea sobre una copia de los datos (réplica transaccional, *logshipping*, réplica de un *availability group* con acceso a lectura, *restore* en otro entorno, clonado de máquina virtual, etc.). El resto de los accesos deberían realizarse mediante aplicaciones correctamente *securizadas* y *controladas*. También deberíamos intentar tener al menos una capa intermedia para acceso a datos que podamos *controlar/securizar* desde IT independientemente de las aplicaciones cliente (más difíciles de *securizar* habitualmente). Evitaremos por tanto las aplicaciones cliente-servidor que ataquen directamente a la base de datos de producción ya que el tienen un riesgo elevado de poder causarnos problemas.

SQL INJECTION



eliminar estos datos, provocando cambios persistentes en el contenido o el comportamiento de la aplicación.

En algunas situaciones, un atacante puede escalar un ataque de inyección SQL para comprometer el servidor subyacente u otra infraestructura de *back-end*, o realizar un ataque de denegación de servicio.

Un ataque de inyección SQL **exitoso** puede resultar en un acceso no autorizado a datos confidenciales, como **contraseñas, detalles de tarjetas de crédito o información personal del usuario**. Muchas violaciones de datos de alto perfil en los últimos años han sido el resultado de ataques de inyección de SQL, lo que ha provocado daños a la reputación y multas reglamentarias. En algunos casos, un atacante puede obtener una puerta trasera persistente en los sistemas de una organización, lo que lleva a un compromiso a largo plazo que puede pasar desapercibido durante un período prolongado.

Tipos de inyección SQL

Existe una amplia variedad de vulnerabilidades, ataques y técnicas de inyección SQL, que surgen en diferentes situaciones. Algunos ejemplos comunes de inyección de SQL incluyen:

- **Recuperación de datos ocultos**, donde puede modificar una consulta SQL para devolver resultados adicionales.
- **Subvirtiendo la lógica de la aplicación**, donde puede cambiar una consulta para interferir con la lógica de la aplicación.
- **Ataques UNION**, donde puede recuperar datos de diferentes tablas de bases de datos.
- **Examinar la base de datos**, donde puede extraer información sobre la versión y estructura de la base de datos.
- **Inyección ciega de SQL**, donde los resultados de una consulta que controlas no se devuelven en las respuestas de la aplicación.

La **inyección SQL** es una vulnerabilidad de seguridad web que permite a un atacante interferir con las consultas que una aplicación realiza a su base de datos. Por lo general, permite que un atacante vea datos que normalmente no puede recuperar. Esto puede incluir datos pertenecientes a otros usuarios o cualquier otro dato al que la propia aplicación pueda acceder. En muchos casos, un atacante puede modificar o

Como detectar las vulnerabilidades

La mayoría de las vulnerabilidades de inyección de SQL se pueden encontrar de forma rápida y confiable utilizando el escáner de vulnerabilidades web de [Burp Suite](#).

La inyección de SQL se puede detectar manualmente mediante el uso de un conjunto sistemático de pruebas en todos los puntos de entrada de la aplicación. Normalmente, esto implica:

- Enviar el carácter de **comilla simple** ' y buscar errores u otras anomalías.
- Enviar alguna sintaxis específica de SQL que evalúe el valor base (original) del punto de entrada y un valor diferente, y buscar diferencias sistemáticas en las respuestas de la aplicación resultante.
- Enviar condiciones booleanas como OR 1 = 1 y OR 1 = 2, y buscar diferencias en las respuestas de la aplicación.
- Enviar cargas útiles diseñadas para desencadenar retrasos cuando se ejecutan dentro de una consulta SQL y buscar diferencias en el tiempo necesario para responder.
- Enviar cargas útiles de OAST diseñadas para desencadenar una interacción de red fuera de banda cuando se ejecuta dentro de una consulta SQL, y monitorear cualquier interacción resultante.

Inyección SQL en diferentes partes de la consulta.

La mayoría de las vulnerabilidades de inyección de SQL surgen dentro de la cláusula *WHERE* de una consulta *SELECT*. Este tipo de inyección SQL generalmente es bien entendido por probadores experimentados.

Pero, en principio, las vulnerabilidades de inyección de SQL pueden ocurrir en cualquier ubicación dentro de la consulta y dentro de diferentes tipos de consultas. Las otras ubicaciones más comunes donde surge la inyección de SQL son:

- En declaraciones *UPDATE*, dentro de los valores actualizados o la cláusula *WHERE*.
- En declaraciones *INSERT*, dentro de los valores insertados.
- En sentencias *SELECT*, dentro del nombre de la tabla o columna.
- En sentencias *SELECT*, dentro de la cláusula *ORDER BY*.

Algunas características centrales del lenguaje SQL se implementan de la misma manera en las plataformas de bases de datos populares, y muchas formas de detectar y explotar las vulnerabilidades de inyección de SQL funcionan de manera idéntica en diferentes tipos de bases de datos.

Sin embargo, también existen muchas diferencias entre las bases de datos comunes. Esto significa que algunas técnicas para detectar y explotar la inyección de SQL funcionan de manera diferente en diferentes plataformas. Por ejemplo:

- Sintaxis para la concatenación de cadenas.
- Comentarios.
- Consultas por lotes (o apiladas).
- API específicas de la plataforma.
- Error de mensajes.

Prevenir la inyección SQL

Entrenar y mantener la conciencia → Todos los implicados en la creación de la aplicación web deben conocer los riesgos asociados a estas vulnerabilidades.

No confiar en ninguna entrada de usuario → Cualquier entrada de usuario que utiliza autenticación SQL representa un potencial riesgo de inyección SQL. Es recomendable tratar a todas las entradas del usuario como si no fuesen de confianza.

Usar listas blancas → Es recomendable filtrar las entradas de usuario mediante listas blancas en vez de listas negras.

Utilizar las últimas tecnologías → Muchas tecnologías de desarrollo web antiguas no cuentan con protección SQLi. Utilizar la versión más reciente del entorno de desarrollo y el lenguaje, así como las tecnologías asociadas a ellos.

Emplear mecanismos verificados → La mayoría de las tecnologías de desarrollo modernas te ofrecen mecanismos de protección contra ataques SQLi. Por ejemplo, es mejor utilizar consultas parametrizadas o procedimientos almacenados que aseguren su seguridad.

Escaneos regulares → Las inyecciones SQL pueden ser introducida por los desarrolladores o a través de bibliotecas/módulos/softwares externos. Lo más recomendable es mantener un escaneo constante utilizando herramientas certificadas y profesionales.

Ransomware

El ransomware es un tipo de software malicioso que bloquea el acceso a los archivos del usuario hasta que se **paga un rescate**. Un "mecanismo de bloqueo" bastante común es cifrar todos los archivos en el PC de un usuario, servidor o incluso en toda la red.

Ese cifrado es un proceso de codificación de datos originales mediante una clave de cifrado. Luego, se requiere la misma clave para descifrar los datos codificados a su formato original. Sin la clave original, los datos se vuelven inutilizables. Todos los archivos de la computadora, servidor o incluso toda la red se convierten en un montón de basura. La única forma de descifrar esa basura es obtener la clave de cifrado original pagando un rescate.

¿Cómo se propaga el ransomware?

De manera similar a cualquier otro virus informático, el ransomware se propagará a través de correos electrónicos de phishing, sitios web infectados y medios físicos como controladores USB.

Si un PC se infecta, puede extenderse a todos los recursos de red a los que tiene acceso su cuenta.

Cómo protegerse contra el ransomware

Tuve la suerte de haber pasado gran parte de mi carrera en centros de datos de nivel empresarial con miles de servidores donde la seguridad era la máxima prioridad. Con suerte, encontrará útiles mis consejos.

- **Separar el acceso** → No usar la misma cuenta de *Active Directory* para las actividades diarias de escritorio, como navegar por Internet, correo electrónico y acceso al servidor. Utilizar la cuenta dedicada para cualquier acceso al servidor y no convertir

la cuenta de administrador de sistemas de SQL Server en una cuenta de administrador local en Windows.

No use las cuentas de administrador para navegar por Internet, etc y nunca otorgar acceso privilegiado a los usuarios finales.

- **Separar las redes** → Aislara la red física o lógica (VLAN) y colocar los servidores en una red separada de sus estaciones de trabajo y otros dispositivos con acceso a Internet. Restringir la capacidad de sus servidores para conectarse a Internet. Nunca se pueden poner los servidores de base de datos en DMZ. De esta manera, está mitigando la exposición del sistema operativo subyacente a una red pública y solo abre puertos SQL en el primer firewall de DMZ
- **Parchear servidores** → una buena práctica es siempre mantener actualizado todo con sus correspondientes parches
- **Hacer copias de seguridad** → Asegurar una copia de seguridad de los usuarios, trabajos de agentes y objetos del sistema. También de las claves de cifrado para cualquier base de datos habilitada para TDE

Una segunda copia de la copia de seguridad local en una red local o en un dispositivo separado.

Una tercera copia de la copia de seguridad en una ubicación externa, podría ser un almacenamiento en línea. Ningún *malware* no podrá acceder al almacenamiento en línea sin claves de acceso y, por lo tanto, esta sería una opción bastante segura.

NOTA: Las copias de seguridad dañadas no sirven.

Cómo recuperarse de un desastre

Tener copias de seguridad válidas es el primer paso hacia el éxito. El segundo paso es tener servidores para restaurar estas copias de seguridad. En los casos en los que servidores completos se vieron afectados y tuvieron que reconstruirse, aprovisionarse o restaurarse, esto podría llevar días o semanas. A menudo, se utiliza un centro de datos secundario para la recuperación ante desastres para acelerar el proceso de recuperación. Servirá la carga hasta que el centro de datos principal vuelva a estar en funcionamiento.

En cualquier caso, el enfoque de recuperación ante desastres dependerá de muchos factores diferentes y de su infraestructura; sin embargo, sin importar cuáles sean, debe realizar pruebas de recuperación ante desastres al menos algunas veces al año para asegurarse de que:

- Sabe cómo reconstruir un servidor e idealmente tenerlo automatizado.
- Sabe cómo restaurar bases de datos, especialmente bases de datos y certificados habilitados para TDE.
- El equipo de aplicaciones sabe cómo restaurar todas las aplicaciones y que se comunican con las bases de datos.
- Si hay un centro de datos secundario, todos deben asegurarse de que los servicios puedan fallar y que todos sepan cómo hacerlo.
- Todo el mundo sabe qué hacer en una situación crítica.

Es muy importante tener todo procedimentado, ya que la situación de recuperación será bajo presión y estrés.

Tools

DDoS

- A nivel **infraestructura**, en el caso de servidores Windows es recomendable configurar ciertos registros que controlan la pila TCP/IP para evitar ataques “SYN flood”. Tales registros son:
 - ❖ **SynAttackProtect** → Determina si la función de protección contra ataques de inundación SYN de TCP/IP está habilitada. La protección contra ataques de inundación SYN está habilitada cuando el valor de esta entrada es 1 y el valor de la entrada TcpMaxConnectResponseRetransmissions (Determina cuántas veces TCP retransmite un SYN-ACK sin respuesta) es al menos 2.
 - ❖ **TcpMaxPortsExhausted** → Determina cuántas solicitudes de conexión puede rechazar el sistema antes de que TCP/IP inicie la protección contra ataques de inundación SYN.
 - ❖ **TcpMaxHalfOpen** → Determina cuántas conexiones puede mantener el servidor en estado semiabierto (SYN-RCVD) antes de que TCP/IP inicie la protección contra ataques de inundación SYN

Además de para Windows, también existen soluciones similares para Linux tales como:

- ❖ **SYN-Cookies** → Syn-proxy es un protocolo propio. Un dispositivo proxy cualquier solicitud TCP, pero sólo lo remitirá al servidor si el apretón de manos de tres vías está completamente establecido.
- ❖ **SYN-Cache** → Dos defensas del host final, llamadas SYN caches y SYN cookies, operan reduciendo la cantidad de estado asignado inicialmente para un TCB generado por un SYN recibido, y posponiendo la instanciación del estado completo
- ❖ **SYN-Proxy** → En contraste con el enfoque de la caché SYN, la técnica de las cookies SYN hace que se genere un estado absolutamente nulo por un SYN recibido.

Este tipo de soluciones, debido a su necesidad de procesamiento, es recomendable instanciarlas en dispositivos que no sean el propio servidor que aloja el servicio principal.

A la hora de configurar un servidor para que aloje nuestra página web, es necesario dotarlo de diferentes medidas de seguridad tales como un **cortafuego**.

- A nivel “**Web Application Firewalls**” → Este tipo de firewalls pueden ser instalados en nuestro servidor como tal, o en otro que esté integrado en nuestra red. Esto hay que tenerlo muy en cuenta, debido a que también consume capacidad de procesamiento de las máquinas ya que tiene que procesar las peticiones con las reglas que le definamos antes de entregarlas al servidor web, por lo que en el caso de estar recibiendo un ataque y que el servicio WAF falle o se degrade, también puede hacerlo nuestra página web.

Existe otra forma hacerlos participes en la seguridad de nuestra red ya que podemos encontrar proveedores que ofrecen el servicio en remoto, es decir, en “cloud”. La manera en cómo funciona esta arquitectura es que la compañía a la que le contratás los servicios despliega el cortafuegos en sus servidores, dirigiendo y procesando allí el tráfico del servicio web antes de ser enviado a tu servidor ya sin ninguna amenaza. Algunos de los proveedores de estos servicios son **Akamai**, **CloudFlare** y **Sucuri**, entre otros.

Además de bloquear ataques de denegación de servicio (DoS), los WAFs también son capaces de detectar y bloquear ataques como son “Cross-Site Scripting” o “SQL injection”.

En algunos casos los WAFs son una solución bastante óptima y sencilla, además puede utilizarse con una configuración “semi-automática”, debido a que los servicios de este tipo cuentan con configuraciones básicas que ya podrían proteger los servicios web de ataques de denegación de servicio. En cualquier caso, es aconsejable definir reglas de configuración a medida para adaptar el WAF a nuestra infraestructura y obtener una mejor protección aprovechando al máximo las características del WAF.

Una modalidad de ataques de denegación de servicio (DoS) son los distribuidos. Los ataques de denegación de servicio distribuido (DDoS) son realizados a través de, por lo general, **botnets**.

- A nivel **aplicaciones web**, la mayoría de las ocasiones los ataques de denegación de servicio dirigido a aplicaciones web no se llevan a cabo a través de la sobrecarga del sistema, saturación del servicio o agotamiento del ancho de banda, sino a través de la explotación de vulnerabilidades en nuestra aplicación, por lo que la regla de seguridad más importante es instalar lo antes posible las actualizaciones de seguridad que sean publicadas y que solucionen posibles problemas de seguridad en la aplicación que utilicemos en nuestra web. En el caso de utilizar aplicaciones con un desarrollo **ad hoc**, es necesario la realización de **auditorías** para identificar problemas de seguridad y de esta manera solucionarlos.

Es muy recomendable disponer del sistema **CAPTCHA** en los formularios de nuestra web, de esta manera no será posible ejecutar un ataque automatizado a través de ellos.

SQL Injection

Puede utilizar **HP Scrawlr**, **URLScan** o **Microsoft Source Code Analyzer for SQL Injection** para buscar en su sitio web y en la base de datos de SQL Server las vulnerabilidades que podrían poner su entorno en riesgo de sufrir un ataque de inyección SQL. Se pueden evitar los ataques de inyección SQL filtrando los campos de entrada en sus páginas web para que los usuarios sólo puedan escribir ciertos valores en el campo de entrada. Diseñar y desarrollar su aplicación teniendo en cuenta la seguridad también puede ayudar a prevenir los ataques de inyección SQL.

Los ataques de inyección SQL no sólo se producen en las bases de datos de SQL Server que respaldan las aplicaciones frontales de ASP y ASP.NET; también pueden producirse en aplicaciones PHP con back-end MySQL y en aplicaciones Java con back-end Oracle. Todas las plataformas de bases de datos son vulnerables a los ataques de inyección SQL. Puede evitar estos ataques filtrando los campos de entrada en sus páginas web para que sólo se permitan los valores permitidos.

Debido al aumento de los ataques de inyección SQL, Microsoft ha publicado un importante aviso de seguridad que señala las siguientes tres importantes herramientas que puede utilizar con sus aplicaciones ASP y ASP.NET para prevenir los ataques de inyección SQL:

- **HP Scrawlr**: esta utilidad de escaneo gratuita puede detectar e identificar si su sitio web es susceptible de sufrir un ataque de inyección SQL. La utilidad rastrea un sitio

web, analizando los campos de entrada de cada página web en busca de vulnerabilidades de inyección SQL a medida que avanza (no funciona contra JavaScript, el análisis sintáctico de Flash o los parámetros POST).

- **URLScan:** esta herramienta de seguridad restringe activamente el tipo de solicitudes HTTP que Microsoft IIS procesará. URLScan no sustituye a la programación adecuada de una aplicación web, pero puede evitar que algunas solicitudes potencialmente dañinas lleguen a la aplicación web y a SQL Server. Funciona en IIS 5.1 y posteriores, incluido IIS 7.0 para Windows Server 2008.
- **Microsoft Source Code Analyzer for SQL Injection:** esta herramienta de línea de comandos analiza su código fuente ASP estático escrito en VBScript (no en ASP.NET) y revela posibles vulnerabilidades a los ataques de inyección SQL. A continuación, la herramienta genera un informe en el que se detallan las vulnerabilidades detectadas y las posibles soluciones.

Aunque cada una de estas herramientas puede ayudar a evitar que un atacante penetre y dañe sitios web y las bases de datos SQL Server, ninguna de ellas es tan eficaz como diseñar y desarrollar una aplicación teniendo en cuenta la seguridad. La inyección SQL es un viejo estilo de piratería de sitios web, y es bastante fácil de prevenir cuando las comprobaciones de valores se escriben en el código desde el principio.

Ransomware

Con esto, el único modo de proteger tanto a los individuos como a las instituciones de pérdidas monetarias que supone un ataque de estas características es obtener un programa de seguridad profesional que pueda defender las redes y sistemas de ataques ransomware antes de que entren estos virus. Aunque la mayoría de los programas anti-ransomware robustos pueden ser comprados, tienen versiones de prueba para que sus clientes comprueben y evalúen su efectividad.



