

BASE DE DATOS EN ORACLE Y SQL SERVER



**Mi BD-2: Maniobras Militares
ASXBD – ASIR**
Juan J. Gómez Fernández
IES Fernando Wirtz Suárez

Indice:

1. Deuda del Segundo Trimestre:

- ✓ [Particiones](#)
- ✓ [Procedimiento Almacenado](#)
- ✓ [Tablas Temporales](#)

SEGURIDAD:

2. Encriptación

- ◆ [Herramienta Veracrypt](#)
- ◆ [Encriptación de columnas de BD](#)
- ◆ [Encriptación de Backups de BD](#)
- ◆ [Encriptación de BD TDE](#)
- ◆ [Data Masking](#)
- ◆ [Row Encryption](#)
- ◆ [Always Encrypted 101](#)
- ◆ [SSMS \(Data Discovery and Classification y Vulnerability and Assesment\)](#)

3. Auditoría

- [Auditoría de Servidor y Especificaciones de Servidor](#)
- [Especificaciones de Bases de Datos](#)

4. [Legislación \(GPR General Data Protection Regulation - EU\)](#)

5. Ataques

- ➔ [DDos](#)
- ➔ [Injection SQL](#)
- ➔ [Ransomware](#)
- ➔ [Herramientas](#)

6. [DOCKER](#)

- [Comandos básicos de Docker](#)
- [MySQL en DOCKER](#)
- [Lanzar múltiples contenedores en Docker](#)
- [DockerHUB](#)

Particiones:

Por parte de la sección del Archivo nos piden una Base de Datos que permita registrar un índice de los documentos oficiales, así como poder registrar el de años anteriores para realizar un histórico, que sería el primer paso para digitalizar los documentos.

Para ello vamos a crear la BD y sus filegroup:

```
DROP DATABASE IF EXISTS gfjj_archivo
GO

CREATE DATABASE [gfjj_archivo]
    ON PRIMARY ( NAME = 'gfjj_archivo',
        FILENAME = 'C:\Data\gfjj_archivo.mdf' ,
        SIZE = 15360KB , MAXSIZE = UNLIMITED, FILEGROWTH = 0)
    LOG ON ( NAME = 'gfjj_archivo_log',
        FILENAME = 'C:\Data\gfjj_archivo_log.ldf' ,
        SIZE = 10176KB , MAXSIZE = 2048GB , FILEGROWTH = 10%)
GO

USE gfjj_archivo
GO

ALTER DATABASE [gfjj_archivo] ADD FILEGROUP [GFJJFILEGROUP_Historico]
GO
ALTER DATABASE [gfjj_archivo] ADD FILEGROUP [GFJJFILEGROUP_2020]
GO
ALTER DATABASE [gfjj_archivo] ADD FILEGROUP [GFJJFILEGROUP_2021]
GO
```

Ahora vamos a comprobar que están creados:

The screenshot shows a SQL Server Management Studio window with a results grid. The query `select * from sys.filegroups` has been run, and the results are displayed in a table with the following data:

	name	data_space_id	type	type_desc	is_default	is_system	filegroup_guid	log_fileg
1	PRIMARY	1	FG	ROWS_FILEGROUP	1	0	NULL	NULL
2	GFJJFILEGROUP_Historico	2	FG	ROWS_FILEGROUP	0	0	0F7A0555-DD21-4C7E-AB3C-38AFC661F333	NULL
3	GFJJFILEGROUP_2020	3	FG	ROWS_FILEGROUP	0	0	2483DF7E-98B9-4A53-864F-65ECF6D71DC0	NULL
4	GFJJFILEGROUP_2021	4	FG	ROWS_FILEGROUP	0	0	449FC31F-E6D0-4A4C-A61F-91C2393E6129	NULL

Creamos los ficheros base del filegroup:

```
ALTER DATABASE [gfjj_archivo] ADD FILE ( NAME = 'Nuevas_Altas_gfjj', FILENAME = 'c:\Data\Nuevas_Altas_gfjj.ndf', SIZE = 5MB, MAXSIZE = 100MB, FILEGROWTH = 2MB ) TO FILEGROUP [GFJJFILEGROUP_Historico]
GO
ALTER DATABASE [gfjj_archivo] ADD FILE ( NAME = 'ARCHIVO_2020', FILENAME = 'c:\Data\ARCHIVO_2020.ndf', SIZE = 5MB, MAXSIZE = 100MB, FILEGROWTH = 2MB ) TO FILEGROUP [GFJJFILEGROUP_2020]
GO
ALTER DATABASE [gfjj_archivo] ADD FILE ( NAME = 'ARCHIVO_2021', FILENAME = 'c:\Data\ARCHIVO_2021.ndf', SIZE = 5MB, MAXSIZE = 100MB, FILEGROWTH = 2MB ) TO FILEGROUP [GFJJFILEGROUP_2021]
```

GO

Podemos comprobar que están creados:

	file_id	file_guid	type	type_desc	data_space_id	name	physical_name	state	st
1	1	64A648D4-5238-45C0-BD66-351DE23E439B	0	ROWS	1	gfjj_archivo	C:\Data\gfjj_archivo.mdf	0	0
2	2	B80BC4C0-7C3F-4DF9-BF1B-F110EDFBCA68	1	LOG	0	gfjj_archivo_log	C:\Data\gfjj_archivo_log.ldf	0	0
3	3	A04ADA49-36A3-423E-985F-D1F90B84DCE7	0	ROWS	2	Nuevas_Altas_gfjj	c:\Data\Nuevas_Altas_gfjj.ndf	0	0
4	4	2D850245-B65C-452E-871F-09F603BE3F62	0	ROWS	3	ARCHIVO_2020	c:\Data\ARCHIVO_2020.ndf	0	0
5	5	41CAC877-88B4-4AAF-94C9-A5DB39B6C1AC	0	ROWS	4	ARCHIVO_2021	c:\Data\ARCHIVO_2021.ndf	0	0

Ahora vamos a crear una función de partición que abarque el año 2020:

```
CREATE PARTITION FUNCTION PART_FUNCTION_gfjj (datetime)
AS RANGE RIGHT
    FOR VALUES ('2020-01-01', '2021-01-01')
GO
```

Luego creamos el esquema asociado a los 3 filegroup:

```
CREATE PARTITION SCHEME PART_SCHEMA_gfjj
AS PARTITION PART_FUNCTION_gfjj
    TO (GFJJFILEGROUP_Historico, GFJJFILEGROUP_2020, GFJJFILEGROUP_2021)
GO
```

Hecho eso, creamos la tabla que nos permita registrar los documentos:

```
DROP TABLE IF EXISTS GFJJ_Registro
GO
CREATE TABLE GFJJ_Registro (
    ID INT NOT NULL IDENTITY,
    Nombre varchar(40) NULL,
    Almacenaje varchar(20) NULL,
    fecha_documento datetime)
ON PART_SCHEMA_GFJJ
    (fecha_documento)
GO
```

Ahora introducimos valores para poder operar con ella:

```
INSERT INTO GFJJ_Registro (Nombre, Almacenaje, fecha_documento)
Values ('BOD Nº 1', 'Secretaria', '2020-01-02'),
('BOD Nº 2', 'Secretaria', '2020-01-03'),
('Sancion Disciplinaria', 'Personal', '2020-05-14'),
('Revista de Defensa', 'OFAPAR', '2020-01-02'),
('Solicitud de Condecoraciones', 'Archivo', '1998-03-13'),
('BOD Nº 1', 'Secretaria', '2021-01-02'),
('Orden de Operaciones Operacion Balmis', 'Operaciones', '2021-02-16'),
('Revista de Defensa', 'Archivo', '2005-08-01'),
('Compromiso Soldado Gómez', 'Personal', '2004-03-24')
GO
```

Comprobamos que están creados correctamente:

```
SQLQuery1.sql - 192...jj_archivo (sa (62)) * X
SELECT * FROM GFJJ_Registro
GO
```

Results

ID	Nombre	Almacenaje	fecha_documento
2	Revista de Defensa	Archivo	2005-08-01 00:00:00.000
3	Compromiso Soldado Gómez	Personal	2004-03-24 00:00:00.000
4	BOD Nº 1	Secretaria	2020-01-02 00:00:00.000
5	BOD Nº 2	Secretaria	2020-01-03 00:00:00.000
6	Sancion Disciplinaria	Personal	2020-05-14 00:00:00.000
7	Revista de Defensa	OFAPAR	2020-01-02 00:00:00.000
8	BOD Nº 1	Secretaria	2021-01-02 00:00:00.000
9	Orden de Operaciones Operacion Balmis	Operaciones	2021-02-16 00:00:00.000

Para comprobar que está repartido correctamente por las distintas particiones, ponemos:

```
SELECT *, $Partition.PART_FUNCTION_GFJJ(fecha_documento) AS Partition
FROM GFJJ_Registro
GO
```

Vemos el resultado:

```
SELECT *, $Partition.PART_FUNCTION_GFJJ(fecha_documento) AS Partition
FROM GFJJ_Registro
GO
```

Results

ID	Nombre	Almacenaje	fecha_documento	Partition
2	Revista de Defensa	Archivo	2005-08-01 00:00:00.000	1
3	Compromiso Soldado Gómez	Personal	2004-03-24 00:00:00.000	1
4	BOD Nº 1	Secretaria	2020-01-02 00:00:00.000	2
5	BOD Nº 2	Secretaria	2020-01-03 00:00:00.000	2
6	Sancion Disciplinaria	Personal	2020-05-14 00:00:00.000	2
7	Revista de Defensa	OFAPAR	2020-01-02 00:00:00.000	2
8	BOD Nº 1	Secretaria	2021-01-02 00:00:00.000	3
9	Orden de Operaciones Operacion Balmis	Operaciones	2021-02-16 00:00:00.000	3

Podemos comprobar que lo que pertenece al 2021 va a la partición 3, lo que pertenece al 2020 va a la partición 2 y todo lo anterior corresponde a la partición 1.