- **La herramientas anti-malware Free Bitdefender:** este programa de eliminación de ransomware usa una técnica de vacuna para proteger el ordenador de específicas familias de ransomware. Particularmente, ayuda a evitar *TeslaCrypt*, *Locky*, *BTC-Locker*, y la versión original del cripto-malware *Petya*. El método que usa es similar al de las vacunas en la vida real → el anti-malware engaña al ordenador y le hace pensar que el virus ya lo ha infectado.

Ya que los criminales no suelen infectar equipos que ya están infectados con programas maliciosos, esta herramienta de eliminación de ransomware hace lo que promete. La cosa es que reinfectar el PC puede hacer que los archivos bloqueados sean incluso indescifrables para los hackers que han creado el ransomware. Por ello, intentan evitar estas acciones para mantener la esperanza de sus víctimas de que ellos sí pueden hacerlo.

La herramienta anti-malware **Bitdefender** crea un archivo que funciona como un incentivo de que el ordenador ya está infectado. De este modo, desvía los ataques. No obstante, esta técnica puede no funcionar siempre, y los desarrolladores de programas lo han diseñado para ejecutarlo con el inicio y ejecutarlo en el segundo plano para obtener protección extra.

Otra característica de esta herramienta de eliminación de ransomware es que es compatible con cualquier ordenador y no requiere ninguna configuración comercial. Sin embargo, solo quedas protegido contra estas cuatro familias ransomware. Por ello, se recomienda usar otro anti-malware para defenderse de otros ataques.

NOTA: Aunque las herramientas descritas protegen contra ataques, es recomendable usar otro programa de seguridad para protegerte contra otras ciber amenazas para proteger contra 4 familias específicas de ransomware.

- Otra herramienta es **Malwarebytes Anti-ransomware**, que detecta cripto-malwares basados en el comportamiento. Este programa de eliminación de ransomware es compatible con estos sistemas operativos:

- Windows 10 (32/64-bit)
- Windows 8.1 (32/64-bit)
- Windows 8 (32/64-bit)
- Windows 7 (32/64-bit)



Malwarebytes Anti-ransomware Beta protege a sus usuarios de ordenador controlando el acceso a importantes dominios o a tipos específicos de archivos. En otros términos, si un programa no autorizado intenta modificar o alcanzar un archivo o localización controlada, el programa de seguridad alerta al usuario sobre un potencial ataque ransomware.

NOTA: programas como Microsoft Office o los componentes de Windows no son reconocidos como amenazas. No obstante, otros pueden encender la luz roja. Por fortuna, el usuario es capaz de añadir a una lista blanca una aplicación con un simple clic así como meterla en cuarentena.

Por desgracia, puede haber un caso donde el ransomware se **impersona** como programa en la lista blanca y alcance silenciosamente tu sistema. Por esta razón, los expertos categorizan esta herramienta como una gran capa de protección extra junto al programa de seguridad principal.

- **Kaspersky Lab** ofrece una gran herramienta de eliminación de ransomware gratuita. **Kaspersky Anti-ransomware** soporta los siguientes sistemas operativos:

- Windows 7
- Windows 8
- Windows 8.1
- Windows 10



Este programa anti-ransomware tiene una gran combinación con el modelo **Kaspersky's System Watcher** junto con el control de comportamiento para una protección completa. También permite a los usuarios insertar programas en una lista blanca que puede, por desgracia, dejar que algunos programas maliciosos se infiltrén en el sistema con la ayuda de engañosas técnicas. No obstante, la herramienta Kaspersky Anti-ransomware ofrece una recuperación de archivos comprometidos en caso de que el PC quede infectado con un ransomware.

Otra excelente característica de este programa de seguridad es que incluso si el virus ransomware logra bloquear la aplicación, ésta se abre instantáneamente y comienza a ejecutarse automáticamente. Hay que saber también que este programa no puede

funcionar junto a otras aplicaciones de Kaspersky, lo cual sería una gran segunda capa de protección.

- **Trend Micro RansomBuster** proporciona múltiples capas de protección. Este programa es gratuito. Su principal característica es que está diseñada para crear copias de seguridad en cuanto su comportamiento controlando enciende una luz roja. De este modo, si la alerta es válida, el programa de seguridad elimina la ciber amenaza y recupera los archivos desde copias de seguridad que ha hayan sido previamente realizadas. Es compatible con estos sistemas:

- Windows 7
- Windows 8
- Windows 8.1
- Windows 10



Adicionalmente, este programa tiene otra habilidad llamada *Folder Shield* que permite proteger los documentos más importantes de modificaciones no autorizadas.



- **Zemana Anti-malware** tiene tanto versiones premium como gratuitas, lo cual difiere en cuanto a las funciones extra. Mientras que la versión gratuita consiste en comprobaciones inteligentes y en la opción arrastrar y soltar, la versión completa ofrece protección a tiempo real, así como comprobaciones profundas. Otra gran función incluida en ambas variantes es la capacidad de contactar 24/7 con el soporte técnico.

Las comprobaciones inteligentes corresponden a detecciones rápidas que comprueban solo los directorios más comunes donde suelen alojarse los ransomwares. También toman poco tiempo las comprobaciones más profundas, las cuales normalmente examinan todos los archivos en el ordenador. Soporta los siguientes sistemas operativos:

- Windows XP with Service Pack 2 o más (32bit y 64bit);
- Microsoft Windows Vista (32bit y 64bit);
- Microsoft Windows 7 (32bit y 64bit);
- Microsoft Windows 8 (32bit y 64bit);
- Microsoft Windows 8.1 (32bit y 64bit);
- Microsoft Windows 10 (32bit y 64bit).

Otra gran función de Zemana Anti-malware es que ambas versiones proporcionan la habilidad de arrastrar y soltar archivos para la comprobación. De acuerdo con las pruebas, esta función es beneficiosa para aquellos que estén usando una versión gratuita, ya que es una exploración más profunda que la comprobación inteligente presente en la versión premium. También está considerado por ser una opción media para aquellos que tengan la versión completa y quieran solo comprobar unos archivos particulares.

Además, es capaz de detectar y eliminar rootkits y bootkits del sistema, así como de prevenir futuras infecciones. Al contrario que otras herramientas anti-ransomware, esta elimina varios programas potencialmente no deseados (PUPs), incluyendo hackers

de navegador, spywares y programas adwares. Por ello, es una buena opción para aquellos que deseen usar un programa para obtener seguridad completa.

- **Reimage** es el único programa que usa su enorme base de datos de archivos de Windows y descarga e instala inmediatamente los archivos rotos, dañados o pendientes en tu sistema operativo. Es incluso capaz de reinstalar por completo Windows sin requerir reinstalar las aplicaciones.



Es una de las mejores opciones cuando se trata de eliminar programas potencialmente no deseados (PUPs), como hackers de navegador, adwares, spywares o incluso ciber amenazas como gusanos, rootkits o troyanos.

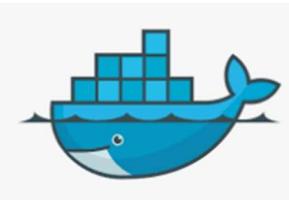
Los ordenadores ejecutando las siguientes versiones de los sistemas operativos son compatibles con Reimage:

- ⊕ Windows XP (32bit);
- ⊕ Windows Vista (32 & 64bit);
- ⊕ Windows 7 (32 & 64bit);
- ⊕ Windows 8 (32 & 64bit);

Proporciona estabilidad al ordenador y muestra un gran ratio de detección contra malware tanto en Mac como en Windows. Sin embargo, los usuarios deben ser conscientes de que este programa no repara ni reemplaza archivos de programas de terceras partes.

DOCKER

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos.



La palabra "DOCKER" se refiere a varias cosas. Esto incluye un proyecto de la comunidad *open source*; las herramientas del proyecto *open source*; Docker Inc., la empresa que es la principal promotora de ese proyecto; y las herramientas que la empresa admite formalmente. El hecho de que las tecnologías y la empresa comparten el mismo nombre puede ser confuso.

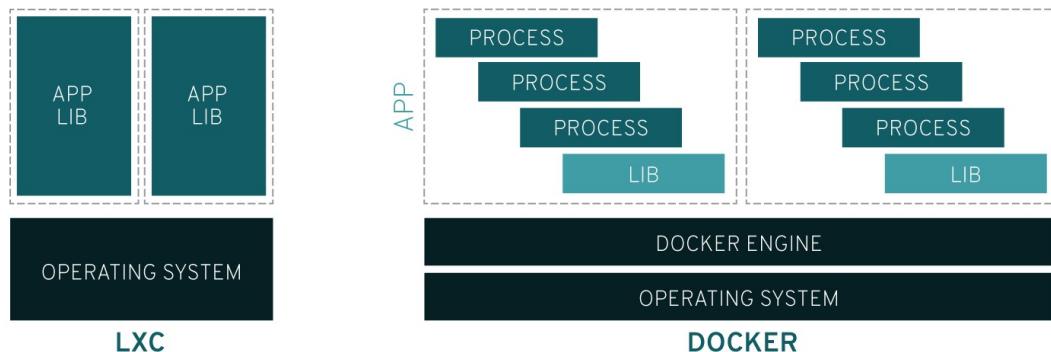
La tecnología Docker usa el kernel de Linux y las funciones de este, como Cgroups y namespaces, para separar los procesos, de modo que puedan ejecutarse de manera independiente. El propósito de los contenedores es esta independencia: la capacidad de ejecutar varios procesos y aplicaciones por separado para hacer un mejor uso de su infraestructura y, al mismo tiempo, conservar la seguridad que tendría con sistemas separados.

Las herramientas del contenedor, como Docker, ofrecen un modelo de implementación basado en imágenes. Esto permite compartir una aplicación, o un conjunto de servicios, con todas sus dependencias en varios entornos. Docker también automatiza la implementación de la aplicación (o conjuntos combinados de procesos que constituyen una aplicación) en este entorno de contenedores.

Estas herramientas desarrolladas a partir de los contenedores de Linux, lo que hace a Docker fácil de usar y único, otorgan a los usuarios un acceso sin precedentes a las aplicaciones, la capacidad de implementar rápidamente y control sobre las versiones y su distribución.

Al principio, la tecnología Docker se desarrolló a partir de la tecnología LXC, lo que la mayoría de las personas asocia con contenedores de Linux "tradicionales", aunque desde entonces se ha alejado de esa dependencia. LXC era útil como virtualización ligera, pero no ofrecía una buena experiencia al desarrollador ni al usuario. La tecnología Docker no solo aporta la capacidad de ejecutar contenedores; también facilita el proceso de creación y diseño de contenedores, de envío de imágenes y de creación de versiones de imágenes (entre otras cosas).

Traditional Linux containers vs. Docker



Los contenedores de Linux tradicionales usan un sistema *init* que puede gestionar varios procesos. Esto significa que las aplicaciones completas se pueden ejecutar como una sola. La tecnología Docker pretende que las aplicaciones se dividan en sus procesos individuales y ofrece las herramientas para hacerlo. Este enfoque granular tiene sus ventajas.

Las características de Docker son las siguientes:

- **Modularidad** → El enfoque Docker para la creación de contenedores se centra en la capacidad de tomar una parte de una aplicación, para actualizarla o repararla, sin necesidad de tomar la aplicación completa. Además de este enfoque basado en los microservicios, puede compartir procesos entre varias aplicaciones de la misma forma que funciona la arquitectura orientada al servicio (SOA).
- **Control de versiones de imágenes y capas** → Cada archivo de imagen de Docker se compone de una serie de capas. Estas capas se combinan en una sola imagen. Una capa se crea cuando la imagen cambia. Cada vez que un usuario especifica un comando, como ejecutar o copiar, se crea una nueva capa.

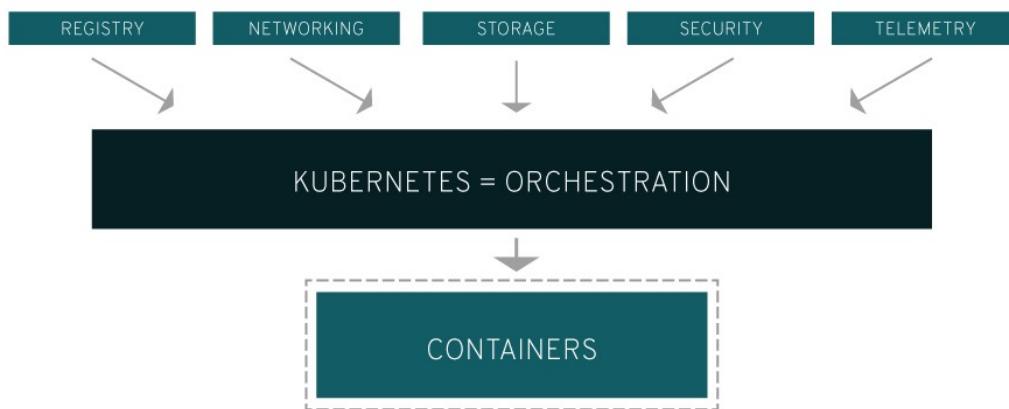
Docker reutiliza estas capas para construir nuevos contenedores, lo cual hace mucho más rápido el proceso de construcción. Los cambios intermedios se comparten entre imágenes, mejorando aún más la velocidad, el tamaño y la eficiencia. El control de versiones es inherente a la creación de capas. Cada vez que se produce un cambio nuevo, básicamente, usted tiene un registro de cambios incorporado: control completo de sus imágenes de contenedor.

- **Restauración** → Probablemente la mejor parte de la creación de capas es la capacidad de restaurar. Toda imagen tiene capas. ¿No le gusta la iteración actual de una imagen? Restáurela a la versión anterior. Esto es compatible con un enfoque de desarrollo ágil y permite hacer realidad la integración e implementación continuas (CI/CD) desde una perspectiva de las herramientas.
- **Implementación rápida** → Solía demorar días desarrollar un nuevo hardware, ejecutarlo, proveerlo y facilitarlo. Y el nivel de esfuerzo y sobrecarga era extenuante. Los contenedores basados en Docker pueden reducir el tiempo de implementación a segundos. Al crear un contenedor para cada proceso, puede compartir rápidamente los procesos similares con nuevas aplicaciones. Y, debido a que un SO no necesita iniciarse para agregar o mover un contenedor, los tiempos de implementación son sustancialmente inferiores. Además, con la velocidad de implementación, puede crear y destruir la información creada por sus contenedores sin preocupación, de forma fácil y rentable.

Por lo tanto, la tecnología Docker es un enfoque más granular y controlable, basado en microservicios, que prioriza la eficiencia.

Al comenzar a utilizar cada vez más contenedores y aplicaciones en contenedores, divididas en cientos de piezas, la gestión y la organización se pueden tornar muy difíciles. Finalmente, debe retroceder y agrupar los contenedores para ofrecer servicios, como redes, seguridad, telemetría, etc., en todos sus contenedores. Es aquí donde aparece **Kubernetes**.

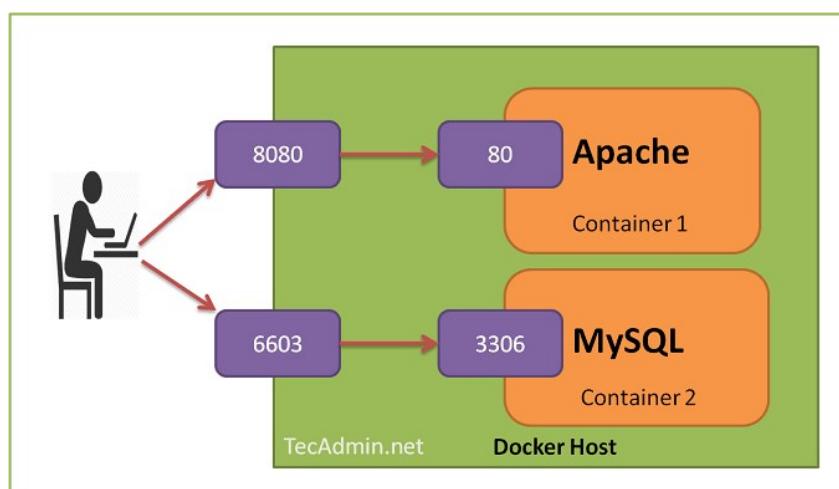
Son una plataforma open source que automatiza las operaciones de los contenedores de Linux. Elimina muchos de los procesos manuales involucrados en la implementación y escalabilidad de las aplicaciones en contenedores. En otras palabras, puede crear un clúster de grupos de hosts que ejecutan contenedores de Linux, y Kubernetes lo ayuda a administrar con facilidad y eficacia esos clústeres. Estos clústeres pueden abarcar hosts en nubes públicas, privadas o híbridas.



Una de las características más importantes de Docker, es que todo el mundo puede crear imágenes para todo tipo de propósitos. Algunas de ellas son subidas a **Docker Hub**, donde estarán a nuestra disposición y las podremos descargar. Además, en **Docker Hub** también encontraremos imágenes oficiales a parte de las imágenes de terceros.

Las imágenes Docker son plantillas (que incluyen una aplicación, los binarios y las librerías necesarias) que se utilizan para construir contenedores Docker y ejecutarlos. Si has trabajado con lenguajes orientados a objetos, son como (valga la redundancia) objetos, una base y estructura para crear una nueva aplicación con características ya establecidas.

Una de las maravillas de Docker, entre sus características está la de permitir exponer un puerto de tu aplicación al resto de tu equipo. La forma en que funciona Docker es que **tiene su propia red interna**, aislada del resto del equipo, tiene la opción de exponer esos puertos “como un espejo” (se asigna un puerto de la computadora a uno de la red de docker).



Se pueden tener las bases de datos (SQL Server, PostgreSQL, MySQL, etc.) en servidores independientes regulares, en clústeres locales o en los servicios **PaaS** en la nube como **Azure SQL DB**. Sin embargo, en los entornos de desarrollo y prueba, el hecho de que las **bases de datos se ejecuten como contenedores** resulta práctico, ya que no tiene ninguna dependencia externa y solo con ejecutar el comando `docker-compose up` ya se inicia toda la aplicación. Tener esas bases de datos como contenedores también es muy útil para las pruebas de integración, porque la base de datos se inicia en el contenedor y siempre se rellena con los mismos datos de ejemplo, por lo que las pruebas pueden ser más predecibles.