Si queremos comprobar cuantos (cantidad, pero sin ver el nominal) registros tiene cada partición podemos realizar la siguiente consulta:

```
DECLARE @TableName NVARCHAR(200) = 'GFJJ_Registro'
SELECT SCHEMA_NAME(o.schema_id) + '.' + OBJECT_NAME(i.object_id) AS [object] ,
p.partition_number AS [p#], GFJJFILEGROUP.name AS [filegroup], p.rows , au.total_pages AS pages ,
CASE boundary_value_on_right WHEN 1 THEN 'menor que' ELSE 'menor o igual' END as comparison ,
rv.value , CONVERT (VARCHAR(6), CONVERT (INT, SUBSTRING (au.first_page, 6, 1) +
```

```

SUBSTRING (au.first_page, 5, 1)) + ':' + CONVERT (VARCHAR(20), CONVERT (INT, SUBSTRING (au.-first_page, 4, 1) + SUBSTRING (au.first_page, 3, 1) + SUBSTRING (au.first_page, 2, 1) + SUBSTRING (au.first_page, 1, 1))) AS first_page FROM sys.partitions p INNER JOIN sys.indexes i ON p.object_id = i.object_id AND p.index_id = i.index_id INNER JOIN sys.objects o ON p.object_id = o.object_id INNER JOIN sys.system_internals_allocation_units au ON p.partition_id = au.container_id INNER JOIN sys.partition_schemes ps ON ps.data_space_id = i.data_space_id INNER JOIN sys.partition_functions f ON f.function_id = ps.function_id INNER JOIN sys.destination_data_spaces dds ON dds.partition_scheme_id = ps.data_space_id AND dds.destination_id = p.partition_number INNER JOIN sys.filegroups GFJJFILEGROUP ON dds.data_space_id = GFJJFILEGROUP.data_space_id LEFT OUTER JOIN sys.partition_range_values rv ON f.function_id = rv.function_id AND p.partition_number = rv.boundary_id WHERE i.index_id < 2 AND o.object_id = OBJECT_ID(@TableName);
GO

```

Nos muestra el siguiente contenido:

	object	p#	filegroup	rows	pages	comparison	value	first_page
1	dbo.GFJJ_Registro	1	GFJJFILEGROUP_Historico	3	9	menor que	2020-01-01 00:00:00.000	3:8
2	dbo.GFJJ_Registro	2	GFJJFILEGROUP_2020	4	9	menor que	2021-01-01 00:00:00.000	4:8
3	dbo.GFJJ_Registro	3	GFJJFILEGROUP_2021	2	9	menor que	NULL	5:8

Ahora vamos a realizar operaciones con las particiones. Para tener más opciones, vamos a crear un filegroup que nos permita almacenar los documentos del año que viene.

```
ALTER DATABASE GFJJ_archivo ADD FILEGROUP GFJJ_particiones;
```

```

ALTER DATABASE GFJJ_archivo
ADD FILE
(
    NAME = GFJJ_particiones,
    FILENAME = 'c:\Data\GFJJ_particiones.ndf',
    SIZE = 5MB,
    MAXSIZE = 100MB,
    FILEGROWTH = 5MB
)
TO FILEGROUP GFJJ_particiones;

```

```
ALTER PARTITION SCHEME PART_SCHEMA_gfjj
NEXT USED GFJJ_particiones;
```

SPLIT

Vamos a modificar la función, haciendo un SPLIT de los documentos de 2022 en adelante. Al hacer esto, y tras haber creado el filegroup anterior y especificado que se utilice el de particiones, debería de enviar los nuevos registros (creados en ese rango) en la partición nueva.

```

ALTER PARTITION FUNCTION PART_FUNCTION_gfjj()
    SPLIT RANGE ('2022-01-01');
GO

```

```

INSERT INTO GFJJ_Registro (Nombre, Almacenaje, fecha_documento)
    Values ('BOD Nº 1', 'Secretaria', '2022-01-02')
GO

```

```

SELECT *, $Partition.PART_FUNCTION_gfjj(fecha_documento) as PARTITION
FROM GFJJ_Registro
GO

```

Podemos comprobar que efectivamente, nos lo añade en la partición nueva:

```
ALTER PARTITION FUNCTION PART_FUNCTION_GFJJ()
    SPLIT RANGE ('2022-01-01');
GO

SELECT *,$Partition.PART_FUNCTION_GFJJ(fecha_documento) as PARTITION
FROM GFJJ_Registro
GO
```

100 %

Results Messages

ID	Nombre	Almacenaje	fecha_documento	PARTITION
4	BOD Nº 1	Secretaria	2020-01-02 00:00:00.000	2
5	BOD Nº 2	Secretaria	2020-01-03 00:00:00.000	2
6	Sancion Disciplinaria	Personal	2020-05-14 00:00:00.000	2
7	Revista de Defensa	OFAPAR	2020-01-02 00:00:00.000	2
8	BOD Nº 1	Secretaria	2021-01-02 00:00:00.000	3
9	Orden de Operaciones Oper...	Operacio...	2021-02-16 00:00:00.000	3
10	BOD Nº 1	Secretaria	2022-01-02 00:00:00.000	4

Podemos visualizarlo con la otra consulta:

```
DECLARE @TableName NVARCHAR(200) = N'GFJJ_Registro'
SELECT SCHEMA_NAME(o.schema_id) + '.' + OBJECT_NAME(i.object_id) AS [object] , p.partition_number AS [p#] , GF
ON P.object_id = o.object_id INNER JOIN sys.system_internals_allocation_units au ON p.partition_id = au.contai
GO
```

00 %

Results Messages

object	p#	filegroup	rows	pages	comparision	value	first_page
dbo.GFJJ_Registro	1	GFJJFILEGROUP_Historico	3	9	menor que	2020-01-01 00:00:00.000	3:8
dbo.GFJJ_Registro	2	GFJJFILEGROUP_2020	4	9	menor que	2021-01-01 00:00:00.000	4:8
dbo.GFJJ_Registro	3	GFJJFILEGROUP_2021	2	9	menor que	2022-01-01 00:00:00.000	5:8
dbo.GFJJ_Registro	4	GFJJ_particiones	1	9	menor que	NULL	6:8

MERGE

Al utilizar la instrucción Merge, vamos a fusionar el contenido de un filegroup en el filegroup anterior. En este caso vamos a realizar el Merge con los registros del año 2020.

```
ALTER PARTITION FUNCTION PART_FUNCTION_GFJJ ()
MERGE RANGE ('2020-01-01');
GO

SELECT *,$Partition.PART_FUNCTION_GFJJ(fecha_documento) as PARTITION
FROM GFJJ_Registro
GO
```

```

SELECT *,$Partition_PART_FUNCTION_gfjj(fecha_documento) as PARTITION
FROM GFJJ_Registro
GO

```

100 %

	ID	Nombre	Almacenaje	fecha_documento	PARTITION
1	5	Solicitud de Condecoraciones	Archivo	1998-03-13 00:00:00.000	1
2	8	Revista de Defensa	Archivo	2005-08-01 00:00:00.000	1
3	9	Compromiso Soldado Gómez	Personal	2004-03-24 00:00:00.000	1
4	1	BOD Nº 1	Secretaria	2020-01-02 00:00:00.000	1
5	2	BOD Nº 2	Secretaria	2020-01-03 00:00:00.000	1
6	3	Sanción Disciplinaria	Personal	2020-05-14 00:00:00.000	1
7	4	Revista de Defensa	OFAPAR	2020-01-02 00:00:00.000	1
8	6	BOD Nº 1	Secretaria	2021-01-02 00:00:00.000	2
9	7	Orden de Operaciones Oper...	Operacio...	2021-02-16 00:00:00.000	2
10	10	BOD Nº 1	Secretaria	2022-01-02 00:00:00.000	3

Como vemos, los registros de 2020 antes formaban parte de la partición 2 y ahora forman parte de la 1. Vamos a consultar los filegroup existentes.

```

DECLARE @TableName NVARCHAR(200) = N'GFJJ_Registro'
SELECT SCHEMA_NAME(o.schema_id) + '.' + OBJECT_NAME(i.object_id) AS [object] , p.partition_number AS [p#] , GF
ON p.object_id = o.object_id INNER JOIN sys.system_internals_allocation_units au ON p.partition_id = au.container_id
GO

```

100 %

object	p#	filegroup	rows	pages	comparison	value	first_page
dbo.GFJJ_Registro	1	GFJJFILEGROUP_Historico	7	9	menor que	2021-01-01 00:00:00.000	3:8
dbo.GFJJ_Registro	2	GFJJFILEGROUP_2021	2	9	menor que	2022-01-01 00:00:00.000	5:8
dbo.GFJJ_Registro	3	GFJJ_particiones	1	9	menor que	NULL	6:8

Como vemos, la cantidad de registros ha cambiado. El filegroup de 2020 ha desaparecido, pero todos sus registros pasaron al anterior filegroup (histórico). Esta operación sería muy interesante para realizar (en este tipo de BD) a principios de mes/año.

SWITCH

La BD está creciendo, así que por motivos de logística vamos a crear una tabla que nos permita almacenar el histórico, y dejamos la tabla de Registro para los más recientes.

```

CREATE TABLE GFJJ_Switch (
    ID INT NOT NULL IDENTITY,
    Nombre varchar(40) NULL,
    Almacenaje varchar(20) NULL,
    fecha_documento datetime)
    ON GFJJFILEGROUP_Historico
GO

ALTER TABLE GFJJ_Registro
SWITCH Partition 1 to GFJJ_Switch
GO

```

Al hacer la consulta de la tabla Registro vemos que le falta contenido (y la partición 1):

```
SELECT *, $Partition.PART_FUNCTION_gfjj(fecha_documento) as PARTITION
FROM GFJJ_Registro
GO
```

	ID	Nombre	Almacenaje	fecha_documento	PARTITION
1	6	BOD Nº 1	Secretaria	2021-01-02 00:00:00.000	2
2	7	Orden de Operaciones Operacion Balmis	Operaciones	2021-02-16 00:00:00.000	2
3	10	BOD Nº 1	Secretaria	2022-01-02 00:00:00.000	3

Vamos a realizar la misma consulta en la tabla Switch (la que almacena el histórico):

```
SELECT *, $Partition.PART_FUNCTION_gfjj(fecha_documento) as PARTITION
FROM GFJJ_Switch
GO
```

	ID	Nombre	Almacenaje	fecha_documento	PARTITION
1	5	Solicitud de Condecoraciones	Archivo	1998-03-13 00:00:00.000	1
2	8	Revista de Defensa	Archivo	2005-08-01 00:00:00.000	1
3	9	Compromiso Soldado Gómez	Personal	2004-03-24 00:00:00.000	1
4	1	BOD Nº 1	Secretaria	2020-01-02 00:00:00.000	1
5	2	BOD Nº 2	Secretaria	2020-01-03 00:00:00.000	1
6	3	Sancion Disciplinaria	Personal	2020-05-14 00:00:00.000	1
7	4	Revista de Defensa	OFAPAR	2020-01-02 00:00:00.000	1

Podemos comprobar que todos los registros de la partición 1 que estaban almacenados en la tabla Tabla Registro, pasaron a la tabla Switch, que pasa a disponer de su partición.

Al consultar los Filegroups de la tabla Registro, podemos ver que el Filegroup Historico no ha desaparecido, si no que contiene 0 registros.

```
DECLARE @TableName NVARCHAR(200) = N'GFJJ_Registro'
SELECT SCHEMA_NAME(o.schema_id) + '.' + OBJECT_NAME(i.object_id) AS [object], p.partition_number AS [p#], GF
ON p.object_id = o.object_id INNER JOIN sys.system_internals_allocation_units au ON p.partition_id = au.contai
GO
```

object	p#	filegroup	rows	pages	comparison	value	first_page
1 dbo.GFJJ_Registro	1	GFJJFILEGROUP_Historico	0	0	menor que	2021-01-01 00:00:00.000	0:0
2 dbo.GFJJ_Registro	2	GFJJFILEGROUP_2021	2	9	menor que	2022-01-01 00:00:00.000	5:8
3 dbo.GFJJ_Registro	3	GFJJ_particiones	1	9	menor que	NULL	6:8

TRUNCATE:

Nuestro Ilustrísimo Coronel nos acaba de informar de que acaba de recibir un atento correo electrónico de que hay que justificar las maniobras del año pasado. Nos ordena que eliminemos todos los documentos del año pasado inmediatamente. Para ello vamos a emplear la función TRUNCATE.

```
TRUNCATE TABLE GFJJ_Registro  
    WITH (PARTITIONS (2));  
GO
```

Vamos a asegurarnos de que hemos eliminado el contenido:

```
SELECT *, $Partition.PART_FUNCTION_gfjj(fecha_documento) as PARTITION  
FROM GFJJ_Registro  
GO
```

The screenshot shows the SSMS interface with the following details:

- Query pane: Contains the T-SQL code for selecting data from GFJJ_Registro using the PARTITION function.
- Results pane: Shows a table with the following data:

ID	Nombre	Almacenaje	fecha_documento	PARTITION	
1	10	BOD N° 1	Secretaria	2022-01-02 00:00:00.000	3

Procedimiento Almacenado:

La sección de Personal está realizando un procedimiento administrativo contra un soldado que llegó tarde y bajo efectos del alcohol a una guardia. Nos solicitan urgentemente que realicemos una tabla donde registrar las sanciones, así como otra tabla que regule la correspondiente sanción diaria según el empleo.