Comandos básicos

docker -v. Comprobar la versión de Docker.

docker run <contenedor>. Ejecutar el contenedor.

docker. Muestra todos los comandos disponibles de Docker

docker <options> COMMAND. Muestra todos los comandos disponibles de Docker dentro de las opciones.

docker ps. Muestra que contenedores están en ejecución

docker ps -a. Muestra que contenedores están en ejecución y los que no

docker image ls. Muestra imágenes presentes en el sistema

docker container ls. Vemos los contenedores que tenemos funcionando.

docker container logs <hash> Nos mostrara el log del container que hemos pedido.

docker info Muestra información de las imágenes, tamaño , fecha creación...

docker-compose up -d // Levantar la maquina con un fichero `docker-compose` (hay que entrar en la carpeta donde este el fichero)

docker-compose down Desmontar la imagen

docker search <name> Para buscar in Docker

docker inspect <friendly-name|container-id>. Devolver información de bajo nivel sobre objetos Docker

docker start. Inicie uno o más contenedores detenidos

docker stop. Detenga uno o más contenedores en ejecución

docker pull <nombreImagen> Este comando sirve para descargar una imagen

docker push <nombreImagen> Este comando sirve para subir una imagen

docker rmi <imagen>. Eliminar una o más imágenes. Con la letra **-f** (`docker rmi -f` forzamos la eliminación de la imagen)

docker kill. Mata a uno o más contenedores en ejecución

docker rm. Retire uno o más contenedores. Con la letra **-f** (`docker rm -f` forzamos la eliminación de un contenedor en ejecución (usa SIGKILL))

HERRAMIENTAS DEL PROYECTO

A partir de ahora, todo lo que voy a hacer sobre este proyecto es en la máquina cliente Ubuntu. Lo primero de todo será instalar las dos herramientas de trabajo en sí: **Visual Studio Code** y **Docker**. Posteriormente instalaré

Instalación de Visual Code Studio

Comenzamos actualizando el índice de «apt». `sudo apt update`

```
root@ubuntu: /home/administrator
File Edit View Search Terminal Help
administrator@ubuntu:~$ sudo su
[sudo] password for administrator:
root@ubuntu:/home/administrator# sudo apt update
Hit:1 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:5 http://packages.microsoft.com/repos/vscode stable InRelease
Hit:6 https://packages.microsoft.com/ubuntu/18.04/mssql-server-2019 bionic InRelease
Hit:7 https://packages.microsoft.com/ubuntu/18.04/prod bionic InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
81 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ubuntu:/home/administrator#
```

Continuamos con paquetes necesarios.

`sudo apt install -y curl apt-transport-https`

```
root@ubuntu:/home/administrator# sudo apt install -y curl apt-transport-https
E: Could not get lock /var/lib/dpkg/lock-frontend - open (11: Resource temporarily unavailable)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontend), is another process using it?
root@ubuntu:/home/administrator# sudo fuser -vkl /var/lib/dpkg/lock-frontend
USER          PID ACCESS COMMAND
/var/lib/dpkg/lock-frontend:
          root          4191 F.... unattended-upgr
Kill process 4191 ? (y/N) y
root@ubuntu:/home/administrator# sudo apt install -y curl apt-transport-https
Reading package lists... Done
Building dependency tree
Reading state information... Done
curl is already the newest version (7.58.0-2ubuntu3.13).
The following packages were automatically installed and are no longer required:
  linux-hwe-5.4-headers-5.4.0-42 linux-hwe-5.4-headers-5.4.0-58
  linux-hwe-5.4-headers-5.4.0-62
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  apt-transport-https
0 upgraded, 1 newly installed, 0 to remove and 79 not upgraded.
Need to get 1,692 B of archives.
After this operation, 154 kB of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 apt-transport-https all 1.6.13 [1,692 B]
Fetched 1,692 B in 0s (4,985 B/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 224085 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_1.6.13_all.deb ...
Unpacking apt-transport-https (1.6.13) ...
Setting up apt-transport-https (1.6.13) ...
```

Descargamos e instalamos la clave «GPG» de Microsoft.

```
curl -sSL https://packages.microsoft.com/keys/microsoft.asc -o microsoft.asc
```

```
sudo apt-key add microsoft.asc
```

```
root@ubuntu:/home/administrator# curl -sSL https://packages.microsoft.com/keys/microsoft.asc -o microsoft.asc
root@ubuntu:/home/administrator# sudo apt-key add microsoft.asc
OK
root@ubuntu:/home/administrator#
```

Agregamos el nuevo repositorio.

```
echo "deb [arch=amd64] https://packages.microsoft.com/repos/vscode stable main" | sudo tee /etc/apt/sources.list.d/vscode.list
```

```
root@ubuntu:/home/administrator# echo "deb [arch=amd64] https://packages.microsoft.com/repos/vscode stable main" | sudo tee /etc/apt/sources.list.d/vscode.list
deb [arch=amd64] https://packages.microsoft.com/repos/vscode stable main
root@ubuntu:/home/administrator#
```

Para la instalación, volvemos a actualizar el índice de apt

```
sudo apt update
```

```
root@ubuntu:/home/administrator# sudo apt update
Hit:1 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:5 https://packages.microsoft.com/ubuntu/18.04/mssql-server-2019 bionic InRelease
Hit:6 https://packages.microsoft.com/ubuntu/18.04/prod bionic InRelease
Hit:7 https://packages.microsoft.com/repos/vscode stable InRelease
Reading package lists... Done
Building dependency tree
Reading state information... Done
79 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ubuntu:/home/administrator#
```

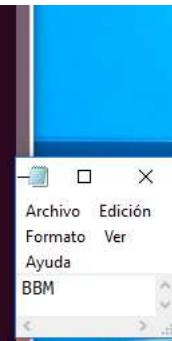
Ahora instalamos con el comando:

```
sudo apt install -y code
```

```
root@ubuntu:/home/administrator# sudo apt install -y code
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-hwe-5.4-headers-5.4.0-42 linux-hwe-5.4-headers-5.4.0-58
  linux-hwe-5.4-headers-5.4.0-62
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  code
0 upgraded, 1 newly installed, 0 to remove and 79 not upgraded.
Need to get 76.4 MB of archives.
After this operation, 291 MB of additional disk space will be used.
Get:1 https://packages.microsoft.com/repos/vscode stable/main amd64 code amd64 1.56.2-1620838498 [76.4 MB]
20% [1 code 19.4 MB/76.4 MB 25%] 784 kB/s 1min 12s
```

Dejamos cargando

```
1.56.2-1620838498 [76.4 MB]
Fetched 76.4 MB in 1min 40s (763 kB/s)
Selecting previously unselected package code.
(Reading database ... 224089 files and directories currently installed.)
Preparing to unpack .../code_1.56.2-1620838498_amd64.deb ...
Unpacking code (1.56.2-1620838498) ...
Setting up code (1.56.2-1620838498) ...
Processing triggers for gnome-menus (3.13.3-11ubuntu1.1) ...
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for desktop-file-utils (0.23-1ubuntu3.18.04.2) ...
Processing triggers for shared-mime-info (1.9-2) ...
root@ubuntu:/home/administrador#
```



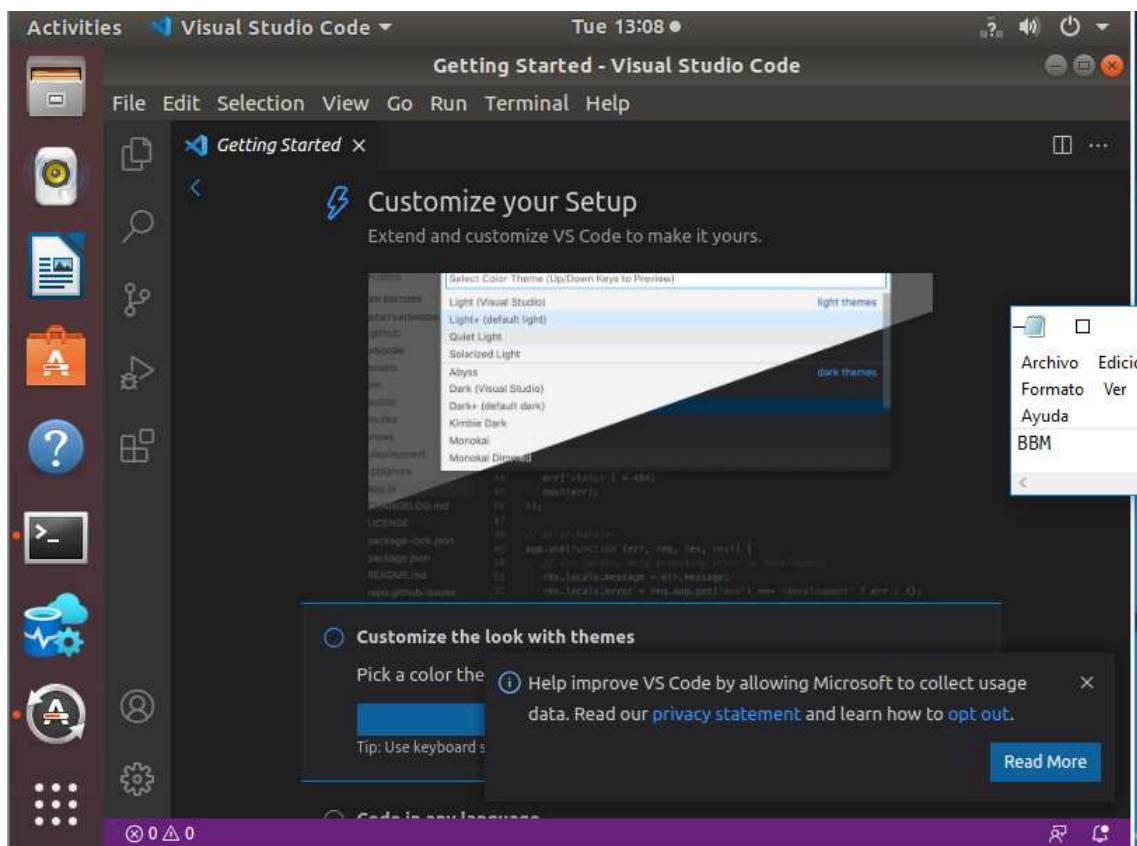
...

```
root@ubuntu:/home/administrador# code
You are trying to start Visual Studio Code as a super user which isn't recommended. If this was intended, please specify an alternate user data directory using the `--user-data-dir` argument.
root@ubuntu:/home/administrador#
```

Press Ctrl+G.



Ahora que está instalado, se puede iniciar desde el menú clásico de tu Linux, o directamente desde la consola con el comando `code`



Instalación de Docker Engine

Para instalar **Docker Engine**, se necesita la versión de 64 bits de una de estas versiones de Ubuntu:

- Ubuntu Hirsute 21.04
- Ubuntu Groovy 20.10
- Ubuntu Focal 20.04 (LTS)
- Ubuntu Bionic 18.04 (LTS)
- Ubuntu Xenial 16.04 (LTS)

Docker Engine es compatible con **x86_64** (o **amd64**) **armhf** y **arm64** arquitecturas.

Primero ejecutamos **sudo apt-get remove docker docker-engine docker.io containerd runc** para desinstalar versiones antiguas

```
root@ubuntu:/home/administrador# sudo apt-get remove docker docker-engine docker.io containerd runc
Reading package lists... Done
Building dependency tree
Reading state information... Done
Package 'docker-engine' is not installed, so not removed
Package 'docker' is not installed, so not removed
Package 'containerd' is not installed, so not removed
Package 'docker.io' is not installed, so not removed
Package 'runc' is not installed, so not removed
The following packages were automatically installed and are no longer required:
  linux-headers-5.4.0-65-generic linux-hwe-5.4-headers-5.4.0-42
  linux-hwe-5.4-headers-5.4.0-58 linux-hwe-5.4-headers-5.4.0-62
  linux-hwe-5.4-headers-5.4.0-65 linux-image-5.4.0-65-generic
  linux-modules-5.4.0-65-generic linux-modules-extra-5.4.0-65-generic
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 50 not upgraded.
root@ubuntu:/home/administrador#
```

El **lsb_release -a** es un subcomando que devuelve el nombre de la distribución de Ubuntu. (también sirve **lsb reléase -cs**)

```
root@ubuntu:/home/administrador# lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.5 LTS
Release:        18.04
Codename:       bionic
```

```
root@ubuntu:/home/administrador# lsb_release -cs
bionic
```

Actualizmos el aptíndice de paquetes e instalamos paquetes para permitir el uso de un repositorio sobre HTTPS:

```
sudo apt-get update
```

```
root@ubuntu:/home/administrador# sudo apt-get update
Hit:1 https://download.docker.com/linux/ubuntu bionic InRelease
Hit:2 http://security.ubuntu.com/ubuntu bionic-security InRelease
Hit:3 https://packages.microsoft.com/ubuntu/18.04/mssql-server-2019 bionic InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu bionic InRelease
Hit:5 https://packages.microsoft.com/ubuntu/18.04/prod bionic InRelease
Get:6 https://packages.microsoft.com/repos/vscode stable InRelease [3,959 B]
Hit:7 http://us.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:8 http://us.archive.ubuntu.com/ubuntu bionic-backports InRelease
Get:9 https://packages.microsoft.com/repos/vscode stable/main amd64 Packages [242 kB]
Fetched 246 kB in 1s (255 kB/s)
Reading package lists... Done
```

```
sudo apt-get install \
apt-transport-https \
ca-certificates \
curl \
gnupg \
lsb-release
```

```
root@ubuntu:/home/administrator# sudo apt-get install \
> apt-transport-https \
> ca-certificates \
> curl \
> gnupg \
> lsb-release
Reading package lists... Done
Building dependency tree
Reading state information... Done
lsb-release is already the newest version (9.20170808ubuntu1).
ca-certificates is already the newest version (20210119~18.04.1).
curl is already the newest version (7.58.0-2ubuntu3.13).
gnupg is already the newest version (2.2.4-1ubuntu1.4).
apt-transport-https is already the newest version (1.6.13). BBM
The following packages were automatically installed and are no longer required:
  linux-headers-5.4.0-65-generic linux-hwe-5.4-headers-5.4.0-42
  linux-hwe-5.4-headers-5.4.0-58 linux-hwe-5.4-headers-5.4.0-62
  linux-hwe-5.4-headers-5.4.0-65 linux-image-5.4.0-65-generic
  linux-modules-5.4.0-65-generic linux-modules-extra-5.4.0-65-generic
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 50 not upgraded.
```

Agregamos la clave GPG oficial de Docker:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo
gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 50 not upgraded.
root@ubuntu:/home/administrator# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
File '/usr/share/keyrings/docker-archive-keyring.gpg' exists. Overwrite? (y/N) Y BBM
```

Utilizamos el siguiente comando para configurar el repositorio **stable**:

```
echo \
"deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
$(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

root@ubuntu:/home/administrator# echo \
> "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \
> $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null BBM
```

PRIMEROS PASOS CON DOCKER:

Comenzar con el contenedor Hellow-World

Para este paso, tan solo hemos de escribir el comando `docker run hello-world`:

```
administrator@ubuntu:~$ sudo docker run hello-world
[sudo] password for administrator: BBM

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
1. The Docker client contacted the Docker daemon.
2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
   (amd64)
3. The Docker daemon created a new container from that image which runs the
   executable that produces the output you are currently reading.
4. The Docker daemon streamed that output to the Docker client, which sent it
   to your terminal.

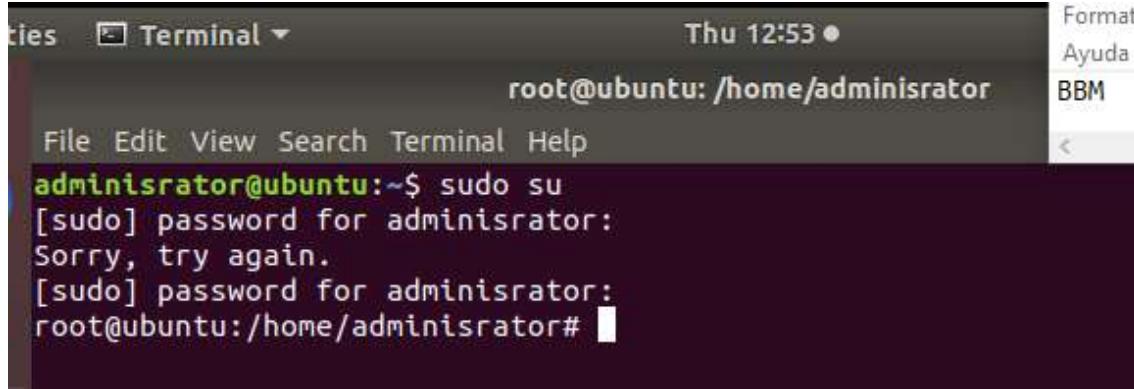
To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

Uso de los comandos básicos de Docker

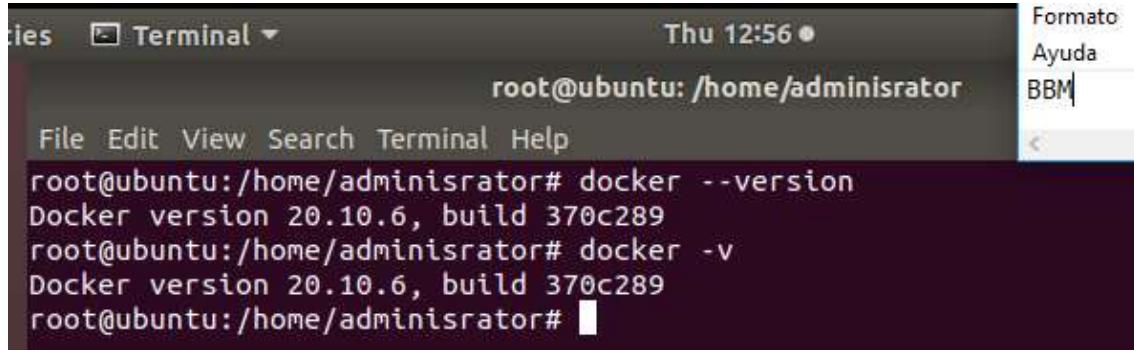
Para trabajar esta parte, nos ponemos en el modo admin



The screenshot shows a terminal window with a dark theme. The title bar says "Terminal". The status bar at the top right shows the date and time as "Thu 12:53". The command entered was `sudo su`, followed by the password prompt "[sudo] password for administrator:", and the error message "Sorry, try again.". The user is still at the root prompt, indicated by the "#".

```
File Edit View Search Terminal Help
administrator@ubuntu:~$ sudo su
[sudo] password for administrator:
Sorry, try again.
[sudo] password for administrator:
root@ubuntu:/home/administrator#
```

Primero tratamos de ver la versión de Docker: `docker --versión`, o bien, `docker -v`



The screenshot shows a terminal window with a dark theme. The title bar says "Terminal". The status bar at the top right shows the date and time as "Thu 12:56". The user is at the root prompt "#". They run two commands: `docker --version` and `docker -v`. Both commands return the same output: "Docker version 20.10.6, build 370c289".

```
File Edit View Search Terminal Help
root@ubuntu:/home/administrator# docker --version
Docker version 20.10.6, build 370c289
root@ubuntu:/home/administrator# docker -v
Docker version 20.10.6, build 370c289
root@ubuntu:/home/administrator#
```

Vamos a darle permisos a la cuenta actual sobre servicios de docker

```
sudo usermod -aG docker ${USER}
su - ${USER}
```

```
root@ubuntu:/home/administrator# sudo usermod -aG docker ${USER}
root@ubuntu:/home/administrator# su - ${USER}
```

Comprobación para ver que hizo bien la suscripción:

```
root@ubuntu:~# su - ${USER}
No password entry for user '$'
root@ubuntu:/home/administrator# su - ${USER}
root@ubuntu:~# id -nG
root docker
root@ubuntu:~#
```

`docker search <imagen>` es el comando para buscar imágenes en Docker Hub

NAME	STARS	OFFICIAL	AUTOMATED	DESCRIPTION
ubuntu	12264	[OK]		Ubuntu is a Debian-based Linux operating system.
sed/Linux-operating-system	12264	[OK]		Docker image to provide a simple way to run a Linux operating system.
dorowu/ubuntu-desktop-lxde-vnc	532		[OK]	Ubuntu desktop environment with LXDE and VNC.
de/HTML5-VNC-interface	532			WebSphere Liberty runtime environment.
websphere-liberty				WebSphere Liberty runtime environment.
ti/architecture/images	271	[OK]		Dockerized SSH server.
rastasheep/ubuntu-sshd	251		[OK]	Ubuntu container with SSH enabled.
e, built on top of official	251			Ubuntu container with SSH enabled.
consol/ubuntu-xfce-vnc				"headless" VNC session.
"headless" VNC session	240		[OK]	Upstart is an event-based replacement for the traditional init system.
ubuntu-upstart				Upstart is an event-based replacement for the traditional init system.
ased/replacement-for-the-traditional-init	110	[OK]		Ubuntu 16.04 LTS Docker image.
1and1/internet/ubuntu-16.04-nginx-php5.6				Ubuntu 16.04 LTS Docker image with Nginx and PHP 5.6.
phpmyadmin/mysql-5.6	50		[OK]	Open Liberty multi-tier application server.
open-liberty				Debootstrap --variant=minbase --components=multiarch.
chitecture/images/based-on-debian	45	[OK]		Ubuntu is a Debian-based Linux operating system.
ubuntu-debootstrap				Ubuntu docker images with minimal components.
=minbase--components=multiarch	44	[OK]		Ubuntu is a Debian-based Linux operating system.
i386/ubuntu				Simple always up-to-date Ubuntu image.
sed/Linux-operating-system	25			Ubuntu 16.04 LTS Docker image.
nuagebec/ubuntu				Ubuntu 16.04 LTS Docker image with Apache, PHP 7.0, MySQL 5.6, and PHPMyAdmin.
Ubuntu-docker-images-with-minimal-components	24		[OK]	Ubuntu 16.04 LTS Docker image with minimal components.
1and1/internet/ubuntu-16.04-apache-php-5.6				Ubuntu 16.04 LTS Docker image with Apache, PHP 5.6, MySQL 5.6, and PHPMyAdmin.
5.6	14		[OK]	Ubuntu 16.04 LTS Docker image with Apache, PHP 7.0, MySQL 5.6, and PHPMyAdmin.
1and1/internet/ubuntu-16.04-apache-php-7.0				Ubuntu 16.04 LTS Docker image with Apache, PHP 7.0, MySQL 5.6, and PHPMyAdmin.

Docker pull <nombre_imagen> para descargar (tirar de una imagen). Si quisiera subirla sería docker pull <nombre_imagen>. Si yo no le pongo nada, tira de la latest que tiene Docker Hub como la oficial

```
root@ubuntu:~# docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
345e3491a907: Pull complete
57671312ef6f: Pull complete
5e9250ddb7d0: Pull complete
Digest: sha256:cf31af331f38d1d7158470e095b132acd126a7180a54f263d386da88eb681c
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
root@ubuntu:~#
```

NOTA: El sha256 es el nombre de la imagen *hasheada*

Con docker images puedo saber las imágenes que tengo y en qué estado están

```
root@ubuntu:~# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu latest 7e0aa2d69a15 3 weeks ago 72.7MB
root@ubuntu:~#
```

Yo la imagen que quiero borrar no la tengo, pero el comando que uso es docker rmi <imagen>

```
root@ubuntu:~# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu latest 7e0aa2d69a15 3 weeks ago 72.7MB
root@ubuntu:~# docker rmi hello/world
Error: No such image: hello/world
root@ubuntu:~# docker rmi hello-world
Error: No such image: hello-world
root@ubuntu:~#
```

NOTA: Si quiero borrar la imagen puedo hacer docker stop o el comando docker rmi -f <imagen> (force)

```
root@ubuntu:/home/administrator# docker rmi hello-world -f
Untagged: hello-world:latest
Untagged: hello-world@sha256:5122f6204b6a3596e048758cabba3c46b1c937a46b5be6225b
835d091b90e46c
Deleted: sha256:d1165f2212346b2bab48cb01c1e39ee8ad1be46b87873d9ca7a4e434980a772
6
```

El comando `docker ps -a` me da los contenedores todos

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
		PORTS NAMES		
d788c1e63324	d1165f221234	"/hello"	25 minutes ago	Exited
(0) 21 minutes ago		admiring_jemison		
9f6990617236	mysql	"docker-entrypoint.s..."	8 days ago	Exited
(0) 8 days ago		some-mysql		
e9df73746d97	d1165f221234	"/hello"	8 days ago	Exited
(0) 8 days ago		wizardly_wozniak		

NOTA: los que pone *Exited* son los parados.

Este comando me da solo los arrancados (en este caso ninguno)

```
root@ubuntu:~# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
root@ubuntu:~#
```

Lanzar un contenedor de MySQL

Ejecturamos el comando `docker run -d --rm --name bbmmysql -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 mysql`, donde:

```
root@ubuntu:/home/administrator# docker run -d --rm --name bbmmysql -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 mysql
293f15f81b06d82d5348db5860db8cc7862102f9471715f065ca4f87f09be898
```

- `-d`: detach
- `-rm`: borrar al cerrar sesión
- `--name`: nombre
- `-e`: enter
- `-p`: puertos mapeados
- `host`: container

Vemos nuestros repositorios:

```
root@ubuntu:/home/administrator# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
mysql latest c0cdc95609f1 13 days ago 556MB
ubuntu latest 7e0aa2d69a15 4 weeks ago 72.7MB
hello-world latest d1165f221234 2 months ago 13.3kB
```

Vemos nuestros contenedores *up and running*:

```
ubuntu latest 7e0aa2d69a15 4 weeks ago 72.7MB
hello-world latest d1165f221234 2 months ago 13.3kB
root@ubuntu:/home/administrator# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS
PORTS NAMES
293f15f81b06 mysql "docker-entrypoint.s..." 2 minutes ago Up 2 minutes
0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp bbmmysql
root@ubuntu:/home/administrator#
```

Para ver los logs usamos `docker logs` (muy útiles a la hora de hacer *troubleshooting*):

```
Fetch the logs of a container
root@ubuntu:/home/administrator# docker ps
CONTAINER ID   IMAGE      COMMAND             CREATED          STATUS           NAMES
              PORTS
293f15f81b06   mysql      "docker-entrypoint.s..."   6 minutes ago   Up 6 minutes   bbmmysql
root@ubuntu:/home/administrator# docker logs 293f15f81b06
2021-05-25 19:41:29+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.25-1debian10 started.
2021-05-25 19:41:31+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2021-05-25 19:41:31+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.25-1debian10 started.
2021-05-25 19:41:32+00:00 [Note] [Entrypoint]: Initializing database files
2021-05-25T19:41:32.065131Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.25) initializing of server in progress as process 41
2021-05-25T19:41:32.074336Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-05-25T19:41:35.146209Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-05-25T19:41:42.358452Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty password ! Please consider switching off the --initialize-insecure option.
2021-05-25 19:41:51+00:00 [Note] [Entrypoint]: Database files initialized
2021-05-25 19:41:51+00:00 [Note] [Entrypoint]: Starting temporary server
2021-05-25T19:41:53.426509Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.25) starting as process 86
```

...

```
hutdown complete (mysqld 8.0.25) MySQL Community Server - GPL.
2021-05-25 19:42:15+00:00 [Note] [Entrypoint]: Temporary server stop
2021-05-25 19:42:15+00:00 [Note] [Entrypoint]: MySQL init process done. Ready for start up.

2021-05-25T19:42:15.953841Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.25) starting as process 1
2021-05-25T19:42:15.961710Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-05-25T19:42:16.705023Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-05-25T19:42:16.832504Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysqlx.sock
2021-05-25T19:42:16.905636Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2021-05-25T19:42:16.905882Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are now supported for this channel.
2021-05-25T19:42:16.907747Z 0 [Warning] [MY-011810] [Server] Insecure configuration for --pid-file: Location '/var/run/mysqld' in the path is accessible to all OS users. Consider choosing a different directory.
2021-05-25T19:42:16.922933Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Version: '8.0.25' socket: '/var/run/mysqld/mysqld.sock' port: 3306 MySQL Community Server - GPL.
```

Con `docker inspect` vemos el contenido:

```
port: 3306  MySQL Community Server - GPL.
root@ubuntu:/home/administrador# docker inspect mysql
[
    {
        "Id": "sha256:c0cdc95609f1fc1daf2c7cae05ebd6adcf7b5c614b4f424949554a240
12e3c09",
        "RepoTags": [
            "mysql:latest"
        ],
        "RepoDigests": [
            "mysql@sha256:d50098d7fcb25b1fcb24e2d3247cae3fc55815d64fec640dc3958
40f8fa80969"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2021-05-12T08:10:33.02552788Z",
        "Container": "d52eeb7ba24f013ff83a9438018e7a3f4f0f19fec2a02c8908810f7be
f7285aa",
        "ContainerConfig": {
            "Hostname": "d52eeb7ba24f",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "ExposedPorts": {
                "3306/tcp": {},
                "33060/tcp": {}
            }
        }
    ...
    "sha256:e82f328cb5e68d3c0fcc6604b3c09a2898d94fde76f",
    "sha256:b2abc2ad4a418fb408384c726f800fa1f722ccb3898",
    "sha256:570df12e998cd93e68915a6de7002f9ca1e4b21bbac5scu1z4v777",
    "sha256:ae477702a51387de5407cbaad4e225c3b3ddfb329cb1b00739b4161
    "sha256:3182d4b853f01f95a1cc30cd97c3e5e4d1aa3011ba4cf6273c2d5b3
    "sha256:cffd1f984514c0d9cfcd194af4fef20c4012e40d00bfe75c669d5b
    "sha256:ea5fd90d1e58fbcd9b68a0cd7daccf8426ac7b0d6c941a877de20b4
    "sha256:74634a9cf30b3935c6e55d5181a6db20f8ae0163368f6bb2fbf6bc2
    "sha256:d605c112cfabb4f748db106728e61b5a47ee084381c6f94e6b3fc46
    "sha256:03a007e88ba3d564d357aad39dd30e9979a7dcfb2329359c6fd9a95
33c6442e6"
    ],
    "Metadata": {
        "LastTagTime": "2001-01-01T00:00:00Z"
    }
]

```

Entramos en el contenedor:

```
[root@ubuntu:/home/administrador# docker exec -it bbmmysql /bin/bash
root@293f15f81b06:/# ]
```

Hacemos `ls` para ver que estamos dentro del contenedor:

```
root@293f15f81b06:/# ls
bin  docker-entrypoint-initdb.d  home  media  proc  sbin  tmp
boot  entrypoint.sh            lib    mnt    root  srv  usr
dev   etc                      lib64  opt    run   sys  var
root@293f15f81b06:/# ]
```

Nos conectamos a MySQL. Contraseña es **Abcd1234**.

```
root@293f15f81b06:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.25 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> ]
```

Hacemos un par de sentencias para ver el funcionamiento:

```
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.01 sec)

mysql> exit
Bye
root@293f15f81b06:/# ]
```

Me salgo del contenedor con `exit`:

```
root@293f15f81b06:/# exit
exit
root@ubuntu:/home/administrador# ]
```

Detenemos el contenedor:

```
root@ubuntu:/home/administrador# docker stop 293f15f81b06
293f15f81b06
```

NOTA: como en la sentencia de instalación del MySQL había un **-rm**, se borra al hacer **stop**. Comprobamos y no necesito borrarlo a mano

```
root@ubuntu:/home/administrador# docker stop 293f15f81b06
293f15f81b06
root@ubuntu:/home/administrador# docker ps
CONTAINER ID   IMAGE      COMMAND   CREATED     STATUS      PORTS     NAMES
root@ubuntu:/home/administrador#
```

Ahora pruebo a crear un volumen. El propio sistema asocia el volumen en una subcarpeta de Docker, con una carpeta de Docker

```
CONTAINER ID   IMAGE      COMMAND   CREATED     STATUS      PORTS     NAMES
root@ubuntu:/home/administrador# docker volume create bbm_mysql_data
bbm_mysql_data
```

NOTA: con el comando **COMMAND**, veo las opciones que tengo

```
root@ubuntu:/home/administrador# docker volume COMMAND
Usage: docker volume COMMAND
Manage volumes
Commands:
  create      Create a volume
  inspect     Display detailed information on one or more volumes
  ls          List volumes
  prune       Remove all unused local volumes
  rm          Remove one or more volumes
Run 'docker volume COMMAND --help' for more information on a command.
root@ubuntu:/home/administrador#
```

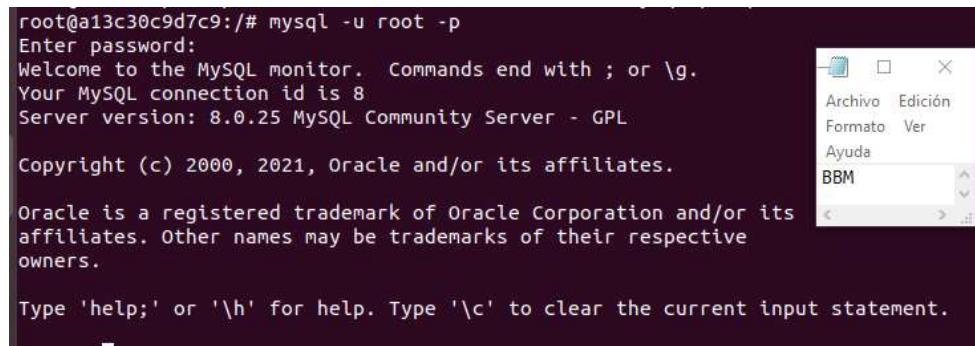
Con **docker volume ls** vemos los volúmenes creados

```
root@ubuntu:/home/administrador# docker volume ls
DRIVER    VOLUME NAME
local     93b4907d9beb269258fe5540dacff65f0c165a7bc370c22392489c4627998d01
local     bbm_mysql_data
root@ubuntu:/home/administrador#
```

Asociamos el volumen al Docker MySQL original

```
root@ubuntu:/home/administrador# docker run -d \
> --name bbmmysql \
> -e MYSQL_ROOT_PASSWORD=Abcd1234. \
> -p 3306:3306 \
> -v bbm_mysql_data:/var/lib/mysql \
> mysql
a13c30c9d7c907eec95b6675e8a3f58c56ff69088352d2f3785968cedb7d36be
root@ubuntu:/home/administrador# docker ps
CONTAINER ID   IMAGE      COMMAND      CREATED     STATUS      PORTS     NAMES
          a13c30c9d7c9   mysql      "docker-entrypoint.s..."   14 seconds ago   Up 13 seconds
          0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   bbmmysql
root@ubuntu:/home/administrador#
```

Entramos en MySQL



```
root@a13c30c9d7c9:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.25 MySQL Community Server - GPL

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

Insertamos este script:

```
DROP DATABASE IF EXISTS BBM_ASPACE;
CREATE DATABASE BBM_ASPACE CHARSET utf8mb4;
USE BBM_ASPACE;

CREATE TABLE BBM_Usuario (
    Usuario_ID INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    Nombre_1 VARCHAR(100) NOT NULL,
    Nombre_2 VARCHAR(100) NOT NULL,
    Apellido_1 VARCHAR(100) NOT NULL,
    Apellido_2 VARCHAR(100) NOT NULL,
    DNI VARCHAR(100) NOT NULL,
    Otros_Detalles VARCHAR(100) NOT NULL
);

INSERT INTO BBM_Usuario VALUES('USERGPP012', 'PACO', 'GONZALEZ',
'PEREZ', '11111012S');
INSERT INTO BBM_Usuario VALUES('USERTEST', 'NOMBRETEST', 'NOMBRE2',
'APELLIDO1', 'APELLIDO2', '123456789', 'LOREM');
```

Hacemos un par de consultas:

```
mysql> show databases
      -> ;
+-----+
| Database |
+-----+
| BBM_ASPACE |
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
5 rows in set (0.01 sec)
```

```
mysql> show tables;
+-----+
| Tables_in_BBMA_ASPACE |
+-----+
| BBM_Usuario |
+-----+
1 row in set (0.00 sec)
```

Hago un par de consultas para comprobar que ha ido bien:

```
mysql> SELECT * FROM BBM_Usuario
      -> ;
+-----+-----+-----+-----+-----+-----+-----+
| Usuario_ID | Nombre_1 | Nombre_2 | Apellido_1 | Apellido_2 | DNI       | Otros_Detalles |
+-----+-----+-----+-----+-----+-----+-----+
|          1 | NOMBRETEST | NOMBRE2 | APELLIDO1 | APELLIDO2 | 123456789 | LOREM      |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Hago una detención.