Primero creamos la tabla que asigna la sanción económica diaria en función del empleo:

```
CREATE TABLE dbo.COSTE
(Empleo VARCHAR (9) PRIMARY KEY,
 San_dia INT NOT NULL,
);

INSERT INTO dbo.COSTE (Empleo, San_dia) VALUES
('Soldado',15),
('Cabo',18),
('Sargento',38)
GO
```

Una vez creada la tabla, vamos a actualizar la tabla dbo.MILITAR para que podamos asignarles un empleo. Hasta ahora nos era irrelevante el empleo de cada uno y no lo registrábamos, pero para la realización de esta nueva tarea nos es necesario.

```
ALTER TABLE dbo.MILITAR ADD Empleo VARCHAR (9) NULL;
UPDATE dbo.MILITAR SET Empleo = 'Soldado' WHERE DNI = '32697849Z';
UPDATE dbo.MILITAR SET Empleo = 'Cabo' WHERE DNI = '32374058Y';
UPDATE dbo.MILITAR SET Empleo = 'Sargento' WHERE DNI = '32409123L';
```

Podemos observar que ya tienen los empleos vinculados correctamente:

	DNI	NOMBRE	NS_Am	Destino	Empleo
1	32374058Y	Jorge Gonzalez Bellon	FN04716	SEG	Cabo
2	32409123L	Guillermo Balaña Perez	5650G	PN	Sargento
3	32637357Y	Alfredo Perez	FN14589	SEG	NULL
4	32697849Z	Juan Gomez Fernandez	FN04770	PLM	Soldado
5	32850642V	Hector Criado Garcia	FN045697	EOS	NULL
6	32941523A	Francisco Bermudez Suarez	OT5780	EOS	NULL
7	36280541X	Lucia Cabanas Lopez	FN14591	EOS	NULL

Ahora vamos a crear la tabla que permita llevar un control del coste de la Sanción:

```
CREATE TABLE dbo.SANCION
(Cod_San INT PRIMARY KEY IDENTITY,
 DNI VARCHAR (9) NOT NULL,
 FOREIGN KEY (DNI) REFERENCES dbo.MILITAR(DNI),
 San_Ini DATE,
 San_Fin DATE,
 San_pago INT
);
```

Lo siguiente será crear un Procedimiento Almacenado que nos permita contar los días que tiene cada sanción, así como calcular el coste de la sanción en función de esos días y el empleo.

```

DROP PROCEDURE IF EXISTS Nueva_Sancion;
CREATE OR ALTER PROCEDURE dbo.Nueva_Sancion
    @dni AS VARCHAR(9),
    @fecha AS DATE,
    @dias AS INT
AS
BEGIN
DECLARE @fecha2 AS DATE;
DECLARE @san AS INT;
DECLARE @empleo AS VARCHAR(9);
DECLARE @pago AS INT;
DECLARE @nombre AS VARCHAR(30);
SET @empleo = (SELECT Empleo FROM MILITAR WHERE DNI = @dni);
SELECT @fecha2 = DATEADD (day, @dias,@fecha);
SELECT @pago = (@dias * (SELECT San_dia FROM COSTE WHERE Empleo = @empleo));
SELECT @nombre = (SELECT NOMBRE FROM MILITAR WHERE DNI = @dni);
INSERT INTO dbo.SANCION (DNI, San_Ini, San_Fin, San_pago) VALUES (@dni, @fecha, @fe-
cha2,@pago);
END;

```

Ahora vamos a probar que funciona correctamente:

```
EXEC dbo.Nueva_Sancion '32374058Y', '20210320', 12;
```

SELECT * FROM dbo.SANCION				
Cod_San	DNI	San_Ini	San_Fin	San_pago
1	1	32374058Y	2021-03-20	2021-04-01

Vamos a hacer otra prueba:

EXEC dbo.Nueva_Sancion '32697849Z', '20080404', 8;				
SELECT * FROM dbo.SANCION				
Cod_San	DNI	San_Ini	San_Fin	San_pago
1	1	32374058Y	2021-03-20	2021-04-01
2	2	32697849Z	2008-04-04	2008-04-12
				120

Tablas Temporales:

El Almirante de Personal acaba de publicar un mensaje Oficial que requiere a las Unidades que faciliten más días libres en función de las guardias, a discreción del Jefe de Unidad, para facilitar la vida laboral y personal. En este sentido, nuestro Coronel otorga un día de Descanso Adicional por cada 7 guardias realizadas. Por parte de la Oficina de Personal nos piden una tabla en la que ir anotando cada vez que alguien hace una guardia, de modo que se pueda ir llevando una cuenta de cuantas guardias realiza cada uno, tanto para efectos estadísticos como para llevar un control de los días que le corresponden a cada uno (que hay mucho listo suelto).

```
USE gfjj_archivo  
GO
```

Antes de nada, vamos a proceder a realizar la tabla que nos permita administrar el control de los DAOS, en donde anotaremos simplemente los datos del personal.

```
IF OBJECT_ID('dbo.DAOS', 'U') IS NOT NULL  
    DROP TABLE dbo.DAOS;  
CREATE TABLE DAOS(  
DNI VARCHAR(9) PRIMARY KEY NOT NULL,  
Nombre VARCHAR (100) NOT NULL,  
Apellidos VARCHAR (100) NOT NULL,  
DAO INT NULL);
```

Vamos a crear una tabla así como su tabla temporal, que permita llevar el histórico.

```
IF OBJECT_ID('tempdb.dbo.Personal', 'U') IS NOT NULL  
    DROP TABLE tempdb.dbo.Personal;  
CREATE TABLE Personal(  
ID INT PRIMARY KEY NOT NULL IDENTITY,  
DNI VARCHAR(9),  
Guardias INT,  
Inicio datetime2 GENERATED ALWAYS AS ROW START NOT NULL,  
Fin datetime2 GENERATED ALWAYS AS ROW END NOT NULL,  
PERIOD FOR SYSTEM_TIME (Inicio,Fin)  
)  
WITH (SYSTEM_VERSIONING = ON(HISTORY_TABLE = dbo.PersonalHistorico));
```

Ya tenemos creadas las tablas. Ahora vamos a facilitarles la vida a los compañeros de Personal, creando dos procedimientos almacenados que les permitan insertar los datos cómodamente. Primero vamos a crear un Procedimiento Almacenado que permita introducir los datos cuando alguien realiza su primera guardia.

```
DROP PROCEDURE Alta_Des;  
CREATE PROCEDURE Alta_Des  
    @dni AS VARCHAR(9),  
    @nombre AS VARCHAR(100),  
    @apellidos AS VARCHAR(100)  
AS  
BEGIN  
    INSERT INTO dbo.DAOS (DNI, Nombre, Apellidos) VALUES (@dni, @nombre, @apellidos);  
    INSERT INTO dbo.Personal (DNI) VALUES (@dni);  
    PRINT @dni + ' ha realizado su primera guardia'  
END
```

Con esto lo que hacemos simplemente es que cargamos el dni, nombre y apellidos de una persona cuando realiza su primera guardia, y el procedimiento le carga la guardia y a mayores lo registra en ambas tablas.

Ahora vamos a hacer un Procedimiento Almacenado, que cuando alguien haga una guardia le pongamos su DNI y le anote esa guardia, actualizando las tablas de modo que por cada 7 guardias nos asigne un día de Descanso Adicional.

```

DROP PROCEDURE Descansos;
CREATE PROCEDURE Descansos
    @dni AS VARCHAR(9)
AS
DECLARE @cuenta AS INT;
DECLARE @dias AS INT;
BEGIN
UPDATE dbo.Personal SET Guardias = Guardias+1 WHERE DNI = @dni;
SET @cuenta = (SELECT COUNT(DNI) FROM dbo.PersonalHistorico WHERE DNI = @dni);
SET @dias = @cuenta/7;
UPDATE dbo.DAOs SET DAO = @dias WHERE DNI = @dni;
PRINT @dni + ' dispone de ' + CAST(@dias AS VARCHAR) + ' días adicionales.'
END

```

Vamos a comprobar que funciona:

```

EXEC Alta_Des '32697849Z', 'Juan Jose', 'Gomez Fernandez';

```

Messages

(1 row affected)

(1 row affected)
32697849Z ha realizado su primera guardia

Podemos comprobar que está creado correctamente.

```

SELECT * FROM dbo.DAOs;

```

Results

DNI	Nombre	Apellidos	DAO
32697849Z	Juan Jose	Gomez Fernandez	NULL

Como podemos comprobar, nos ha registrado los valores en ambas tablas.

```

SELECT * FROM Personal;

```

Results

ID	DNI	Guardias	Inicio	Fin
1	32697849Z	NULL	2021-05-23 16:05:37.6333081	9999-12-31 23:59:59.9999999

Ahora vamos a probar a introducirle más guardias, a ver si realiza la operación correctamente.

```

EXEC Descansos '32697849Z';

```

Messages

(1 row affected)

(1 row affected)
32697849Z dispone de 0 dias adicionales.

En principio nos comunica que dispone de 0 días adicionales. Vamos a comprobar cómo queda en las Tablas.

```
SELECT * FROM dbo.DAOS;
```

	DNI	Nombre	Apellidos	DAO
1	32697849Z	Juan Jose	Gomez Fernandez	0

En la tabla DAOS nos figura que le corresponden 0 DAOs.

```
SELECT * FROM Personal;
```

	ID	DNI	Guardias	Inicio	Fin
1	1	32697849Z	NULL	2021-05-23 16:08:57.0400807	9999-12-31 23:59:59.9999999

En la tabla Personal nos figura el registro “Inicio” con otra hora diferente. Vamos a consultar la tabla histórica.

```
SELECT * FROM dbo.PersonalHistorico;
```

	ID	DNI	Guardias	Inicio	Fin
1	1	32697849Z	NULL	2021-05-23 16:05:37.6333081	2021-05-23 16:08:57.0400807

Podemos comprobar que el registro que estaba antes en la tabla Personal, ahora pasa a estar en la tabla Histórica. Eso nos indica que en la tabla Personal solamente vamos a almacenar un valor único como DNI, pero en la histórica guardaremos todos los cambios que se hayan realizado.

Ahora vamos a cargarle 6 guardias más y comprobamos el resultado:

```
EXEC Descansos '32697849Z';
```

(1 row affected)

(1 row affected)

32697849Z dispone de 1 días adicionales.

Nos está indicando que dispone de 1 día de descanso adicional. Vamos a cotejarlo en las Tablas.

En la tabla DAOS nos figura ese día Adicional.