Detengo el contenedor y como lleva la sentencia `--rm` en el arranque, se borra automáticamente.

```
root@ubuntu:/home/administrator# docker stop bbmysql
bbmysql
root@ubuntu:/home/administrator# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
d788c1e63324 d1165f221234 "/hello" 2 hours ago Exited (0) 2 hours ago
9f6990617236 mysql "docker-entrypoint.s..." 8 days ago Exited (0) 8 minutes ago
e9df73746d97 d1165f221234 "/hello" 8 days ago Exited (0) 8 days ago
wizardly_wozniak
```

MySQL con/sin presencia de datos

Los datos generados durante la operación del contenedor no se escribirán en la imagen. Reiniciar un nuevo contenedor con esta imagen inicializará la imagen y agregará una nueva capa de lectura y escritura para guardar los datos.

Para lograr la persistencia de los datos, la ventana acopiable debe proporcionar volúmenes de datos o volúmenes de contenedores de datos para resolver el problema.

Además, también puede enviar una nueva imagen a través del compromiso para guardar los datos de producción.

Ejemplo creación de un contendor MySQL sin persistencia de datos

Un contenedor Docker que no tiene persistencia de datos quiere decir que cuando finalice la ejecución perderá

todo el contenido que hayamos creado durante la ejecución. Es decir, si durante la ejecución del contenedor

hemos creado varias bases de datos en MySQL Server, éstas se perderán cuando el contenedor se detenga.

El comando que a usar para lanzar el contenedor Docker con MySQL Server sin persistencia de datos podría ser el siguiente: `docker run -d --rm --name BBM -e MYSQL_ROOT_PASSWORD=root -p 3306:3306 mysql:5.7.22`

- `docker run` es el comando que nos permite crear un contenedor a partir de una imagen Docker.
- El parámetro `-d` nos permite ejecutar el contenedor en modo detached, es decir, ejecutándose en segundo plano.
- El parámetro `--rm` hace que cuando salgamos del contenedor, éste se elimine y no ocupe espacio en nuestro disco.
- El parámetro `--name` nos permite asignarle un nombre a nuestro contenedor. Si no le asignamos un nombre Docker nos asignará un nombre automáticamente.
- El parámetro `-e` es para pasárselo al contenedor una variable de entorno. En este caso le estamos pasando la variable de entorno `MYSQL_ROOT_PASSWORD` con el valor de la contraseña que tendrá el usuario root para MySQL Server.
- El parámetro `-p` nos permite mapear los puertos entre nuestra máquina local y el contenedor. En este caso, estamos mapeando el puerto 3306 de nuestra máquina local con el puerto 3306 del contenedor.
- `mysql:5.7.22` es el nombre de la imagen y la versión que vamos a utilizar para crear el contenedor. Si no se indica lo contrario buscará las imágenes en el repositorio oficial Docker Hub.

```
root@ubuntu:/etc# docker run -d --rm --name BBM -e MYSQL_ROOT_PASSWORD=root -p 3306:3306 mysql:5.7.22
Unable to find image 'mysql:5.7.22' locally
5.7.22: Pulling from library/mysql
be8881be8156: Pull complete
c3995dabd1d7: Pull complete
9931fdda3586: Pull complete
bb1b6b6eff6a: Pull complete
a65f125fa718: Pull complete
2d9f8dd09be2: Pull complete
37b912cb2afe: Pull complete
79592d21cb7f: Pull complete
00bfe968d82d: Pull complete
79cf546d4770: Pull complete
2b3c2e6bacee: Pull complete
Digest: sha256:aaba540cdd9313645d892f4f20573e8b42b30e5be71c054b7befed2f7da5f85b
Status: Downloaded newer image for mysql:5.7.22
af8b82222103d47ccf2a6b23bf162ef2999c26da956df451f5c21bd45006bdfa
```

Ejemplo creación de un contendor MySQL sin persistencia de datos

Para que los datos del contenedor sean persistentes, hay que crear un volumen donde se indique el directorio de la máquina local. Se va a almacenar el directorio `/var/lib/mysql`, que es el directorio que utiliza MySQL Server para almacenar las bases de datos.

el parámetro `-v` dentro de los comandos de Docker, es el que se utiliza para la creación de volúmenes.

Docker nos ofrece dos posibilidades para implementar persistencia de datos en los contenedores:

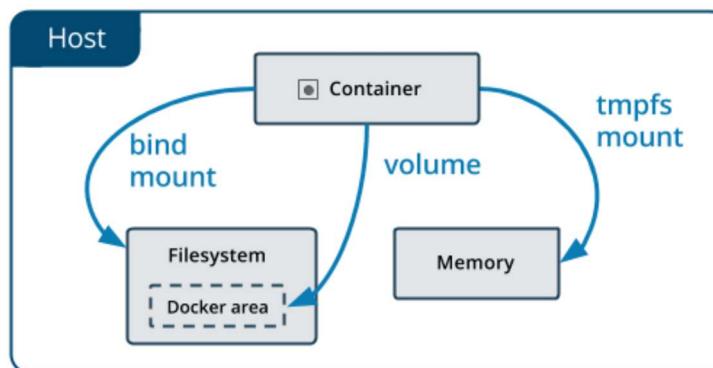
- **bind mount:** pueden estar almacenados en cualquier directorio del sistema de archivos de la máquina host. Estos archivos pueden ser consultados o modificados por otros procesos de la máquina host o incluso por otros contenedores Docker.

Ejemplo de uso del parámetro `-v` para crear un volumen de tipo bind_mount: `-v /home/bbm/data:/var/lib/mysql`

Se puede hacer uso de la variable de entorno `$PWD` para indicar como crear el volumen en nuestro directorio actual : `-v "$PWD":/var/lib/mysql`

En este caso el directorio `/home/bbm/data` de la máquina local estará sincronizado con el directorio `/var/lib/mysql` del contenedor con MySQL Server.

- **volume:** se almacenan en la máquina host dentro del área del sistema de archivos que gestiona Docker. Otros procesos de la máquina host no deberían modificar estos archivos, sólo deberían ser modificados por contenedores Docker.



Ejemplo de uso del parámetro `-v` con un volumen de tipo volume:

```
-v bbmysql_data:/var/lib/mysql
```

El comando que podríamos usar para lanzar nuestro contenedor Docker con MySQL Server con persistencia de datos en un volumen es el siguiente: `docker run -d --rm --name BBM_persistence -e MYSQL_ROOT_PASSWORD=root -p 3306:3306 -v bbmysql_data:/var/lib/mysql mysql:5.7.22`

- `docker run` es el comando que nos permite crear un contenedor a partir de una imagen Docker.
- El parámetro `-d` nos permite ejecutar el contenedor en modo detached, es decir, ejecutándose en segundo plano.
- El parámetro `--rm` hace que cuando salgamos del contenedor, éste se elimine y no ocupe espacio en nuestro disco.
- El parámetro `--name` nos permite asignarle un nombre a nuestro contenedor. Si no le asignamos un nombre Docker nos asignará un nombre automáticamente.
- El parámetro `-e` es para pasarle al contenedor una variable de entorno. En este caso le estamos pasando la variable de entorno `MYSQL_ROOT_PASSWORD` con el valor de la contraseña que tendrá el usuario root para MySQL Server.
- El parámetro `-p` nos permite mapear los puertos entre nuestra máquina local y el contenedor. En este caso, estamos mapeando el puerto 3306 de nuestra máquina local con el puerto 3306 del contenedor.
- El parámetro `-v` nos permite crear un volumen para tener persistencia de datos al finalizar el contenedor.
- `mysql:5.7.22` es el nombre de la imagen y la versión que vamos a utilizar para crear el contenedor. Si no se indica lo contrario buscará las imágenes en el repositorio oficial Docker Hub.

```
root@ubuntu:/etc# docker run -d --rm --name BBM_persistence -e MYSQL_ROOT_PASSWORD=root -p 3306:3306 -v bbmysql_data:/var/lib/mysql mysql:5.7.22
02cfa8489aa9f7ce1fecc4e58e4f0af0d3b63bd0e99e8a58c3f194ea3b3f4c7d
```

Uso del flag `--link` y de user-defined bridge network

Voy a utilizar varios contenedores enlazados de dos formas (por el momento sin utilizar `docker file` ni `docker-compose`)

Voy a utilizar primero MySQL con Adminer (como phpMyAdmin) y luego con phpMyAdmin propiamente dicho.

En los dos casos se puede enlazar los dos contenedores de dos formas:

- Utilizando flag `--link` (deprecated)
- Utilizando un user-defined bridge network. (una especie de VPN)

Utilizando un flag-link.

Primero quiero tener el entorno limpio. Compruebo los contenedores que tengo.

```
root@ubuntu:/home/administrator# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
d788c1e63324        d1165f221234    "/hello"                10 hours ago       Exited (0) 10 hours ago   admiring_jemison
9f6990617236        mysql               "docker-entrypoint.s..."  8 days ago         Exited (0) 9 hours ago   some-mysql
e9df73746d97        d1165f221234    "/hello"                8 days ago         Exited (0) 8 days ago    wizardly_wozniak
```

Uso el comando `docker rm -f $(docker ps -qa)` para forzar los cierres de las máquinas abiertas y con `$ps -qa` los limpio todos. Una vez lo ejecuto, compruebo que está limpio

```
root@ubuntu:/home/administrator# docker rm -f $(docker ps -qa)
d788c1e63324
9f6990617236
e9df73746d97
root@ubuntu:/home/administrator# docker ps -a
CONTAINER ID   IMAGE      COMMAND   CREATED     STATUS      PORTS      NAMES
root@ubuntu:/home/administrator#
```

BBM

Para este ejemplo necesito dos contenedores. Primero el de MySQL, que lo conseguimos como en los ejemplos anteriores:

```
docker run -d \
--name bbmmysqladminer \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=Abcd1234. \
-v mysql_data:/var/lib/mysql \
mysql:5.7.28
```

```
root@ubuntu:/home/administrator# docker run -d \
> --name bbmmysqladminer \
> -p 3306:3306 \
> -e MYSQL_ROOT_PASSWORD=Abcd1234. \
> -v mysql_data:/var/lib/mysql \
> mysql:5.7.28
> Unable to find image 'mysql:5.7.28' locally
> 5.7.28: Pulling from library/mysql
> 804555ee0376: Pull complete
> c53bab458734: Pull complete
> ca9d72777f90: Pull complete
> 2d7aad6cb96e: Pull complete
> 8d6ca35c7908: Pull complete
> 6ddae009e760: Pull complete
> 327ae67bbe7b: Pull complete
> 31f1f8385b27: Pull complete
> a5a3ad97e819: Pull complete
> 48bede7828ac: Pull complete
> 380afa2e6973: Pull complete
> Digest: sha256:b38555e593300df225daea22aeb104eed79fc80d2f064fde1e16e1804d00d0fc
> Status: Downloaded newer image for mysql:5.7.28
> 4756b1da7acf3c5f38027e6edf53b6b42adb951ef8659eff3f83f07b87d43c00
```

BBM

- `-p`: mapea el puerto 3306 del host con el 3306 del contenedor
- `-e`: variables de entorno para pedirme la contraseña
- `-v`: persistencia de datos mapeando el directorio en el host con la ruta
- tiro de la imagen 5.7.28

Una vez que tengo la instancia, voy a crear el contenedor de Adminer.

```
docker run -d \
--rm \
--link bbmmysqladminer \
-p 8080:8080 \
adminer
```

```
root@ubuntu:/home/administrator# docker run -d \
> --rm \
> --link bbmmysqladminer \
> -p 8080:8080 \
> adminer
BBM
Unable to find image 'adminer:latest' locally
latest: Pulling from library/adminer
540db60ca938: Pull complete
933cf2f4a68f: Pull complete
93c5cc202a60: Pull complete
74403c16157d: Pull complete
3ec3ea2589bd: Pull complete
506c26a1a3fd: Pull complete
1884ed429bfb: Pull complete
0426bea1cb5c: Pull complete
07191f3eab51: Pull complete
7a640a6dfcf7: Pull complete
6c2333ef0265: Pull complete
fa2df1129045: Pull complete
39358ec262c5: Pull complete
c3892c2e52ed: Pull complete
6d3841cc0d17: Pull complete
Digest: sha256:ace4f8160eae416242599f63a740317a99936a417fe99e0780aa09f8ea717ff
Status: Downloaded newer image for adminer:latest
208a083ca15509e2405111c796d3bfb16914af1b2d8b440f8082a04383f232eb
```

NOTA: con *detach*, que desaparezca el contenedor cuando se cierre, escuchando en el puerto 8080. La diferencia es el comando *--link*, que lo que hace es enlazarme ambos contenedores.

Hago una comprobación de que me ha creado los dos, y se ve que uno lo crea con el nombre que yo quiero, y al otro le da una especie de nombre simbólico.

CONTAINER ID	IMAGE	COMMAND	CREATED	BBM	STATUS	PORTS	NAMES
208a083ca155	adminer	"entrypoint.sh docke."	About a minute ago		Up 56 seconds	0.0.0.0:8080->8080/tcp, :::8080->8080/tcp	recurring_meninsky
4756b1da7acf	mysql:5.7.28	"docker-entrypoint.s..."	2 minutes ago		Up 2 minutes	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 3306/tcp	bbmmysqladminer

Si vamos a la ruta */etc/hosts*, se debería haber creado una nueva línea gracias a ese comando *--link*. Lo que hace es asignar direcciones privadas para que los dos se vean uno con otro. Lo hacemos a mano

```
root@ubuntu:/etc# more hosts
127.0.0.1      localhost
127.0.1.1      ubuntu
192.168.0.2    WIN-G9HU7TCV86N.bbmaspace.local

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

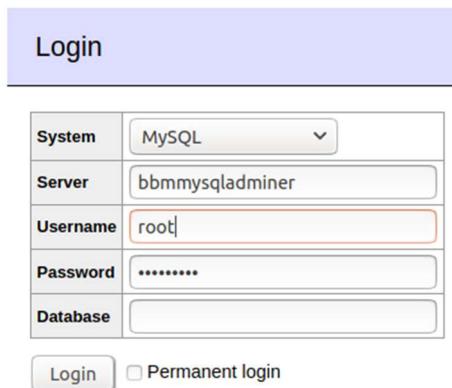
```
root@ubuntu:/etc# more hosts
127.0.0.1      localhost
127.0.1.1      ubuntu
92.168.0.2     WIN-G9HU7TCV86N.bbmaspace.local

# The following lines are desirable for IPv6 capable hosts
::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
72.16.0.3 208a083ca155
72.16.0.2 bbmmysqladminer 4756b1da7acf
```

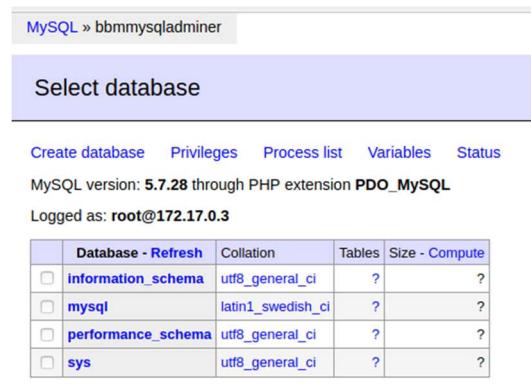
ANTES

DEPUÉS

Hacemos una comprobación: observamos que si introducimos en un navegador web `http://localhost:8080`, el contenedor `adminer` puede conectar con el contenedor `mysql`.



The screenshot shows the Adminer login page. The 'System' dropdown is set to 'MySQL'. The 'Server' field contains 'bbmmysqladminer'. The 'Username' field is filled with 'root'. The 'Password' field contains '*****'. The 'Database' field is empty. Below the form are two buttons: 'Login' and 'Permanent login'.



The screenshot shows the Adminer interface after logging in. The title bar says 'MySQL > bbmmysqladminer'. A 'Select database' button is visible. Below it is a table of databases:

	Database - Refresh	Collation	Tables	Size - Compute
<input type="checkbox"/>	information_schema	utf8_general_ci	?	?
<input type="checkbox"/>	mysql	latin1_swedish_ci	?	?
<input type="checkbox"/>	performance_schema	utf8_general_ci	?	?
<input type="checkbox"/>	sys	utf8_general_ci	?	?

NOTA: el nombre del servidor al que queremos conectarnos no es el de `mysql`, sino que queremos conectarnos a `bbmmysqladminer`, que es el nombre del contenedor de MySQL

Utilizando un user-defined bridge network

Primero eliminamos todo para empezar de cero

```
root@ubuntu:/etc# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
208a083ca155        adminer             "/entrypoint.sh docke..."   36 minutes ago     Up 36 minutes      0.0.0.0:8080->8080/tcp, ::1:8080->8080/tcp   recursing_meninsky
4756b1da7acf        mysql:5.7.28       "/docker-entrypoint.s..."  38 minutes ago     Up 38 minutes      0.0.0.0:3306->3306/tcp, ::1:3306->3306/tcp, 33060/tcp   bbmmysqladminer
root@ubuntu:/etc# docker rm -f $(docker ps -qa)
208a083ca155
4756b1da7acf
root@ubuntu:/etc# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
BBM
```

Creo una red con la sentencia `docker network`

```
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
root@ubuntu:/etc# docker network COMMAND
Usage: docker network COMMAND
Manage networks
Commands:
  connect      Connect a container to a network
  create       Create a network
  disconnect   Disconnect a container from a network
  inspect      Display detailed information on one or more networks
  ls           List networks
  prune        Remove all unused networks
  rm           Remove one or more networks
Run 'docker network COMMAND --help' for more information on a command.
BBM
```

Creamos primero una red interna con el comando `docker network create <nombre>`