	DNI	Nombre	Apellidos	DAO
1	32697849Z	Juan Jose	Gomez Fernandez	1

La tabla Personal sigue siendo “similar”, ya que solamente nos muestra un registro (en este caso, sería el último día de guardia que realizó)

	ID	DNI	Guardias	Inicio	Fin
1	1	32697849Z	NULL	2021-05-23 16:15:33.6958314	9999-12-31 23:59:59.9999999

Pero, ahora sí, podemos ver en la tabla histórica que guarda todos los registros anteriores (6)

	ID	DNI	Guardias	Inicio	Fin
1	1	32697849Z	NULL	2021-05-23 16:05:37.6333081	2021-05-23 16:08:57.0400807
2	1	32697849Z	NULL	2021-05-23 16:08:57.0400807	2021-05-23 16:13:04.8523482
3	1	32697849Z	NULL	2021-05-23 16:13:04.8523482	2021-05-23 16:14:47.2106138
4	1	32697849Z	NULL	2021-05-23 16:14:47.2106138	2021-05-23 16:14:52.3362029
5	1	32697849Z	NULL	2021-05-23 16:14:52.3362029	2021-05-23 16:15:28.2445803
6	1	32697849Z	NULL	2021-05-23 16:15:28.2445803	2021-05-23 16:15:31.0386778

Analizando lo que nos pueden otorgar o no las tablas temporales, me gustaría recalcar:

- Las tablas temporales son MUY interesantes para realizar consultas estadísticas o registrar los cambios. En este caso lo he empleado para contar registros automáticamente, pero empleado para estudios estadísticos tiene mucho potencial, así como para realizar consultas indexadas en tablas con miles de registros (que no es mi caso).
- Es importante recalcar que NO SE PUEDE borrar el contenido de una tabla temporal. En este caso, si por casualidad el energúmeno de turno introduce el DNI de una persona que no estaba realizando la guardia, la carga en el histórico y en principio el dato queda bloqueado. Para poder borrarlo requiere un ALTER TABLE para modificarle el campo SYSTEM_VERSIONING = OFF y borrar los registros introducidos de Fecha X a Y ... y luego volver a poner el campo SYSTEM_VERSIONING en ON para volver a darle estabilidad. Vamos, que no es práctico hacer cambios en ella. Puede dar mucha estabilidad y confianza en buenas manos... en malas manos es un quebradero de cabeza para el administrador.
- Las tablas temporales, son precisamente temporales. Eso indica que si se reinicia el servidor o se cierra sesión, se pierde toda la información de la tabla. En este caso, después de realizar todas las inserciones sí dejaría guardados cuántos días corresponden a cada uno, pero si reiniciamos y queremos cargar una guardia más... habría que volver a cargar todos los registros. En este caso concreto sería tan fácil como poner una columna adicional en la tabla DAOS que anote las guardias del valor asignado en @cuuenta, pero es algo que hay que tener en cuenta.
- Aunque se llama tabla temporal, ocupa recursos en el Sistema, en este caso Disco Duro.

Veracrypt:

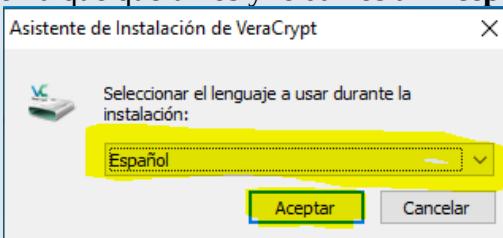
Veracrypt es un software de código abierto utilizado para cifrar archivos, carpetas, unidades y/o discos. Es una app multiplataforma y está basado en TrueCrypt (proyecto similar que cerró).

Características de Veracrypt:

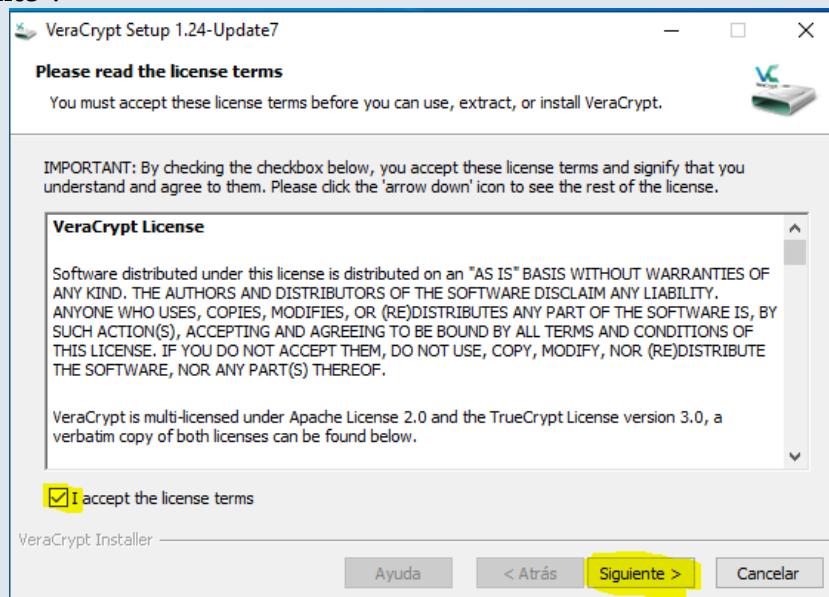
- ★ Creación de discos cifrados virtuales en un simple archivo
- ★ Cifrado de dispositivos de almacenamiento extraíble como USB, tarjetas SD e incluso discos duros.
- ★ Cifrado de cualquier partición de estos dispositivos de almacenamiento extraíble.
- ★ Cifrado de la partición o disco completo donde 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 (chantaje, extorsión etc.)

Como vemos, es una potente herramienta de seguridad. Primero vamos a instalarla. Para ello la descargamos desde [este](#) enlace. En mi caso descargué la [Versión 7 de Windows](#).

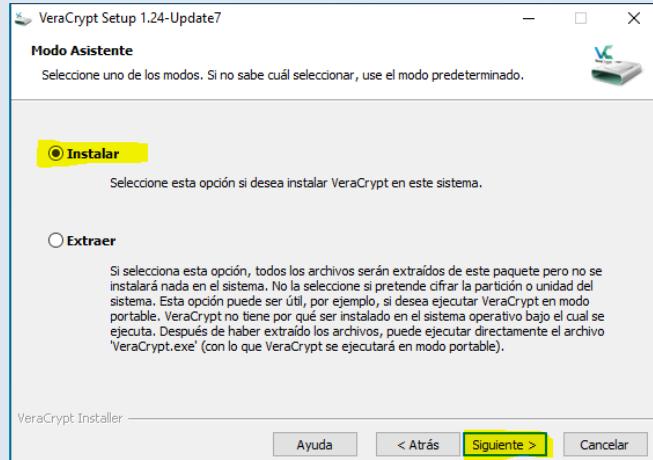
Una vez descargado, abrimos el ejecutable. Aparece un asistente que nos pregunta el idioma de instalación. Seleccionamos el idioma que queramos y le damos a “Aceptar”.



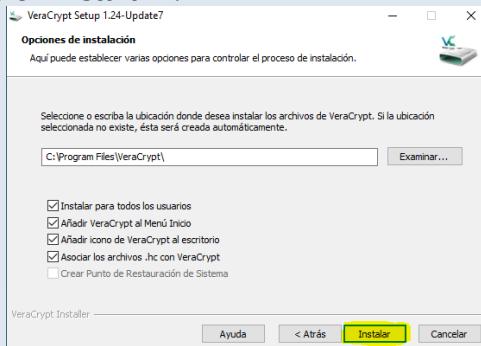
Nos sale que el EULA que como todo el mundo sabe es importante darle a Aceptar sin leerlo. Le damos a “Siguiente”.



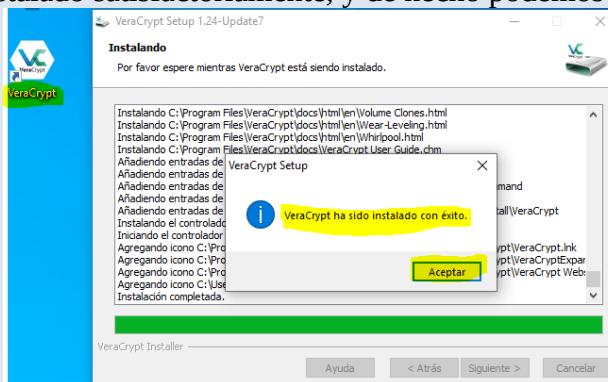
Ahora viene algo MUY importante. Nos pregunta si queremos instalarlo o si vamos a utilizarlo como Portable. Es importante recordar que si vamos a Cifrar los datos del Disco Duro donde tenemos la partición de arranque, tenemos que tenerlo instalado. En mi caso, al estar trabajando en un dominio y para evitar problemas absurdos en un futuro no voy a cifrar el contenido completo. Aún así voy a instalar la aplicación.



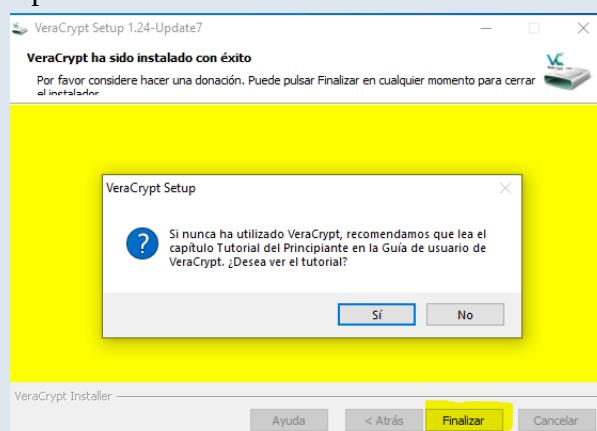
Nos pregunta en dónde queremos realizar la instalación, así como las opciones de menú de Inicio y demás. Lo dejamos por defecto e “**Instalar**”.



Nos indica que ha sido instalado satisfactoriamente, y de hecho podemos ver el Acceso directo.

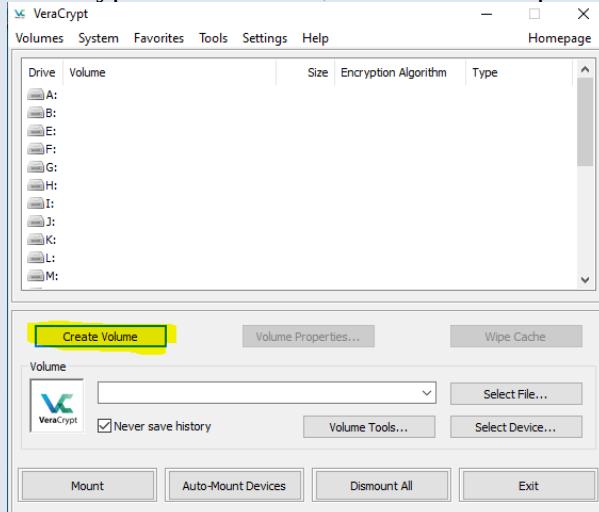


Después de instalarlo sale una ventana que nos invita a donar dinero. Al darle a Finalizar nos encienda a utilizar el tutorial si es la primera vez que lo utilizamos. Si estamos más perdidos que un piojo en una calva es muy importante leerse el tutorial.



Creación de un volumen de cifrado no oculto:

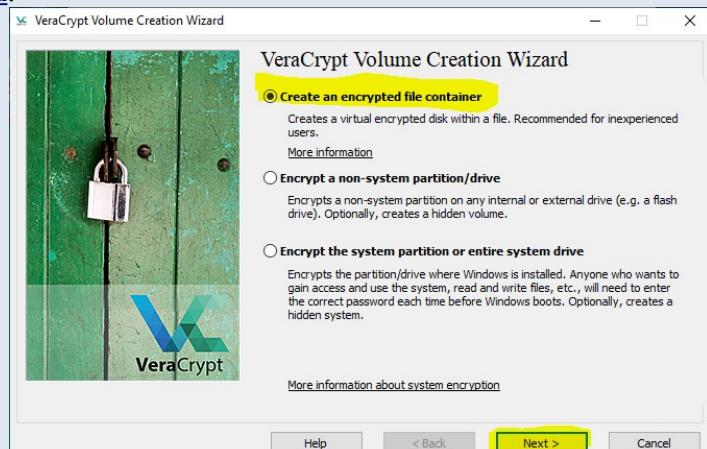
Antes de nada arrancamos Veracrypt. Al arrancarlo, lo veremos tal que así:



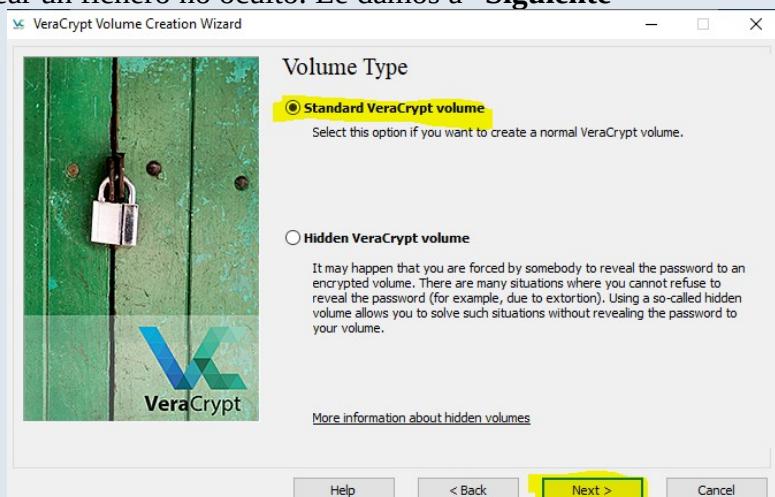
Le damos a “**Create Volume**” y nos abre un asistente que nos arroja 3 opciones:

- 1) Crear un contenedor de archivos cifrado
- 2) Encriptar una partición que no sea de arranque
- 3) Encriptar una partición de sistema o arranque

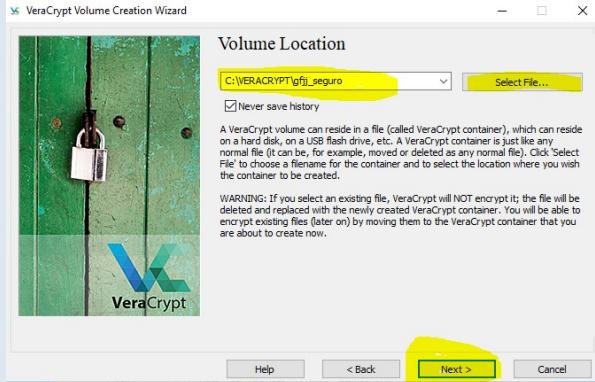
En nuestro caso, como solamente tenemos una partición y es la de arranque, y trabajamos en un Dominio, vamos a probar la opción más sencilla, que sería crear un contenedor de archivos cifrado, para evitar [liarla parda](#).



Ahora tenemos que elegir entre crear un fichero normal u oculto. Como no somos políticos ni banqueros, vamos a crear un fichero no oculto. Le damos a “**Siguiente**”



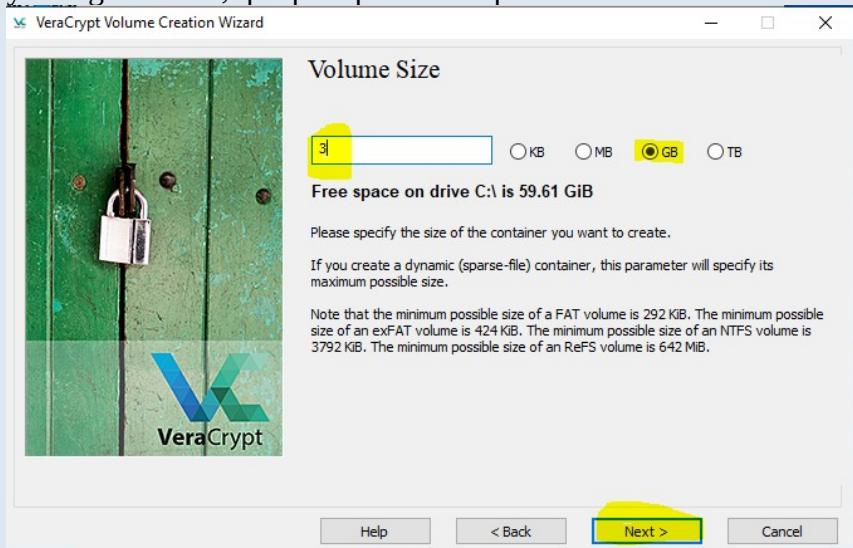
Ahora nos pregunta la ubicación del fichero. Es importante tener en cuenta dónde lo ponemos, porque si no lo recordamos y ciframos algo, podemos dejar de trabajar como Administradores.



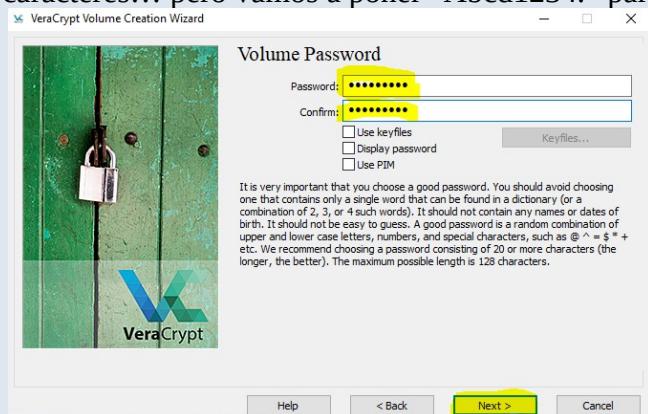
Nos pregunta el algoritmo de cifrado que queremos utilizar, así como el algoritmo HASH. Trae un montón, algunos de ellos que ni me suenan, así que voy a utilizar AES y SHA-512, que son “seguros” y habituales.



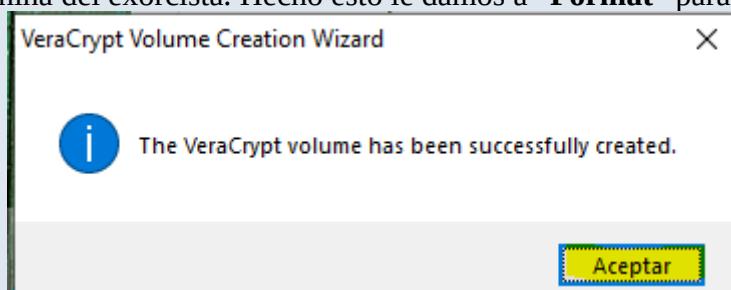
Nos pregunta cuánto espacio queremos designar al Contenedor. Ojo, es muy importante, estamos hablando de espacio dinámico, de modo que solamente va a consumir el que estemos utilizando, no lo reserva. Le voy a asignar 3 GB, que para probar me parece suficiente.



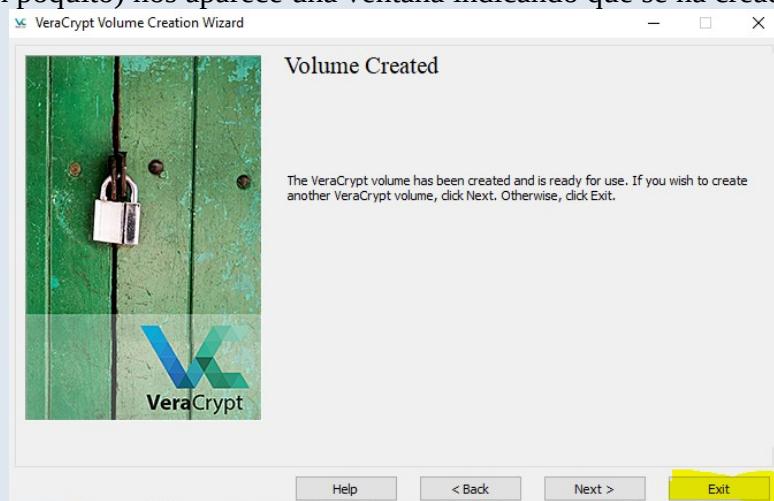
Hecho eso, nos pide que especifiquemos una contraseña. Nos da opciones muy interesantes (ficheros de claves, etc..) y nos recomienda poner una con mayúsculas, minúsculas, caracteres especiales y longitud superior a 20 caracteres... pero vamos a poner “Abcd1234.” para homogeneizar en clase.



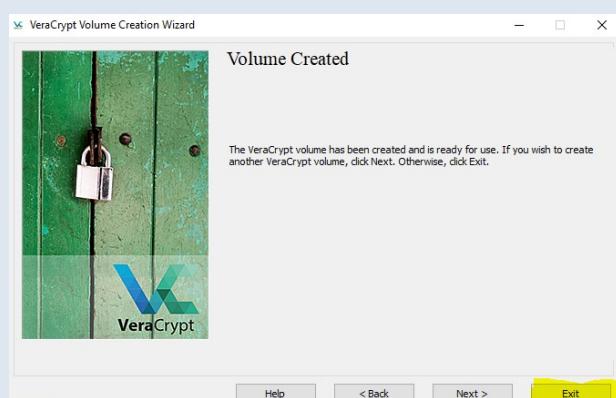
Seleccionamos el tipo de ficheros NTFS (de Windows) y movemos el ratón hacia todas direcciones como si fuéramos la niña del exorcista. Hecho esto le damos a “Format” para crear el contenedor.



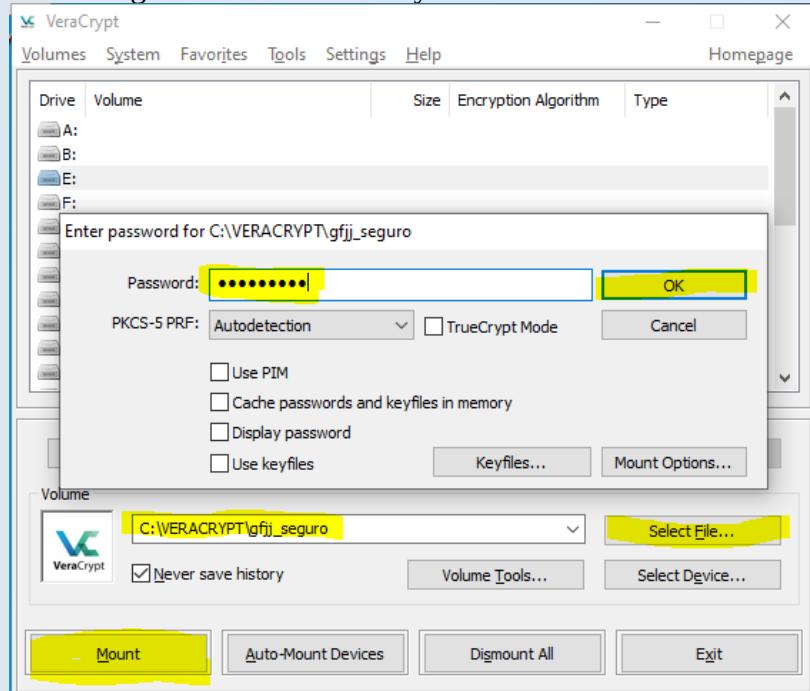
Pasado un tiempo (dependiendo de lo que estemos creando, seguramente para cifrar la partición completa tardará un poquito) nos aparece una ventana indicando que se ha creado correctamente.