```
root@ubuntu:/etc# docker network create bbmnetwork
29ab2ebd3b9f7c3f5147a773579db8c4b46c1f7aeccf58e450608febea37bd52
```

El resto es seguir un poco los pasos anteriores. Primero creo ambos contenedores:

```
docker run -d \
--rm \
--name bbmysqlc \
--network bbmnetwork \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=Abcd1234. \
-v bbmysql_data:/var/lib/mysql \
mysql:5.7.28
```

```
root@ubuntu:/etc# docker run -d \
> --rm \
> --name bbmysqlc \
> --network bbmnetwork \
> -p 3306:3306 \
> -e MYSQL_ROOT_PASSWORD=Abcd1234. \
> -v bbmysql_data:/var/lib/mysql \
> mysql:5.7.28
```

```
docker run -d \
--rm \
--network bbmnetwork \
-p 8080:8080 \
adminer
```

```
root@ubuntu:/etc# docker run -d \
> --rm \
> --network bbmnetwork \
> -p 8080:8080 \
> adminer
cdfb0d7a85a1ca2d1f4d297fe82af609b3fb1cc86cfed691ccfad25c13a53a57
```

BBM

NOTA: Ambos están asociados por la red

Compruebo la existencia de ambos

```
root@ubuntu:/etc# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
cacf2bfe178c mysql:5.7.28 "docker-entrypoint.s..." 6 seconds ago Up 5 seconds 0.0.0.0:3306->3306/tcp, ::3306/tcp, 33060/tcp bbmysqlc
cdfb0d7a85a1 adminer "entrypoint.sh docke..." 8 minutes ago Up 8 minutes 0.0.0.0:8080->8080/tcp, ::8080/tcp xenodochial_proskuriakova
```

Volvemos a lanzar `http://localhost:8080`

Login

System	MySQL
Server	bbmmysqlc
Username	root
Password	*****
Database	

Permanent login

MySQL » bbmyqlc

Select database

Create database Privileges Process list Variables Status

MySQL version: 5.7.28 through PHP extension PDO_MySQL

Logged as: root@172.18.0.2

	Database - Refresh	Collation	Tables	Size - Compute
<input type="checkbox"/>	information_schema	utf8_general_ci	?	?
<input type="checkbox"/>	mysql	latin1_swedish_ci	?	?
<input type="checkbox"/>	performance_schema	utf8_general_ci	?	?
<input type="checkbox"/>	sys	utf8_general_ci	?	?

Selected (0)

Lanzar un contenedor de phpMyAdmin

Una vez explicado estos conceptos con MySQL, voy a intentar el mismo ejemplo con **phpMyAdmin**. Para este ejemplo usaremos la imagen oficial de phpMyAdmin.



Utilizando el flag --link

```
docker run -d \
--rm \
--name bbmmysqlink \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=Abcd1234. \
-v bbmmysql_data:/var/lib/mysql \
mysql:5.7.28

$ docker run -d \
--rm \
--link bbmmysqlink \
-e PMA_ARBITRARY=1 \
-p 8080:80 \
phpmyadmin/phpmyadmin
```

```
root@ubuntu:/etc# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
acf2bfe178c mysql:5.7.28 "docker-entrypoint.s..." 6 seconds ago Up 5 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp bbmyqlc
cdfb0d7a85a1 adminer "entrypoint.sh docker." 8 minutes ago Up 8 minutes 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp xenodochial_proskuriakova

root@ubuntu:/etc# docker rm -f $(docker ps -qa)
acf2bfe178c
cdfb0d7a85a1
root@ubuntu:/etc# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
root@ubuntu:/etc# docker run -d \
> /bin/bash \
> -p 3306:3306 \
> -e MYSQL_ROOT_PASSWORD=Abcd1234. \
> -v bbmmysql_data:/var/lib/mysql \
> mysql:5.7.28
57f5ab3f09416f5fb1b582e0010b73234a1b8dfb28992e0c82569e5cca8f0802f
```

```
root@ubuntu:/etc# docker run -d --rm --link bbmmysqlink -e PMA_ARBITRARY=1 -p 8080:8080 phpmyadmin/phpmyadmin
Unable to find image 'phpmyadmin/phpmyadmin:latest' locally
latest: Pulling from phpmyadmin/phpmyadmin
6f28985ad184: Pull complete
db883aae18bc: Pull complete
ffae70ea03a9: Pull complete                                BBM
1e8027612378: Pull complete
3ec32e53dce5: Pull complete
3bb74037bf77: Pull complete
fedaa0fb085b1: Pull complete
b2244185b327: Pull complete
8852ae668073: Pull complete
985e21deb66e: Pull complete
f262da4e7afa: Pull complete
157f3d683e13: Pull complete
990684a56233: Pull complete
1a17ee268b78: Pull complete
1d4c31287d81: Pull complete
cebcede43b15c: Pull complete
887594e7bbfa: Pull complete
b0f7e76820a4: Pull complete
Digest: sha256:8911fb0cfef21dc9fb385ad02cc3254179cd7df87bab3d3a6fa04d1f0549463f
Status: Downloaded newer image for phpmyadmin/phpmyadmin:latest
f698db1a374c1aid045e208bd06d474767227b4098013ae6052bd66635cdbfb9
```

NOTA: la variable de entorno `-e PMA_ARBITRARY=1` se usa para que en la página principal de phpMyAdmin me aparezca un campo en el formulario donde pueda indicar el servidor al que quiero conectarme.

Utilizando una user-defined bridge network

En primer lugar, creamos una user-defined bridge network con el comando `docker network create <nombre>`. Previamente elimino todo

```
root@ubuntu:/etc# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                 NAMES
f698db1a374c        phpmyadmin/phpmyadmin   "/docker-entrypoint..."   3 minutes ago       Up 3 minutes      80/tcp, 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   xenodochial_lederberg
57f5ab3f0941        mysql:5.7.28          "docker-entrypoint.s..."   8 minutes ago      Up 8 minutes      0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp   bbmmysqlink
f698db1a374c        f698db1a374c1aid045e208bd06d474767227b4098013ae6052bd66635cdbfb9
57f5ab3f0941
root@ubuntu:/etc# docker rm -f $(docker ps -qa)
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS           NAMES
root@ubuntu:/etc# docker network phppnetworkbbm
Usage: docker network COMMAND

Manage networks

Commands:
  connect    Connect a container to a network
  create     Create a network
  disconnect Disconnect a container from a network
  inspect   Display detailed information on one or more networks
  ls         List networks
  prune     Remove all unused networks
  rm         Remove one or more networks

Run 'docker network COMMAND --help' for more information on a command.
root@ubuntu:/etc# docker network create phppnetworkbbm
5b44adb6fb9a94410af3618aed2zf7434df9943fcf73723e6a22230aaa3d177
root@ubuntu:/etc# docker network ls
NETWORK ID        NAME        DRIVER      SCOPE
29ab2ebd3b9f      bbnetwork   bridge      local
c021e6a4a5616      bridge      bridge      local
202967e7f9f9      host        host        local
bc916718c3c7      none       null       local
5b44adb6fb9a      phppnetworkbbm   bridge      local
```

Creamos un contenedor con MySQL indicando que queremos que esté en la red `--network phppnetworkbbm`.

```
$ docker run -d \
--rm \
--name bbmmysqlink \
--network phppnetworkbbm \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=root \
-v bbmysql_data:/var/lib/mysql \
mysql:5.7.28
```

```
root@ubuntu:/etc# docker run -d --rm --name bbmmyslink --network phpnetworkbbm -p 3306:3306 -e MYSQL_ROOT_PASSWORD=root -v bbmysql_data:/var/lib/mysql mysql: 5.7.28
3cd9c454327dfd39aea3246f92bf3bc1e39793e2c0b4f410528cc08dbb2b32f1
```

Creamos un contenedor con phpMyAdmin indicando que queremos que esté en la red -**-network phpnetworkbbm**.

```
$ docker run -d \
--rm \
--network phpnetworkbbm \
-e PMA_ARBITRARY=1 \
-p 8080:80 \
phpmyadmin/phpmyadmin
```

```
root@ubuntu:/etc# docker run -d \
> --rm \
> --network phpnetworkbbm \
> -e PMA_ARBITRARY=1 \
> -p 8080:80 \
> phpmyadmin/phpmyadmin
2cb7e6d13b328a3ae07504bbb7238b5e6140d73822d516e76dc18fe435671fd2
```

Comprobamos que el contenedor phpMyAdmin puede conectar con el contenedor mysql abriendo un navegador web y accediendo a la URL: <http://localhost:8080>.



The screenshot shows the phpMyAdmin configuration interface. At the top, the URL is localhost:8080/bbmmys. The main sections visible are:

- General settings:** Includes options for Change password, Server connection collation (set to utf8mb4_unicode_ci), and More settings.
- Appearance settings:** Includes Language (set to English) and Theme (set to pmahomme).
- Database server:** Shows a console tab.

Below the interface, a terminal window displays the command to remove Docker networks:

```
root@ubuntu:/etc# docker network rm phpnetworkbbm
phpnetworkbbm
root@ubuntu:/etc# docker network rm bbmnetwork
bbmnetwork
```

Lanzar múltiples contenedores: Wordpress + MySQL + phpMyAdmin



Utilizamos las imágenes oficiales que hay en Docker Hub para WordPress, MySQL y phpMyAdmin. A continuación, se detallan los pasos que a seguir:

1. Creamos una user-defined bridge network para todos los contenedores (bbmmultiplecontainer)

```
root@ubuntu:/etc# docker network create bbmmultiplecontainer
83cb58a812d5fd1226050db3cc4fd5a2c00fc6fea67815aa38f6bf593d5f750d
```

root@ubuntu:/etc# docker network ls			
NETWORK ID	NAME	DRIVER	SCOPE
83cb58a812d5	bbmmultiplecontainer	bridge	local
c021e6a45616	bridge	bridge	local
202967e7f9f9	host	host	local
bc916718c3c7	none	null	local

2. Creamos un volumen nuevo para almacenar los datos de MySQL. Por ejemplo, este volumen se puede llamar **bbmstoredata**.

```
root@ubuntu:/etc# docker volume create bbmstoredata  
bbmstoredata
```

MUY IMPORTANTE: Para que las variables de entorno de MySQL tengan efecto, debemos trabajar sobre un volumen que no tenga datos y que no haya sido usado previamente porque si el volumen ya tiene datos, la base de datos y el usuario que le estamos indicando en las variables de entorno no se crearán.

3. Creamos una instancia de un contenedor con MySQL con las siguientes características:

- Se ejecuta en modo detached (background).
- Está en la red **bbmmultiplecontainer**.
- Redirige el puerto 3306 del host al puerto 3306 del contenedor.
- Tiene la variable de entorno **MYSQL_ROOT_PASSWORD** y un valor asignado.
- Tiene la variable de entorno **MYSQL_DATABASE** y un valor asignado.
- Tiene la variable de entorno **MYSQL_USER** y un valor asignado.
- Tiene la variable de entorno **MYSQL_PASSWORD** y un valor asignado.

Usamos el volumen que creamos en el paso 2 (bbmstoredata) para montarlo en el directorio **/var/lib/mysql** del contenedor.

```
$ docker run -d \  
--rm \  
--name bbmmysqlcont \  
--network bbmmultiplecontainer \  
-p 3306:3306 \  
-e MYSQL_ROOT_PASSWORD=root \  
-e MYSQL_DATABASE=BBM_ASPACE \  
-e MYSQL_USER=BBM \  
-e MYSQL_PASSWORD=Abcd1234. \  
-v bbmstoredata:/var/lib/mysql \  
mysql:5.7.28
```

```
root@ubuntu:/etc# docker run -d --rm --name bbmmysqlcont --network bbmultipleco
ntainer -p 3306:3306 -e MYSQL_ROOT_PASSWORD=root -e MYSQL_DATABASE=BBM_ASPACE -
e MYSQL_USER=BBM -e MYSQL_PASSWORD=Abcd1234. -v bbmstoredata:/var/lib/mysql mys
ql:5.7.28
```

```
16cc2ecfbb5f7e1d4684edaf2fdecc4f6e309ce61f2f6891f0712b0c81adf016
```

Creamos una instancia de un contenedor con phpMyAdmin con las siguientes características:

- Se ejecuta en modo detached (background).
- Está en la red bbmmultiplecontainer.
- Redirige el puerto 8080 del host al puerto 80 del contenedor.
- Creamos la variable de entorno PMA_ARBITRARY y le asignamos el valor 1, para que nos permita indicar el nombre del servidor de base de datos al que queremos conectarnos.

```
docker run -d \
--rm \
--network bbmultiplecontainer \
-e PMA_ARBITRARY=1 \
-p 8080:80 \
phpmyadmin/phpmyadmin
```

```
root@ubuntu:/etc# docker run -d --rm --network bbmultiplecontainer -e PMA_ARBIT
RARY=1 -p 8080:80 phpmyadmin/phpmyadmin
66b3c4fdce6c1c1a9e017d831ee681e87adde63de8844f6da4d91e90d713d873
```

4. Creamos una instancia de un contenedor con WordPress con las siguientes características:

- Se ejecuta en modo detached (background).
- Está en la red bbmmultiplecontainer.
- Redirige el puerto 80 del host al puerto 80 del contenedor.
- Creamos la variable de entorno WORDPRESS_DB_HOST y asignamos el nombre del contenedor que está ejecutando MySQL.
- Creamos la variable de entorno WORDPRESS_DB_NAME y asignamos el valor de la base de datos que has creado en la instancia de MySQL.
- Creamos la variable de entorno WORDPRESS_DB_USER y asignamos el valor del usuario de la base de datos que has creado en la instancia de MySQL.
- Creamos la variable de entorno WORDPRESS_DB_PASSWORD y asignamos el valor de la contraseña del usuario de la base de datos que has creado en la instancia de MySQL.

Necesitaremos crear un volumen (bbm_wordpressvolume) para montarlo en el directorio /var/www/html del contenedor.

```
root@ubuntu:/etc# docker volume create bbm_wordpressvolume
bbm_wordpressvolume
```

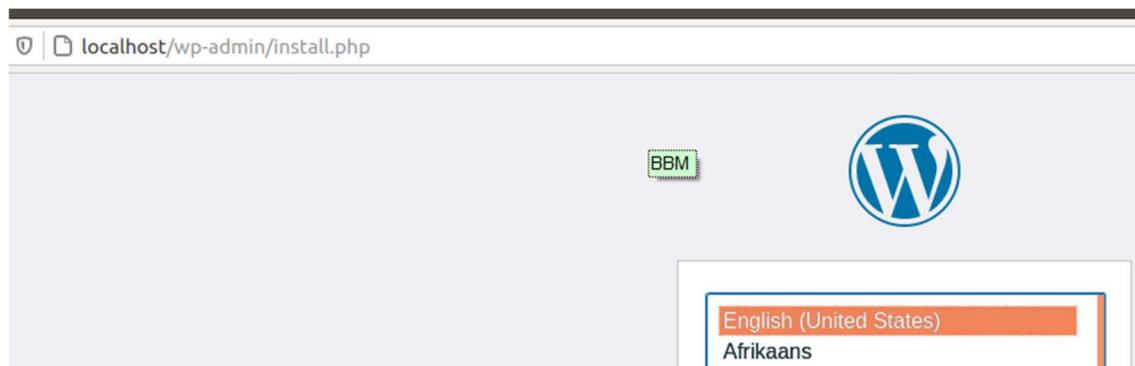
```
docker run -d \
--rm \
--name bbmwordpressc \
--network bbmmultiplecontainer \
-p 80:80 \
-e WORDPRESS_DB_HOST= bbmmysqlcont \
-e WORDPRESS_DB_NAME=BBM_ASPACE \
-e WORDPRESS_DB_USER=BBM \
-e WORDPRESS_DB_PASSWORD=Abcd1234. \
-v bbm_wordpressvolume:/var/www/html \
wordpress
```