Por último, nos permite crear más volúmenes o darle a Salir del asistente. En nuestro caso le damos a “Exit”

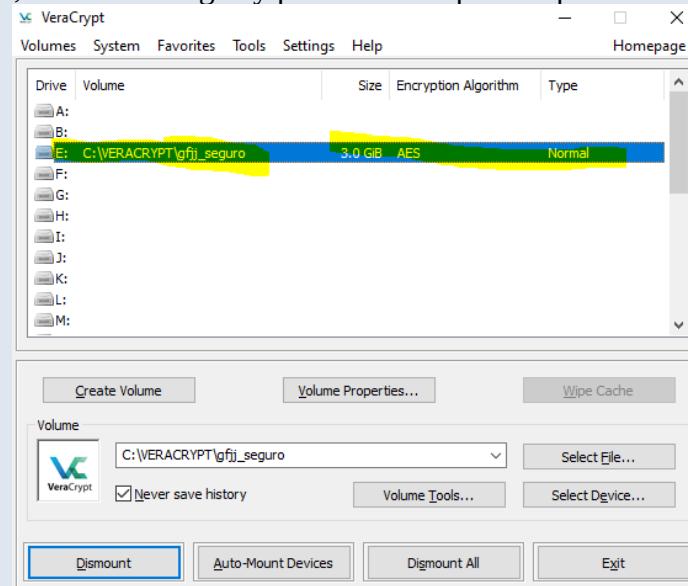


Para probarlo, vamos a escoger la ruta del fichero y darle a “Mount”.

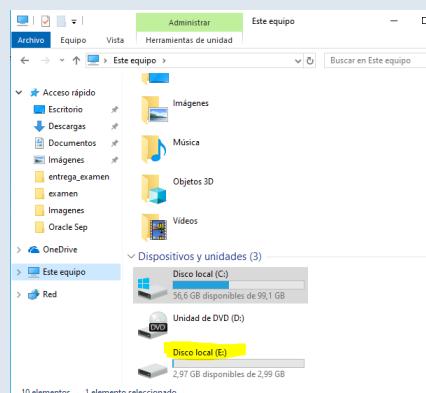


Nos abre una ventana en la que tenemos que introducir la contraseña y darle a “Ok”.

Pasados unos segundos, monta la Imagen y podemos comprobar que está creada correctamente.



Podemos comprobar que nos acaba de crear una Unidad Virtual (E) con la capacidad que hemos señalado, en este caso 3 GB.



Veracrypt es una potente herramienta a tener en cuenta para cifrar información, tanto el disco duro completo como para crear un contenedor cifrado en donde depositar información “segura”.

Encriptación de Columnas de BD:

Desde la Enfermería nos solicitan la creación de una BD más “segura”, en donde poder organizar entre otras cosas el listado de vacunación contra el COVID-19 del personal. Antes de nada, vamos a crear la BD y el usuario y contraseña para ellos:

```
USE MASTER
GO
CREATE LOGIN SANIDADlogin WITH PASSWORD='Abcd1234.'
GO
CREATE DATABASE SANIDAD_GFJJ
GO
USE SANIDAD_GFJJ
GO
CREATE USER SANIDAD FOR LOGIN SANIDADLogin
GO
CREATE TABLE VACUNACION
(DNI VARCHAR(20) PRIMARY KEY,
nombre varchar(100) NOT NULL,
apellidos varchar(100) NOT NULL,
fecha_ult DATE NOT NULL,
observaciones varchar(200) NOT NULL)
GO
GRANT SELECT, INSERT, UPDATE, DELETE ON VACUNACION TO SANIDAD
GO
```

Tenemos la BD, el usuario y la tabla. Lo supervisa la Capitán Encargada del servicio sanitario y nos interroga si podemos cifrar el contenido de las columnas nombre y apellidos, para que a simple vista no se pueda visualizar a quien corresponde. Como solamente nos pide cifrar esos campos, vamos a crear una clave simétrica que nos permita realizarlo.

```
CREATE SYMMETRIC KEY Sanitario_Key
AUTHORIZATION SANIDAD
WITH ALGORITHM=AES_256
ENCRYPTION BY PASSWORD='Abcd1234.'
GO
EXECUTE AS USER='SANIDAD'
GO
```

Tenemos creada la clave simétrica, vamos a abrirla utilizando la siguiente instrucción:

```
OPEN SYMMETRIC KEY [Sanitario_Key] DECRYPTION BY PASSWORD='Abcd1234.'
GO
```

Para comprobar que funciona correctamente, vamos a introducir datos en la tabla.

```
INSERT INTO VACUNACION VALUES ('32697849Z',EncryptByKey(Key_GUID('Sanitario_Key'), 'Juan
Jose'),EncryptByKey(Key_GUID('Sanitario_Key'), 'Gomez Fernandez'), '2021-04-04', 'AstraZeneca')
)
INSERT INTO VACUNACION VALUES ('36434580I',EncryptB-
yKey(Key_GUID('Sanitario_Key'), 'Adrian'),EncryptByKey(Key_GUID('Sanitario_Key'), 'Iglesias
Cid'), '2021-04-04', 'No se vacuna')
)
INSERT INTO VACUNACION VALUES ('32950257E',EncryptByKey(Key_GUID('Sanitario_Key'), 'Os-
car'),EncryptByKey(Key_GUID('Sanitario_Key'), 'Liao Mera'), '2021-01-08', 'Pfizer'
)
GO
```

Comprobamos que funciona correctamente.

```
SELECT * FROM VACUNACION
GO
```

	DNI	nombre	apellidos	fecha_ult	observaciones
1	32697849Z			2021-04-04	AstraZeneca
2	32950257E			2021-01-08	Pfizer
3	36434580I			2021-04-04	No se vacuna

Podemos comprobar que no aparece nada en la columna nombre ni apellidos, está “en blanco”. Ahora vamos a probar a hacer la consulta desencriptando los campos a través de la función CONVERT.

```
SELECT DNI, CONVERT(VARCHAR, DecryptByKey(nombre)) + ' ' + CONVERT(VARCHAR, DecryptByKey(apellidos)) AS 'Nombre Completo', observaciones
FROM VACUNACION
GO
```

	DNI	Nombre Completo	observaciones
1	32697849Z	Juan Jose Gomez Fernandez	AstraZeneca
2	32950257E	Oscar Liao Mera	Pfizer
3	36434580I	Adrian Iglesias Cid	No se vacuna

Como podemos observar, si desencriptamos nos muestra el contenido perfectamente. Ahora vamos a cerrar todas las claves simétricas.

```
CLOSE ALL SYMMETRIC KEYS
GO
```

	DNI	Nombre Completo	observaciones
1	32697849Z	NULL	AstraZeneca
2	32950257E	NULL	Pfizer
3	36434580I	NULL	No se vacuna

Como podemos comprobar, al tener cerradas las claves simétricas no se vería el contenido (aparece vacío en este caso). Es fundamental tener en cuenta que con este tipo de cifrado, dependemos de:

1. Asignar el cifrado/descifrado por permisos solamente a los usuarios que lo necesiten.
2. Fundamental que el usuario se acostumbre a abrir las claves cuando lo requiera y cerrarlas cuando no las esté utilizando, para mayor seguridad.

Encriptación de Backups de BD:

Nos comunican en la Oficina de Personal que recientemente uno de sus ~~manazas~~-usuarios, manipulando la BD quiso copiar el contenido, lo cortó y no disponen de copia de seguridad. Nos solicitan a nosotros una copia del contenido... que no tenemos, puesto que la habíamos dejado en manos de los usuarios. Para evitar que ocurra, el suboficial encargado nos ordena realizar una BD adicional en donde puedan administrar los backups. Como contienen información sensible, nos piden que encriptemos dichas copias de seguridad. Antes de nada, vamos a realizar la BD indicada:

```
USE MASTER
GO
DROP DATABASE IF EXISTS Backups_GFJJ
GO
CREATE DATABASE Backups_GFJJ;
GO
USE Backups_GFJJ;
GO

DROP TABLE IF EXISTS ORDEN
GO
CREATE TABLE ORDEN (
    ID int IDENTITY(1,1000) PRIMARY KEY,
    num int
);
GO
```

Ahora lo que hacemos es introducirle los registros mediante un Procedimiento Almacenado, para poder introducir una buena cantidad para testear.

```
CREATE OR ALTER PROCEDURE AutoAlta
AS
DECLARE @i int = 1
WHILE @i <201
    BEGIN
        INSERT ORDEN (num) VALUES (@i)
        Set @i +=1
    END
GO

EXECUTE AutoAlta;
GO

SELECT * FROM ORDEN;
GO
```

Tenemos creados 200 registros. Sería con Órdenes diarias, cada una con su contenido. Ahora creamos la clave maestra.

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

Creamos el certificado:

```
DROP CERTIFICATE Back_GFJJ
GO
CREATE CERTIFICATE Back_GFJJ
    WITH SUBJECT = 'Back_GFJJ_cert Certificado de Backups';
GO
```

Vamos a exportar el certificado a un fichero. No es obligatorio, pero es altamente recomendable porque si perdemos el certificado del backup sería muy complicado recuperar la información.

```

BACKUP CERTIFICATE Back_GFJJ
TO FILE = 'c:\SQLBackups\Back_GFJJ.cert'
WITH PRIVATE KEY (
    FILE = 'c:\SQLBackups\Back_GFJJ.key',
    ENCRYPTION BY PASSWORD = 'Abcd1234.')
GO

```

Ahora hacemos el backup cifrado de la BD, por lo que pueda pasar:

```

BACKUP DATABASE Backups_GFJJ
TO DISK = 'C:\SQLBackups\Backup_Enc_GFJJ.bak'
WITH
ENCRYPTION
(
ALGORITHM = AES_256,
SERVER CERTIFICATE = Back_GFJJ
)
GO

```

The screenshot shows a T-SQL query window with the following command:

```

BACKUP DATABASE Backups_GFJJ
TO DISK = 'C:\SQLBackups\Backup_Enc_GFJJ.bak'
WITH
ENCRYPTION
(
ALGORITHM = AES_256,
SERVER CERTIFICATE = Back_GFJJ
)
GO

```

Below the command, the 'Messages' pane displays the execution results:

```

100 % < 
Messages
Processed 400 pages for database 'Backups_GFJJ', file 'Backups_GFJJ' on file 1.
Processed 2 pages for database 'Backups_GFJJ', file 'Backups_GFJJ_log' on file 1.
BACKUP DATABASE successfully processed 402 pages in 0.204 seconds (15.359 MB/sec).

```

Ahora viene el lío. Vamos a cargarnos la BD así como el certificado, para ver qué pasaría si alguien obtiene el backup (pero no dispone del certificado):

```

DROP DATABASE Backups_GFJJ;
GO

```

Luego nos cargamos el certificado:

```

DROP CERTIFICATE Back_GFJJ
GO

```

Intentamos restaurar la BD y nos da error.

The screenshot shows a T-SQL query window with the following command:

```

RESTORE DATABASE Backups_GFJJ
FROM DISK = 'c:\SQLBackups\Backup_Enc_GFJJ.bak'
WITH RECOVERY,
REPLACE, STATS = 10;
GO

```

Below the command, the 'Messages' pane displays the error message:

```

100 % < 
Messages
Msg 33111, Level 16, State 3, Line 5
Cannot find server certificate with thumbprint '0x6103E830B0D67D292FAED72811C554FE97658357'.
Msg 3013, Level 16, State 1, Line 5
RESTORE DATABASE is terminating abnormally.

```

Llegados a este punto, si no recuperamos el certificado, toda la información del Backup es irrecuperable. Por suerte, hemos creado una copia del certificado en un fichero. Podemos restaurar el certificado:

```

CREATE CERTIFICATE SQL_encriptar_BBMDBCert
FROM FILE = 'c:\SQLBackups\Back_GFJJ.cert'
WITH PRIVATE KEY (FILE = 'c:\SQLBackups\Back_GFJJ.key',
DECRYPTION BY PASSWORD = 'Abcd1234.');
GO

```

Podemos restaurar la BD sin problema ninguno:

```
RESTORE DATABASE Backups_GFJJ
FROM DISK = 'c:\SQLBackups\Backup_Enc_GFJJ.bak'
WITH RECOVERY,
REPLACE, STATS = 10;
GO
```

The screenshot shows a SQL Server Management Studio window. In the top pane, a T-SQL script is displayed:

```
RESTORE DATABASE Backups_GFJJ
FROM DISK = 'c:\SQLBackups\Backup_Enc_GFJJ.bak'
WITH RECOVERY,
REPLACE, STATS = 10;
GO
```

In the bottom pane, the 'Messages' tab is selected, showing the execution progress and results:

100 % ▶

Messages

```
81 percent processed.
91 percent processed.
100 percent processed.
Processed 400 pages for database 'Backups_GFJJ', file 'Backups_GFJJ' on file 1.
Processed 2 pages for database 'Backups_GFJJ', file 'Backups_GFJJ_log' on file 1.
RESTORE DATABASE successfully processed 402 pages in 0.113 seconds (27.728 MB/sec).
```

Al igual que ocurriría al descargar una moneda virtual, es FUNDAMENTAL garantizar tanto la seguridad del certificado, como la accesibilidad. Ya que si guardamos el certificado y luego no lo encontramos, no tenemos forma alguna de acceder a la información.

Por último, vamos a eliminar el certificado y la MASTER KEY para seguir realizando pruebas.

```
USE MASTER
GO
DROP CERTIFICATE Back_GFJJ
GO
DROP MASTER KEY;
DROP DATABASE Backups_GFJJ;
GO
```

Encriptación de BD de TDE:

La sección de Habilitación maneja datos confidenciales (nóminas, datos bancarios, etc..) del personal de la Unidad, de modo que solicitan la creación de una BD que les permita manejar y gestionar esos datos respetando la confidencialidad que entrañan. Para eso, el Alto Mando nos encomienda a gestionarlo a través de encriptación mediante Cifrado Transparente de Datos (TDE).

Primero vamos a crear la clave maestra y el certificado:

```
USE MASTER
```

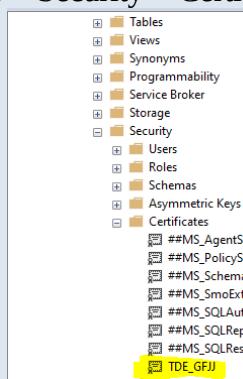
```
GO
```

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

```
CREATE CERTIFICATE TDE_GFJJ
    WITH SUBJECT = 'TDE Certificado';
GO
```

Desde SSMS podemos comprobar el certificado, a través de:

Databases > System Databases > master > Security > Certificates > *Nombredelcertificado*



Eso para localizarlo en entorno gráfico. Por consola sería:

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

Como hemos aprendido durante la creación del certificado de Backups, hay que hacer copia de seguridad de los certificados (por lo que pueda pasar). Obviamente habría que guardarlos en un dispositivo distinto al servidor, por si hiciera falta. Procedemos a hacer el backup:

```
BACKUP CERTIFICATE TDE_GFJJ
TO FILE = 'c:\certificados\TDE_GFJJ.cert'
WITH PRIVATE KEY (
    FILE = 'c:\certificados\TDE_GFJJ.key',
    ENCRYPTION BY PASSWORD = 'Abcd1234.')
GO
```

Ahora creamos la BD nueva con la que vamos a trabajar.

```
DROP DATABASE IF EXISTS HABILITACION_GFJJ
GO
CREATE DATABASE HABILITACION_GFJJ
GO
```

Accedemos a la BD y creamos el certificado.

```
USE HABILITACION_GFJJ;
CREATE DATABASE ENCRYPTION KEY
    WITH ALGORITHM = AES_256
    ENCRYPTION BY SERVER CERTIFICATE TDE_GFJJ;
GO
```

Ahora habilitamos la encriptación. Para ello, primero tenemos que salir de la BD, volvemos a Master y la activamos:

```
USE [master];
ALTER DATABASE HABILITACION_GFJJ SET ENCRYPTION ON;
GO
```

Comprobamos el estado de la BD con la siguiente instrucción:

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

Nos muestra el siguiente estado:

The screenshot shows a SQL Server Management Studio window with the following content:

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

The results pane displays a table with two rows:

	HABILITACION_GFJJ	encryption_state
1	tempdb	3
2	HABILITACION_GFJJ	3

Vemos que figura en estado 3. Vamos a consultar la teoría para ver qué significa:

0. Ninguna clave de cifrado de la base de datos, sin cifrado
1. Sin cifrar
2. Cifrado en curso
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)

Siguiendo la teoría y viendo que está en Estado 3, significa que está cifrado. Vamos a crear una tabla e introducirle datos:

```
USE HABILITACION_GFJJ;
CREATE TABLE NOMINAS
(DNI VARCHAR(20) PRIMARY KEY,
nombre varchar(100) NOT NULL,
apellidos varchar(100) NOT NULL,
nomina INT NOT NULL,
cuenta varchar(200) NOT NULL)
GO

INSERT INTO NOMINAS VALUES ('32697849Z', 'Juan Jose', 'Gomez Fernandez', 1160, 'ES3635567373')
INSERT INTO NOMINAS VALUES ('33755843F', 'Andrea', 'Gallo Santabaya', 1210, 'ES12241649876')
INSERT INTO NOMINAS VALUES ('36594003W', 'Francisco Javier', 'Garcia Ramirez', 1322, 'ES43878957759')
INSERT INTO NOMINAS VALUES ('32156599N', 'Edmilton', 'Wilson Fernandes', 1087, 'ES302158352')
GO
```

Ahora vamos a hacer copia de seguridad de la BD, por lo que pueda pasar:

```
USE MASTER;
BACKUP DATABASE HABILITACION_GFJJ
TO DISK = 'C:\backup\TDE_GFJJ.bak';
GO
```

```
Processed 344 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ' on file 1.  
Processed 5 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ_log' on file 1.  
BACKUP DATABASE successfully processed 349 pages in 0.197 seconds (13.837 MB/sec).
```

Ahora vamos a hacer una copia de seguridad del Log.

```
BACKUP LOG HABILITACION_GFJJ  
TO DISK = 'C:\backup\HABILITACION_GFJJ_log.bak'  
With NORECOVERY  
GO
```

```
Processed 7 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ_log' on file 1.  
BACKUP LOG successfully processed 7 pages in 0.024 seconds (2.115 MB/sec).
```

Ahora desconectamos la instancia de SSMS y volvemos a abrirla. Vamos a intentar cargar el Backup en la segunda instancia.

```
USE MASTER;  
RESTORE DATABASE HABILITACION_GFJJ  
FROM DISK = 'C:\backup\TDE_GFJJ.bak'  
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',  
MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';  
GO
```

```
Processed 344 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ' on file 1.  
Processed 5 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ_log' on file 1.  
RESTORE DATABASE successfully processed 349 pages in 0.106 seconds (25.717 MB/sec).
```

Vemos que nos deja perfectamente. Nos deja porque disponemos del certificado y de la clave maestra. Ahora vamos a cargarnos el certificado y probamos de nuevo:

```
USE MASTER;  
DROP DATABASE HABILITACION_GFJJ;  
DROP CERTIFICATE TDE_GFJJ;  
RESTORE DATABASE HABILITACION_GFJJ  
FROM DISK = 'C:\backup\TDE_GFJJ.bak'  
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',  
MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';  
GO
```

Nos arroja el error esperado de que no disponemos del certificado.

```
RESTORE DATABASE HABILITACION_GFJJ  
FROM DISK = 'C:\backup\TDE_GFJJ.bak'  
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',  
MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';  
GO
```

0 %

Messages

```
Msg 33111, Level 16, State 3, Line 4  
Cannot find server certificate with thumbprint '0xF0E3AC6D070D1C71AD429089F377397766D676C9'.  
Msg 3013, Level 16, State 1, Line 4  
RESTORE DATABASE is terminating abnormally.
```

Como ya nos ocurría en su momento con la copia del Backup, al no disponer del certificado no nos permite cargar la copia de seguridad. Vamos a volver a cargar el certificado:

```
DROP CERTIFICATE TDE_GFJJ;  
CREATE CERTIFICATE TDE_GFJJ  
FROM FILE = 'c:\certificados\TDE_GFJJ.cert'  
WITH PRIVATE KEY (  
FILE = 'c:\certificados\TDE_GFJJ.key',  
DECRYPTION BY PASSWORD = 'Abcd1234.')  
GO
```

```

RESTORE DATABASE HABILITACION_GFJJ
FROM DISK = 'C:\backup\TDE_GFJJ.bak'
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',
      MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';
GO

```

0 % ▶

Messages

Processed 344 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ' on file 1.
 Processed 5 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ_log' on file 1.
 RESTORE DATABASE successfully processed 349 pages in 0.115 seconds (23.704 MB/sec).

Ahora vamos a intentar cargar la copia utilizando una clave maestra falsa. Nos cargamos todo y creamos la clave distinta a la anterior.

```

USE MASTER;
DROP DATABASE HABILITACION_GFJJ;
DROP CERTIFICATE TDE_GFJJ;
DROP MASTER KEY;
CREATE MASTER KEY
  ENCRYPTION BY PASSWORD = 'Pirata.2021';
GO

```

Ahora creamos el certificado, con nombre similar al del backup.

```

CREATE CERTIFICATE TDE_GFJJ
  WITH SUBJECT = 'TDE Certificado';
GO

```

Intentamos cargar la BD y nos manifiesta el mismo error que antes, pero sabemos que es porque la clave maestra no corresponde.

```

RESTORE DATABASE HABILITACION_GFJJ
FROM DISK = 'C:\backup\TDE_GFJJ.bak'
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',
      MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';
GO

```

```

DROP MASTER KEY;
CREATE MASTER KEY
  ENCRYPTION BY PASSWORD = 'Pirata.2021';
GO
CREATE CERTIFICATE TDE_GFJJ
  WITH SUBJECT = 'TDE Certificado';
GO
RESTORE DATABASE HABILITACION_GFJJ
FROM DISK = 'C:\backup\TDE_GFJJ.bak'
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',
      MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';
GO

```

0 % ▶

Messages

Msg 33111, Level 16, State 3, Line 11
 Cannot find server certificate with thumbprint '0xF0E3AC6D070D1C71AD429089F377397766D676C9'.
 Msg 3013, Level 16, State 1, Line 11
 RESTORE DATABASE is terminating abnormally.