```
root@ubuntu:/etc# docker run -d \
> --rm \
> --name bbmwordpressc \
> --network bbmmultiplecontainer \
> -p 80:80 \
> -e WORDPRESS_DB_HOST=bbmmysqlcont \
> -e WORDPRESS_DB_NAME=BBM_ASPACE \
> -e WORDPRESS_DB_USER=BBM \
> -e WORDPRESS_DB_PASSWORD=Abcd1234. \
> -v bbm_wordpressvolume:/var/www/html \
> wordpress
bcea52acde72b0a672e74d659896e624f63d4dc7763eec4302ee86330c97bd8b
```

Resultado con el comando `docker ps -a`

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
bcea52acde72	wordpress	"docker-entrypoint.s..."	43 seconds ago	Up 40 seconds	0.0.0.0:80->80/tcp, :::80->80/tcp	bbmwordpressc
66b3c4fdce0c	phpmyadmin/phpmyadmin	"/docker-entrypoint..."	18 minutes ago	Up 18 minutes	0.0.0.0:8080->80/tcp, :::8080->80/tcp	loving_villani
16cc2ecrbbsf	mysql:5.7.28	"docker-entrypoint.s..."	20 minutes ago	Up 20 minutes	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 3306/tcp	bbmmysqlcont

Para ejecutar localhost:80



Ejecutamos localhost:8080 en el navegador

Language

English

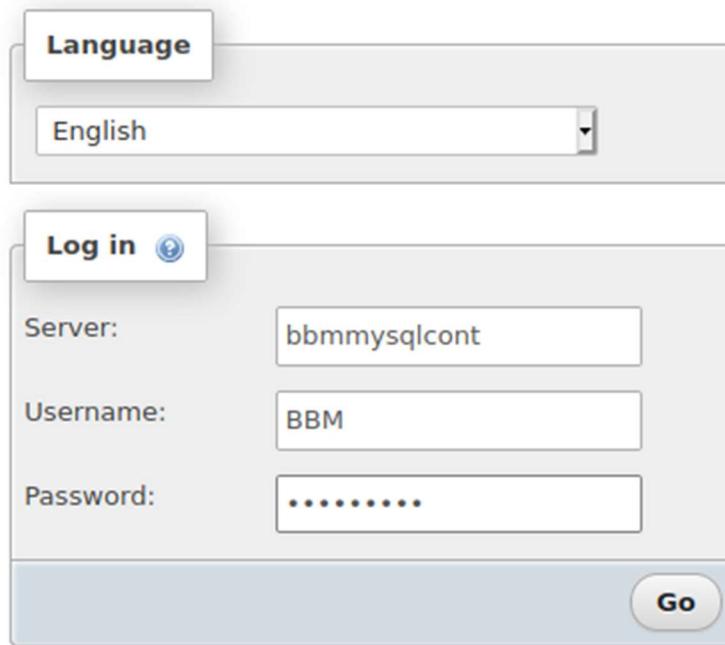
Log in

Server: bbmmysqlcont

Username: BBM

Password: *****

Go



localhost:8080/index.php?route=/&route=%2F
Server: bbmmysqlcont
BBM

Databases SQL Status Export Import Settings VariablesCharsets Engines Plugins

General settings

- Change password
- Server connection collation: utf8mb4_unicode_ci
- More settings

Appearance settings

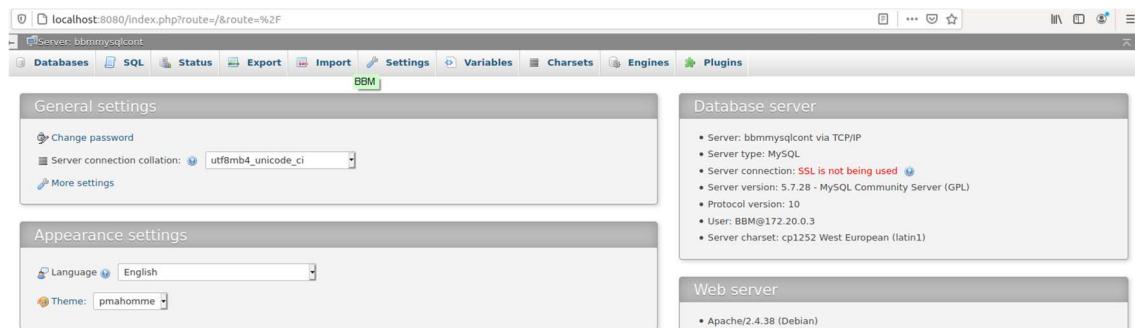
- Language English
- Theme: pmahomme

Database server

- Server: bbmmysqlcont via TCP/IP
- Server type: MySQL
- Server connection: SSL is not being used
- Server version: 5.7.28 - MySQL Community Server (GPL)
- Protocol version: 10
- User: BBM@172.20.0.3
- Server charset: cp1252 West European (latin1)

Web server

- Apache/2.4.38 (Debian)



Creación de un contenedor Docker con MariaDB



Ejecutamos el comando para crear un contenedor de MariaDB: `docker run -p 127.0.0.1:3306:3306 --name bbm-mariadb -e MARIADB_ROOT_PASSWORD=Abcd1234. -d mariadb` para descargar la última versión.

```
root@ubuntu:/etc# docker run -p 127.0.0.1:3306:3306 --name bbm-mariadb -e MARIADB_ROOT_PASSWORD=Abcd1234. -d mariadb
Unable to find image 'mariadb:latest' locally
latest: Pulling from library/mariadb
345e3491a907: Already exists
57671312ef6f: Already exists
5e9250ddb7d0: Already exists
2d512e2ff778: Pull complete
57c1a7dc2af9: Pull complete
b846faf4774a: Pull complete
66409f940bd2: Pull complete
82d8723e99d8: Pull complete
55edbfb0fe73e: Pull complete
c34793730add: Pull complete
8f1925a0d734: Pull complete
72904fb5fd0b: Pull complete
Digest: sha256:0c3c560359a6da112134a52122aa9b78fec5f9dd292a01ee7954de450f25f0c1
Status: Downloaded newer image for mariadb:latest
fe919cfcc36c2b7e6305c8c53cdccf71d2fc4f8877e906bca654d89d31a03
```

Comprobamos el contenedor:

```
root@ubuntu:/etc# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
fe919cfcc36c        mariadb            "docker-entrypoint.s..."   4 minutes ago      Up 3 minutes       127.0.0.1:3306->3306/tcp   bbm-mariadb
```

Comenzamos a trabajar en el contenedor:

```
root@ubuntu:/etc# docker start fe919cfcc36c
fe919cfcc36c
```

Entramos en MariaDB:

```
root@fe919cfcc36c:/# mysql -h 127.0.0.1 -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 3
Server version: 10.5.10-MariaDB-1:10.5.10+maria~focal mariadb.org binary distribution

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)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
+-----+
3 rows in set (0.000 sec)
```

Hacemos unas pruebas para ver el funcionamiento:

```
MariaDB [(none)]> DROP DATABASE IF EXISTS BBM_ASPACE;
Query OK, 0 rows affected, 1 warning (0.000 sec)

MariaDB [(none)]> CREATE DATABASE BBM_ASPACE CHARSET utf8mb4;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> USE BBM_ASPACE;
Database changed
MariaDB [BBM_ASPACE]>
MariaDB [BBM_ASPACE]> CREATE TABLE BBM_Usuario (
    -> Usuario_ID INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    -> Nombre_1 VARCHAR(100) NOT NULL,
    -> Nombre_2 VARCHAR(100) NOT NULL,
    -> Apellido_1 VARCHAR(100) NOT NULL,
    -> Apellido_2 VARCHAR(100) NOT NULL,
    -> DNI VARCHAR(100) NOT NULL,
    -> Otros_Detalles VARCHAR(100) NOT NULL
    -> );
Query OK, 0 rows affected (0.043 sec)

MariaDB [BBM_ASPACE]>
MariaDB [BBM_ASPACE]>
MariaDB [BBM_ASPACE]> INSERT INTO BBM_Usuario VALUES('1', 'NOMBRETEST', 'NOMBRE2', 'APELIDO1', 'APELIDO2', '123456789', 'LOREM');
Query OK, 1 row affected (0.001 sec)

MariaDB [BBM_ASPACE]>
MariaDB [BBM_ASPACE]>
MariaDB [BBM_ASPACE]> show databases;
+-----+
| Database |
+-----+
| BBM_ASPACE |
| information_schema |
| mysql |
| performance_schema |
+-----+
4 rows in set (0.000 sec)

MariaDB [BBM_ASPACE]> select * from BBM_Usuario
-> ;
+-----+-----+-----+-----+-----+-----+-----+
| Usuario_ID | Nombre_1 | Nombre_2 | Apellido_1 | Apellido_2 | DNI | Otros_Detalles |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | NOMBRETEST | NOMBRE2 | APELIDO1 | APELIDO2 | 123456789 | LOREM |
+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.000 sec)
```

Creación de un contenedor con PostgreSQL

Voy a utilizar el ejemplo de instalación de [docker-compose](#). Primero creo un directorio en raíz un directorio:



```
root@ubuntu:~# mkdir postgres_compose
root@ubuntu:~# cd postgres_compose/
root@ubuntu:~/postgres_compose# nano stack.yml
```

En él creamos un archivo .yml

```
root@ubuntu:~/postgre_compose# more bbmpstcompose.yml
# Use postgres/example user/password credentials
version: '3.1'

services:

  db:
    image: postgres
    restart: always
    environment:
      POSTGRES_PASSWORD: Abcd1234.

  adminer:
    image: adminer
    restart: always
    ports:
      - 8080:8080
```

Para poder usar el comando `docker-compose`, tenemos que hacer el comando apt install `docker-compose`:

```
root@ubuntu:~/postgre_compose# apt install docker-compose
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-hwe-5.4.0-42 linux-hwe-5.4.0-58 linux-hwe-5.4.0-62 linux-hwe-5.4.0-65
Use 'sudo apt autoremove' to remove them.

...
Setting up python-requests (2.18.4-2ubuntu0.1) ...
Setting up python-jsonschema (2.6.0-2) ...
update-alternatives: using /usr/bin/python2-jsonschema to provide /usr/bin/jsonschema (jsonschema) in auto mode
Setting up python-openssl (17.5.0-1ubuntu1) ...
Setting up python-docker (2.5.1-1) ...
Setting up docker-compose (1.17.1-2) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
root@ubuntu:~/postgre_compose#
```

Usamos el comando `docker run --name BBMPostgres -e POSTGRES_PASSWORD=Abcd1234. -d postgres` up para obtener la instancia de PostgreSQL:

```
root@ubuntu:~# docker run --name BBMPostgres -e POSTGRES_PASSWORD=Abcd1234. -d postgres
90defeb6827f622c713cc43028771f9359b579732a7d86789c7c4fe3c8ade7a9
root@ubuntu:~# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
90defeb6827f  postgres "docker-entrypoint.s..." 3 seconds ago Up 2 seconds 5432/tcp BBMPostgres
root@ubuntu:~#
```

Arrancamos el contenedor:

```
root@ubuntu:~# docker start 90defeb6827f
90defeb6827f
```

Utilizamos el comando `docker run -it --rm --link some-postgres:postgres postgres psql -h postgres -U postgres` para entrar dentro de PostgreSQL por consola

```
root@ubuntu:~# docker run -it --rm --link some-postgres:postgres postgres psql -h postgres -U postgres
docker: Error response from daemon: could not get container for some-postgres: No such container: some-postgres.
See 'docker run --help'.
root@ubuntu:~# docker run -it --rm --link BBMPostgres:postgres postgres psql -h postgres -U postgres
Password for user postgres:
psql (13.3 (Debian 13.3-1.pgdg100+1))
Type "help" for help.

postgres=# show databases;
ERROR:  unrecognized configuration parameter "databases"
postgres=#

```

Probamos que funciona:

```
postgres=# \l
      List of databases
   Name    |  Owner   | Encoding | Collate | Ctype | Access privileges
-----+-----+-----+-----+-----+-----+
bbm_aspace | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
pedidos    | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
postgres   | postgres | UTF8     | en_US.utf8 | en_US.utf8 |
template0  | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres      +
template1  | postgres | UTF8     | en_US.utf8 | en_US.utf8 | =c/postgres      +
                                         |           |           |           |           | postgres=CTc/postgres
                                         |           |           |           |           | =c/postgres      +
                                         |           |           |           |           | postgres=CTc/postgres
(5 rows)
```

MongoDB



Instalamos **MongoDB Express** siguiendo los pasos de <https://github.com/mongo-express/mongo-express>.

Primero lanzamos el comando `apt install npm`:

```
root@ubuntu:~# apt install npm
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-hwe-5.4-headers-5.4.0-42 linux-hwe-5.4-headers-5.4.0-58 linux-hwe-5.4-headers-5.4.0-62 linux-hwe-5.4-headers-5.4.0-65
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  build-essential dkpg-dev fakeroot g++-7 g++-7-gcc libalgorithm-diff-perl libalgorithm-merge-perl libfakeroot libjs-async libjs-inherits libjs-jquery
  libjs-node-uid libjs-underscore libssl1.0-dev libstdc++-7-dev libuv libuv-dev make node-abbrev node-ansi node-archy node-async node-balanced-match node-block-stream
  node-brace-expansion node-builtln-modules node-combined-stream node-concat-map node-cookie-jar node-delayed-stream node-evergreen-agent node-form-data node-fs.realpath node-fstream node-fstream-ignore
  node-github-auth node-inflight node-graceful-fs node-gyp node-hosted-git-info node-inflight node-inherits node-imd-builtin-module node-isexe node-json-stringify-safe node-lockfile
  node-mime-type node-minimatch node-minimatch node-node-pkg-require node-normalize-package-data node-npmlog node-once node-osenv node-path-is-absolute node-pseudomap node-qsa
  node-read node-read-package-json node-request node-retry node-rimraf node-server node-sha node-slide node-spxd-correct node-spxd-expression-parse node-spxd-license-ids node-tar node-tunnel-agent
  node-underscore node-validate-npm-package-license node-which node-wrapify node-yallist nodejs-dev nodejs-doc python-pkg-resources
Suggested packages:
  debian-keyring g++-multilib gcc-7-doc libstdc++-0-7-dbg apache2 | lighttpd | httpd libstdc++-7-dbg make-doc node-hawk node-aws-sign node-oauth-sign node-http-signature debhelper
  python-pip
The following NEW packages will be installed:
  build-essential dkpg-dev fakeroot g++-7 g++-7-gcc libalgorithm-diff-perl libalgorithm-merge-perl libfakeroot libjs-async libjs-inherits libjs-jquery
  libjs-node-uid libjs-underscore libssl1.0-dev libstdc++-7-dev libuv libuv-dev make node-abbrev node-ansi node-archy node-async node-balanced-match node-block-stream
  node-brace-expansion node-builtln-modules node-combined-stream node-concat-map node-cookie-jar node-delayed-stream node-evergreen-agent node-form-data node-fs.realpath node-fstream node-fstream-ignore
  node-github-auth node-inflight node-graceful-fs node-gyp node-hosted-git-info node-inflight node-inherits node-imd-builtin-module node-isexe node-json-stringify-safe node-lockfile
  node-mime-type node-minimatch node-minimatch node-node-pkg-require node-retry node-rimraf node-server node-sha node-slide node-spxd-correct node-spxd-expression-parse node-spxd-license-ids node-tar node-tunnel-agent
  node-underscore node-validate-npm-package-license node-which node-wrapify node-yallist nodejs-dev nodejs-doc python-pkg-resources
0 upgraded, 88 newly installed, 0 to remove and 53 not upgraded.
After this operation, 94.9 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

...

```
Setting up node-form-data (0.1.0-1) ...
Setting up node-request (2.26.1-1) ...
Setting up node-minimatch (3.0.4-3) ...
Setting up node-normalize-package-data (2.3.5-2) ... BBM
Setting up node-ansi-color-table (1.0.0-1) ...
Setting up node-npmlog (0.0.4-1) ...
Setting up node-glob (7.1.2-4) ...
Setting up node-rimraf (2.6.2-1) ...
Setting up node-read-package-json (1.2.4-1) ...
Setting up node-fstream (1.0.10-1ubuntu0.18.04.1) ...
Setting up node-fstream-ignore (0.0.6-2) ...
Setting up node-tar (2.2.1-1) ...
Setting up node-gyp (3.6.2-1ubuntu1) ...
Setting up npm (3.5.2-0ubuntu4) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for libc-bin (2.27-3ubuntu1.4) ...
root@ubuntu:~#
```

Lanzamos el comando `npm install -g mongo-express`:

```
root@ubuntu:~# npm install -g mongo-express BBM
[!] engine mongo-express@0.0-alpha.2 wanted: {"node":">=10.0.0","npm":">=3.0.0"} (current: {"node":"8.10.0","npm":3.5.2"})
npm WARN deprecated express-fileupload@0.4.0: Please upgrade express-fileupload to version 1.1.8+ due to a security vulnerability with the parseNested option
npm WARN deprecated-dep update-notifier@3.1.0
npm WARN deprecated uuid@2.0.3: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.
loadDep:registry-url -> /usr/local/bin/mongo-express -> /usr/local/lib/node_modules/mongo-express/app.js
```

...

```
deep-extend@0.6.0
  ini@1.3.8
  strip-json-comments@2.0.1
  registry-url@3.1.0
lazy-req@1.1.0
semver-diff@2.1.0
xdg-basedir@2.0.0
  os-homedir@1.0.2
root@ubuntu:~#
```

Creamos el directorio `mongodb` y `bbm_mongodb_compose`, con el comando `mkdir -pv mongodb/bbm_mongodb_compose`:

```
root@ubuntu:~# mkdir -pv mongodb / bbm_mongodb_compose
root@ubuntu:~# ls
- bbmkeytab.keytab  bbm_mongodb_compose  mongodb  postgre_compose  pv  snap
```

Creamos el archivo `.yml` `docker-compose.yml` en el directorio `mongodb`:

```

root@ubuntu:~# cd mongodb/
root@ubuntu:~/mongodb# nano docker-compose.yml
root@ubuntu:~/mongodb# more docker-compose.yml
# Use root/example as user/password credentials
version: '3.1'

services:

mongo:
  image: mongo
  restart: always
  environment:
    MONGO_INITDB_ROOT_USERNAME: root
    MONGO_INITDB_ROOT_PASSWORD: Abcd1234.

mongo-express:
  image: mongo-express
  restart: always
  ports:
    - 8081:8081
  environment:
    ME_CONFIG_MONGODB_ADMINUSERNAME: root
    ME_CONFIG_MONGODB_ADMINPASSWORD: Abcd1234.

```

Usamos el comando `sudo docker-compose up -d` para montar el contenedor:

```

root@ubuntu:~/mongodb# sudo docker-compose up -d
Creating network "mongodb_default" with the default driver
Pulling mongo (mongo:latest)...
latest: Pulling from library/mongo
01bf7da0a88c: Pull complete
f3b4a5f15c7a: Pull complete
57ffbe87baa1: Pull complete
77d5e5c7eab9: Pull complete
43798cf18b45: Pull complete
67349a81f435: Pull complete
590845b1f17c: Pull complete
1f2ff17242ce: Pull complete
6f11b2ce0594: Pull complete
91532386f4ec: Pull complete
705ef0ab262e: Pull complete
e6238126b609: Pull complete
Digest: sha256:8b35c0a75c2dbf23110ed2485fecfa567ec9ab743feee7a0d7a148f806daf5e86
Status: Downloaded newer image for mongo:latest
Pulling mongo-express (mongo-express:latest)...
latest: Pulling from library/mongo-express
ddad3d7c1e96: Pull complete
3a8370f05d5d: Pull complete
71a8563b7fea: Pull complete
119c7e14957d: Pull complete
c06612553eef: Pull complete
931f05f69fde: Pull complete
2766ec5ce375: Pull complete
a60269e588ca: Pull complete
Digest: sha256:1df4d44b722aadb31335105972c62e9c971e015f83e68623cec24e6a1a3f0d38
Status: Downloaded newer image for mongo-express:latest
Creating mongodb_mongo_1 ...
Creating mongodb_mongo-express_1 ...
Creating mongodb_mongo_1
Creating mongodb_mongo-express_1 ... done

```

Verificamos que está:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4c56c36dd243	mongo-express	"tini -- /docker-entrypoint.s..."	About a minute ago	Up 54 seconds	0.0.0.0:8081->8081/tcp, :::8081->8081/tcp	mongodb_mongo-express_1
61fb2277692d	mongo	"docker-entrypoint.s..."	About a minute ago	Up 55 seconds	27017/tcp	mongodb_mongo_1

Ejecutamos el contenedor:

```
root@01fb2277692d:/# mongo
MongoDB shell version v4.4.6
Connecting to: mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("c8a693f0-7a94-4270-a1af-63fbba9497fb") }
MongoDB server version: 4.4.6
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
    https://docs.mongodb.com/
Questions? Try the MongoDB Developer Community Forums
    https://community.mongodb.com
> BBM
```

Comprobamos la red:

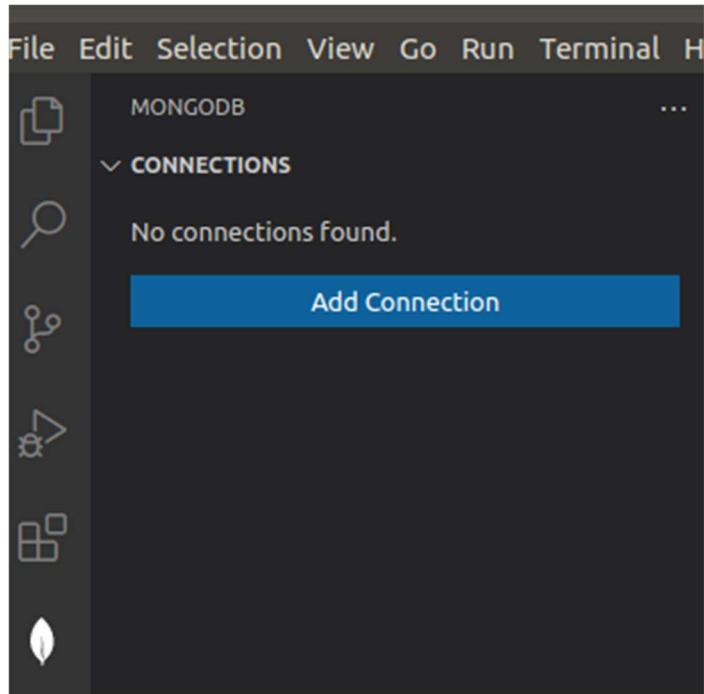
```
oot@ubuntu:~/mongodb# docker network ls
NETWORK ID      NAME          DRIVER      SCOPE
1d34163f8c5    bridge        bridge      local
02967e7f9f9    host          host       local
46d5c388e8b    mongodb_default bridge      local
c916718c3c7    none          null       local
oot@ubuntu:~/mongodb# BBM
```

Vemos que en el navegador lo tenemos:

Databases		Database Name	+ Create Database
View	admin	Del	
View	config	Del	
View	local	Del	

Ahora abrimos el **Visual Studio Code**. Para ello, primero instalamos la extensión seleccionando el ícono y escribiendo en el buscador **MongoDB**:

Al acabar nos aparecerá el ícono . Le damos a Add Connection



New connection

General SSL/TLS SSH Tunnel Advanced

Connection Type

Hostname: localhost Port: 8081

Authentications

None **Username / Password** SCRAM-SHA-256 LDAP

X.509

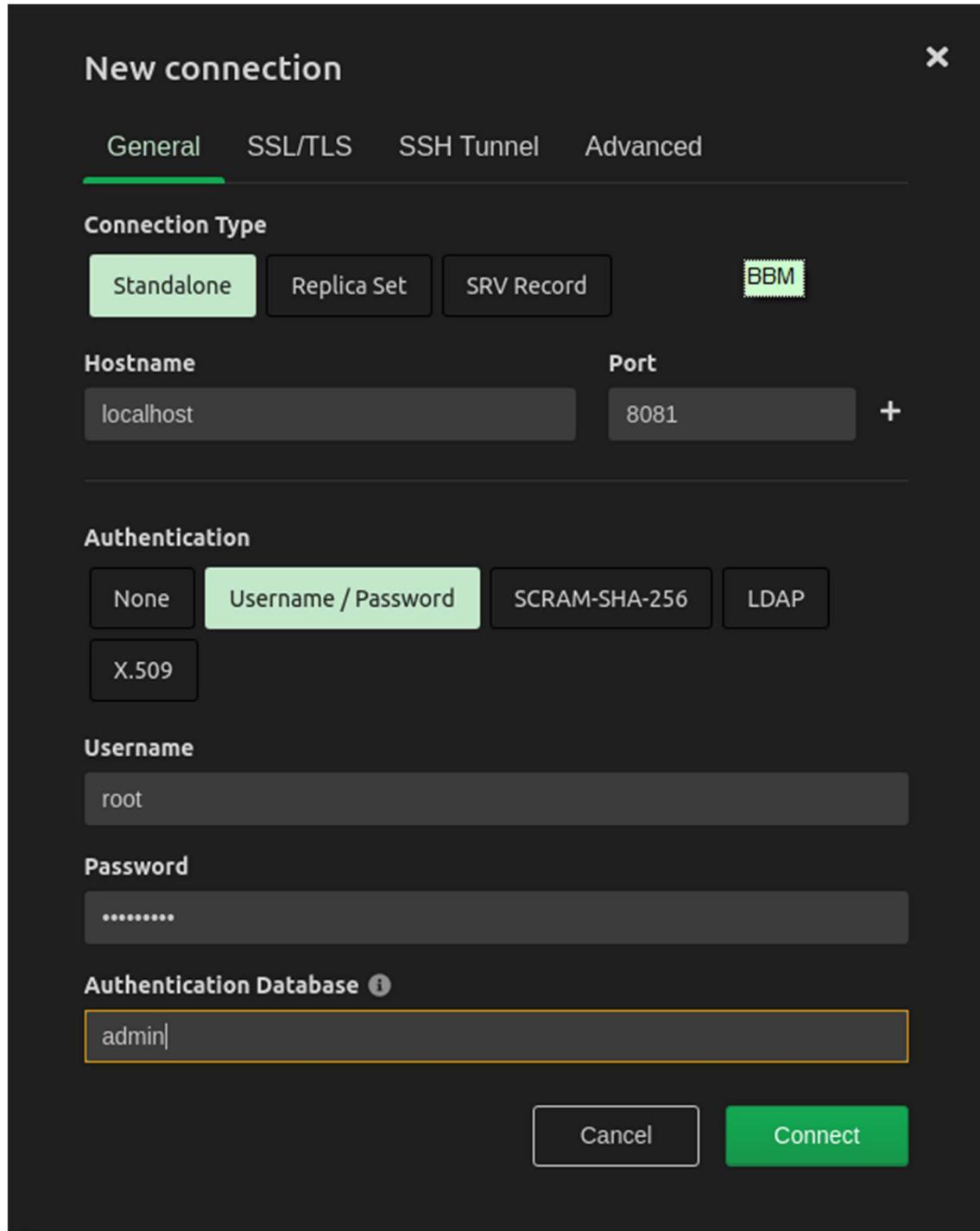
Username: root

Password: *********

Authentication Database i

admin|

Cancel Connect



NOTA: A pesar de introducir bien los parámetros, no conecta con MongoDB

Docker Hub

Es un repositorio en línea basado en la nube que almacena ambos tipos de repositorios, es decir, el repositorio público y el privado. Los repositorios públicos son accesibles para todos, pero el privado es accesible para el propietario interesado de los repositorios; También hay un costo asociado si almacenamos más de un cierto número de repositorios como privado. Para empezar a trabajar con él, primero nos vamos a la página <https://hub.docker.com/> y nos registramos.

Get Started Today for Free

Already have an account? [Sign In](#)

BBM

bbmaspace

clase.bbm@gmail.com

.....

Send me occasional product updates and announcements.

No soy un robot 

[Sign Up](#)

By creating an account, you agree to the [Terms of Service](#), [Privacy Policy](#), and [Data Processing Terms](#).

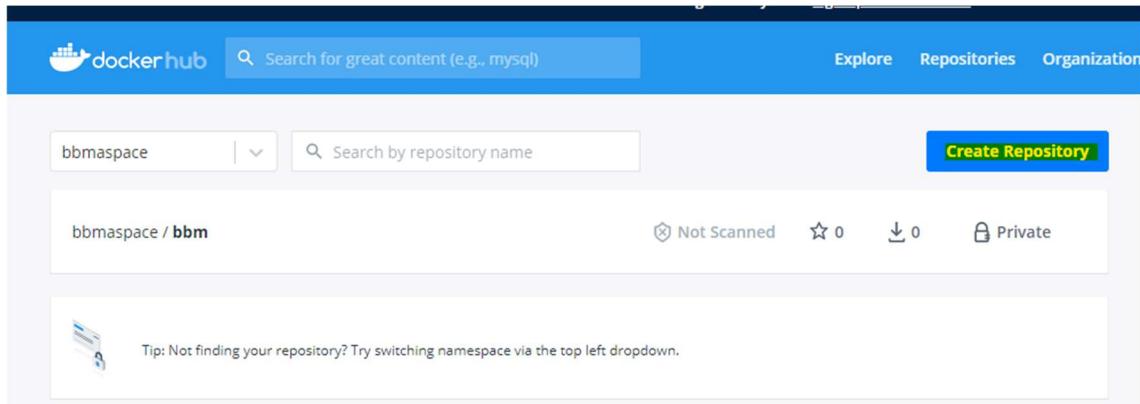
En las opciones que nos ofrece, elegimos free:

Free	Pro	Team
FOR EVERYBODY <ul style="list-style-type: none"> ∞ Unlimited public repositories ✓ Docker Desktop continuously updated ✓ Docker Desktop includes Docker Engine and Kubernetes ✓ Limited container image requests ✓ Two-factor authentication <p>BBM</p> <p>\$0 /month</p> <p>Continue with Free</p>	FOR INDIVIDUALS <ul style="list-style-type: none"> ← Everything in Free ∞ Unlimited private repositories ∞ Unlimited container image requests ✓ 2 parallel builds ✓ 300 monthly Hub image vulnerability scans ✓ Premium customer support for Desktop and Hub <p>\$5 /month</p> <p>With annual plan</p> <p>Buy Now</p>	FOR ORGANIZATIONS <ul style="list-style-type: none"> ← Everything in Pro ∞ Unlimited teams ∞ Unlimited monthly Hub image vulnerability scans ✓ 3 parallel builds per org ✓ Role-based access control ✓ Audit log <p>\$7 user/month</p> <p>Starts at \$25 for 5 users with annual plan</p> <p>Buy Now</p>

NOTA: tendremos que aceptar un mail de verificación y ya podríamos empezar a trabajar con ello

Creando un repositorio

Una vez dentro, creamos el repositorio haciendo clic en **Create Repository**:



Cubrimos los campos de nombre y descripción y lo asociamos a nuestro **GitHub**. Para acabar le damos a **Create&Build**

Create Repository

bbmaspace | **bbm**

REPOSITORIO PARA CLASE ASAX

Visibility

Using 1 of 1 private repositories. [Get more](#)

Public 
 Appears in Docker Hub search results

Private 
 Only visible to you

Build Settings (optional)

Autobuild triggers a new build with every git push to your source code repository. [Learn More](#).

 Connected  Disconnected

 BBMASAX  BBMSPACE 

BUILD RULES 

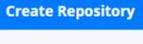
The build rules below specify how to build your source into Docker images.

Source Type	Source	Docker Tag	Dockerfile location	Build Caching
Branch 	master	latest	Dockerfile	 



Queda algo así al crear

bbmaspace |  Search by repository name 

bbmaspace / **bbm**  Not Scanned  0  0  Private

Conectándose a Docker Hub

Por último me conecto desde el terminal con el comando `docker login -u bbmaspace`:

```
root@ubuntu:/home/administrator# docker login -u bbmaspace BBM
Password: ** Message: 10:53:31.221: Remote error from secret service: org.freedesktop.DBus.Error.UnknownMethod: No such interface 'org.freedesktop.Secret.Collection' on object at path /org/freedesktop/secrets/collection/Login/BBM
Error saving credentials: error storing credentials - err: exit status 1, out: 'No such interface 'org.freedesktop.Secret.Collection' on object at path /org/freedesktop/secrets/collection/login'
```

NOTA: me da este error

Buscando por foros, encontré que insertando este comando se resuelve `sudo apt install gnupg2 pass`:

```
root@ubuntu:/home/administrator# sudo apt install gnupg2 pass
Reading package lists... Done BBM
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  linux-hwe-5.4-headers-5.4.0-42 linux-hwe-5.4-headers-5.4.0-58 linux-hwe-5.4-headers-5.4.0-62 linux-hwe-5.4-headers-5.4.0-65
```

...

```
Setting up xclip (0.12+svn84-4build1) ...
Setting up pass (1.7.1-3) ...
Setting up libqrencode3:amd64 (3.4.4-1build1) ...
Setting up gnupg2 (2.2.4-1ubuntu1.4) ...
Setting up qrencode (3.4.4-1build1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Processing triggers for libc-bin (2.27-3ubuntu1.4) ...
```

Pruebo y efectivamente funciona:

```
root@ubuntu:/home/administrator# docker login -u bbmaspace
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
root@ubuntu:/home/administrator#
```

BIBLIOGRAFÍA

https://www.corobori.com/Corobori_Como_encriptar_una_columna_en_una_base_de_datos_SQL_Server-MTI=.aspx

<https://www.sothis.tech/seguridad-en-microsoft-sql-server/#:~:text=La%20seguridad%20en%20SQL%20Server,software%20y%20bases%20de%20datos.>

<https://www.redeszone.net/tutoriales/seguridad/veracrypt-cifra-archivos-gratis/>

<https://docs.microsoft.com/es-es/sql/relational-databases/system-stored-procedures/sp-addlogin-transact-sql?view=sql-server-ver15>

<https://docs.microsoft.com/es-es/sql/relational-databases/security/encryption/sql-server-encryption?view=sql-server-ver15>

<https://docs.microsoft.com/es-es/dotnet/framework/data/adonet/sql/overview-of-sql-server-security>

<https://proteccióndatos-lopd.com/empresas/nueva-ley-protección-datos-2018/>

<https://aleson-itc.com/adaptar-sql-server-al-rgpd-gdpr-serie-gdpr-15/>

<https://docs.microsoft.com/es-es/sql/t-sql/statements/create-master-key-transact-sql?view=sql-server-ver15>

<https://docs.microsoft.com/es-es/sql/t-sql/statements/create-certificate-transact-sql?view=sql-server-ver15>

<https://portswigger.net/web-security/sql-injection>

<https://kinsta.com/es/blog/inyección-sql/>

<https://marcin.gminski.net/blog/ransomware-and-sql-server-how-to-protect-data/>

<https://solutioncenter.apexsql.com/es/encriptacion-de-copias-de-seguridad-sql-server/#:~:text=La%20encriptaci%C3%B3n%20es%20un%20proceso,encriptaci%C3%B3n%20correspondiente%20la%20contrase%C3%A1n.&text=SQL%20Server%202008%20introduce%20Transparent,una%20base%20de%20datos%20complete.>

<https://www.incibe-cert.es/blog/medidas-protección-frente-ataques-denegación-servicio-dos>

<https://www.sqlshack.com/using-dynamic-data-masking-in-sql-server-2016-to-protect-sensitive-data/>

<https://www.sqlshack.com/dynamic-data-masking-in-sql-server/>

[https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc938202\(v=technet.10\)?redirectedfrom=MSDN](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc938202(v=technet.10)?redirectedfrom=MSDN)

[https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc938214\(v=technet.10\)](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc938214(v=technet.10))

<https://www.itprotoday.com/sql-server/3-free-tools-prevent-sql-injection-attacks>

<https://docs.microsoft.com/es-es/sql/relational-databases/system-dynamic-management-views/sys-dm-database-encryption-keys-transact-sql?view=sql-server-ver15>

<https://losvirus.es/las-mejores-herramientas-de-eliminacion-de-ransomware/>

<https://comodossalstore.com/resources/hashing-vs-encryption-simplifying-the-differences/>

<https://docs.microsoft.com/es-es/sql/relational-databases/user-defined-functions/user-defined-functions?view=sql-server-ver15>

<https://docs.microsoft.com/es-es/sql/relational-databases/security/row-level-security?view=sql-server-ver15>

<https://docs.microsoft.com/es-es/sql/relational-databases/security/encryption/always-encrypted-database-engine?view=sql-server-ver15>

<https://www.sqlservercentral.com/articles/sql-server-2016-always-encrypted>

<https://docs.microsoft.com/es-es/sql/t-sql/functions/system-functions-transact-sql?view=sql-server-ver15>

<https://docs.microsoft.com/es-es/sql/relational-databases/security/sql-vulnerability-assessment?view=sql-server-ver15>

<https://docs.microsoft.com/en-us/sql/relational-databases/security/auditing/sql-server-audit-database-engine?view=sql-server-ver15#:~:text=The%20SQL%20Server%20Audit%20object,the%20output%20of%20the%20results.>

<https://www.xataka.com/otros/docker-a-kubernetes-entendiendo-que-contenedores-que-mayores-revoluciones-industria-desarrollo>

[https://es.wikipedia.org/wiki/Docker_\(software\)](https://es.wikipedia.org/wiki/Docker_(software))

<https://www.redhat.com/es/topics/containers/what-is-docker>

<https://www.ibm.com/blogs/systems/mx-es/tag/docker/>

<https://www.redhat.com/es/topics/containers/what-is-kubernetes>

<https://www.adictosaltrabajo.com/2015/11/25/docker-para-bases-de-datos/>

<https://clouding.io/hc/es/articles/360010283060-Trabajando-con-im%C3%A1genes-en-Docker>

<https://danielcastanera.com/comandos-utiles-en-docker/>

<https://docs.docker.com/>

<https://docs.docker.com/engine/install/ubuntu/>

<https://www.sololinux.es/instalar-visual-studio-code-en-ubuntu-y-derivados/>

https://hub.docker.com/_/mysql

https://hub.docker.com/_/mongo

<https://josejuansanchez.org/bd/practica-05/index.html>

<https://mariadb.com/kb/en/installing-and-using-mariadb-via-docker/>

https://hub.docker.com/_/postgres

https://apuntes-snicker.readthedocs.io/es/latest/programacion/postgresql/comandos_consola_sql.html

<https://www.bmc.com/blogs/mongodb-docker-container/>

<https://hub.docker.com/>