Esta vez vamos a cargar la clave maestra inicial, así como cargar los certificados y la BD desde el backup.

```

DROP CERTIFICATE TDE_GFJJ;
DROP MASTER KEY;
CREATE MASTER KEY
  ENCRYPTION BY PASSWORD = 'Abcd1234.';
GO
CREATE CERTIFICATE TDE_GFJJ
  FROM FILE = 'c:\certificados\TDE_GFJJ.cert'
  WITH PRIVATE KEY (
    FILE = 'c:\certificados\TDE_GFJJ.key',
    DECRYPTION BY PASSWORD = 'Abcd1234.')
GO
RESTORE DATABASE HABILITACION_GFJJ

```

```
FROM DISK = 'C:\backup\TDE_GFJJ.bak'
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',
      MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';
GO
```

The screenshot shows a SQL Server Management Studio window. In the top pane, there is a code editor containing T-SQL commands to restore a database from a backup file. The commands are:

```
RESTORE DATABASE HABILITACION_GFJJ
FROM DISK = 'C:\backup\TDE_GFJJ.bak'
WITH MOVE 'HABILITACION_GFJJ' TO 'C:\data\HABILITACION_GFJJ_2ndServer.mdf',
      MOVE 'HABILITACION_GFJJ_log' TO 'C:\data\HABILITACION_GFJJ_2ndServer_log.mdf';
GO
```

In the bottom pane, under the 'Messages' tab, the results of the restore operation are displayed. The output shows:

```
Processed 344 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ' on file 1.
Processed 5 pages for database 'HABILITACION_GFJJ', file 'HABILITACION_GFJJ_log' on file 1.
RESTORE DATABASE successfully processed 349 pages in 0.108 seconds (25.241 MB/sec).
```

Tenemos que tener en cuenta que nos arroja mayor seguridad, ya que si nos falta o bien el certificado o bien la clave maestra no podemos cargar la copia de seguridad, de modo que si transportamos por dos vías/unidades diferentes el certificado y la clave es muy complicado que vulneren nuestra seguridad.

Por contra, como Administradores tenemos que estar más atentos tanto a la hora de volcar las copias/certificados/claves (si no coinciden estamos perdidos) como a la hora de resguardarlas, ya que si necesitamos recurrir a ellas necesitamos de ambas.

Dynamic Data Masking:

Se ha filtrado información confidencial de la Unidad y ha salido en la prensa. Se está realizando el trabajo de investigación para vislumbrar quien y cuando filtró la información, pero mientras tanto el Alto Mando quiere atajar el problema para evitar que se repita. Como en toda empresa piramidal, los principales sospechosos forman la base de la pirámide, que a su vez son los encargados de acometer las tareas rutinarias. Nos ordenan enmascarar las columnas de las tablas en función al usuario, de modo que los usuarios privilegiados (suboficiales y oficiales) puedan consultar todas las columnas, pero no así los usuarios comunes (tropa). Para ello vamos a utilizar el enmascaramiento de datos dinámico (DDM).

Vamos a empezar creando la tabla:

```
DROP DATABASE If Exists Personal_GFJJ
GO
CREATE DATABASE Personal_GFJJ
GO
USE Personal_GFJJ
GO
CREATE SCHEMA Perso;
DROP TABLE Perso.LISTADO;
CREATE TABLE Perso.LISTADO (
    CodVacante      int NOT NULL CONSTRAINT PKLISTADO PRIMARY KEY,
    Nombre          nvarchar(20) NULL,
    Apellido         nvarchar(20) NULL,
    ISFAS           varchar(20) NOT NULL,
    Situacion        varchar(20) CONSTRAINT DFLTSituacion DEFAULT ('Activo')
                           CONSTRAINT CHKSituacion
                           CHECK (Situacion in ('Activo', 'Licencia', 'RED')),
    Email            nvarchar(100) NULL,
    Alta_Cuartel     date NOT NULL);
GO

INSERT INTO Perso.LISTADO (CodVacante, Nombre, Apellido, ISFAS, Email, Alta_Cuartel)
VALUES(1, 'Juan Jose', 'Gomez Fernandez', '682412560', 'awjuanjose@gmail.com', '2004-04-24'),
      (2, 'Ernesto', 'Grial Facundo', '682954456', 'jcerlf2@mdef.com', '2021-04-12'),
      (3, 'Lucia', 'Garrote Lopez', '682120385', 'fjvywq2@mdef.com', '1/1/1959');
GO
```

Creamos los usuarios y le damos permisos en la tabla. Tambien le vamos a dar permisos al usuario de suboficiales para ver lo que está enmascarado.

```
CREATE USER tropa WITHOUT LOGIN;
CREATE USER subof WITHOUT LOGIN;
GO

GRANT SELECT ON Perso.Listado TO tropa;
GRANT SELECT ON Perso.Listado TO subof;
GRANT UNMASK TO subof;
GO
```

Tenemos los usuarios y la tabla. Ahora vamos a estudiar qué tipo de enmascaramiento nos interesa más para cada caso.

DEFAULT

El primer tipo de enmascaramiento según los manuales es del tipo Default. Las columnas encriptadas con Default, muestran el campo en blanco (como nos pasaba con la encriptación de columnas de BD). Esto nos resulta interesante para el campo email, en mi opinión debido a que no se debe de mostrar esa información a todo el mundo. Alteramos la tabla:

```
ALTER TABLE Perso.LISTADO
ALTER COLUMN Email varchar(200) MASKED WITH (FUNCTION = 'default()');
GO
```

Lo comprobamos con el usuario de tropa:

```
EXECUTE AS USER='tropa';
```

```
GO
```

```
SELECT *
FROM Perso.LISTADO;
GO
```

CodVacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	Juan Jose	Gomez Fernandez	682412560	Activo	xxxx	2004-04-24
2	Ernesto	Grial Facundo	682954456	Activo	xxxx	2021-04-12
3	Lucia	Garrote Lopez	682120385	Activo	xxxx	1959-01-01

Ahora deshacemos y comprobamos con el usuario de suboficiales que sí lo puede ver correctamente.

```
REVERT
```

```
GO
```

```
EXECUTE AS USER='subof';
```

```
GO
```

```
SELECT *
```

```
FROM Perso.LISTADO;
```

```
GO
```

```
REVERT
```

```
GO
```

CodVacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	Juan Jose	Gomez Fernandez	682412560	Activo	awjuanjose@gmail.com	2004-04-24
2	Ernesto	Grial Facundo	682954456	Activo	jcerff2@mdef.com	2021-04-12
3	Lucia	Garrote Lopez	682120385	Activo	fjyywq2@mdef.com	1959-01-01

Como vemos, ahora mismo los suboficiales sí verían el email pero la tropa no podría verlo en principio ni tendrían algún indicio de cual es, si les hiciera/hiciese falta podría facilitarlo el suboficial. Vamos a comprobar el estado de enmascaramiento de cada columna, mediante un procedimiento almacenado.

```
CREATE OR ALTER PROC ElZorro
```

```
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 ElZorro  
GO
```

name	table_name	is_masked	masking_function	
1	Email	LISTADO	1	default()

Nos indica que la columna Email tiene el enmascaramiento default(). Eso nos guía mayormente para distinguir si está o no enmascarada. En la de tipo default es raro que pongamos XXX a todo (al menos en tablas de trabajo), pero en el enmascaramiento aleatorio sí que puede dar lugar a confusiones.

PARTIAL

El segundo tipo de cifrado que podemos utilizar es el de tipo Partial. Este cifrado nos permite enmascarar todos los caracteres salvo los X que especifiquemos, de tal modo que por ejemplo podemos mostrar solamente los 3 primeros, los últimos 5, etc. Se utiliza por ejemplo con el banco en tiendas online, ves los últimos números de cuenta y a veces la ENTIDAD (dos primeras) pero nada más. Así no se ve cual es tu número pero tú si dispones de varios números de cuenta puedes saber si es uno u otro.

Vamos a enmascarar parcialmente la columna ISFAS, para que no se vea completamente pero sí se pueda percibir los últimos 4 dígitos, que los utilizan en Personal para registrar el Portal Personal de cada uno.

```
ALTER TABLE Perso.LISTADO  
ALTER COLUMN ISFAS ADD MASKED WITH (FUNCTION = 'partial(0,"XXXX",4)')  
GO
```

Lo comprobamos con el usuario de tropa. Solamente debería mostrarnos los 4 últimos:

```
EXECUTE AS USER='tropa';  
GO
```

```
SELECT *  
FROM Perso.LISTADO;  
GO
```

CodVacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	Juan Jose	Gomez Fernandez	XXXX2560	Activo	xxxx	2004-04-24
2	Emesto	Grial Facundo	XXXX4456	Activo	xxxx	2021-04-12
3	Lucia	Garrote Lopez	XXXX0385	Activo	xxxx	1959-01-01

Comprobamos con el usuario de suboficiales que podemos verlo correctamente:

```

EXECUTE AS USER='subof';
GO
SELECT *
FROM Perso.LISTADO;
GO

```

```

REVERT
GO

```

Cod/vacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	Juan Jose	Gomez Fernandez	682412560	Activo	awjuanjose@gmail.com	2004-04-24
2	Emesto	Grial Facundo	682954456	Activo	jcerif2@mdef.com	2021-04-12
3	Lucia	Garrote Lopez	682120385	Activo	fjivwq2@mdef.com	1959-01-01

Como vemos, los suboficiales sí pueden ver el número completo. Vamos a comprobar el estado de enmascaramiento de la tabla:

name	table_name	is_masked	masking_function
ISFAS	LISTADO	1	partial(0, "XXXXX", 4)
Email	LISTADO	1	default()

Nos arroja tanto el enmascaramiento de tipo parcial como la condición (los 4 últimos a la derecha, ninguno a la izquierda)

RANDOM

El enmascaramiento aleatorio es muy interesante en las columnas de tipo numérico, puesto que nos muestra un valor numérico aleatorio de entre un rango. En este tipo de columnas es incluso mejor que otro tipo de cifrados, puesto que no estás mostrando la información real si no la falsa, de modo que si una persona no autorizada accede al contenido, está llevándose información falsa pensando que está correcto. Es interesante si tenemos que registrar por ejemplo lo que cobra cada empleado. En este caso, queremos enmascarar el código de vacante, que va asociado a un puesto, de modo que la tropa no pueda saber si la persona X que tiene el código de vacante Y está realmente en el puesto que le corresponde por vacante, para evitar reclamaciones. Procedemos con el cifrado:

```

ALTER TABLE Perso.LISTADO
ALTER COLUMN CodVacante ADD MASKED WITH (FUNCTION = 'random(1, 140)')
GO

```

Ahora comprobamos con el usuario tropa que efectivamente asigna valores aleatorios. Tenían valores 1, 2 y 3.

```

EXECUTE AS USER='tropa';
GO

```

```

SELECT *
FROM Perso.LISTADO;
GO

```

```

EXECUTE AS USER='tropa';
GO

SELECT *
FROM Perso.LISTADO;
GO

```

100 %

	CodVacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	75	Juan Jose	Gomez Fernandez	XXXXX2560	Activo	xxxx	2004-04-24
2	35	Ernesto	Grial Facundo	XXXXX4456	Activo	xxxx	2021-04-12
3	83	Lucia	Garote Lopez	XXXXX0385	Activo	xxxx	1959-01-01

Podemos observar que nos está dando distinto código de vacante. Este tipo de cifrado es interesante cara a la protección... pero hay que tener mucho cuidado. Si no somos conscientes de que tenemos el cifrado puesto, podemos hacer una desgracia. Comprobamos con el usuario de suboficiales que lo ven correctamente.

```

EXECUTE AS USER='subof';
GO

SELECT *
FROM Perso.LISTADO;
GO

```

100 %

	CodVacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	1	Juan Jose	Gomez Fernandez	682412560	Activo	awjuanjose@gmail.com	2004-04-24
2	2	Ernesto	Grial Facundo	682954456	Activo	jcerf2@mdef.com	2021-04-12
3	3	Lucia	Garote Lopez	682120385	Activo	fjvywq2@mdef.com	1959-01-01

El enmascaramiento de las columnas queda así:

	name	table_name	is_masked	masking_function
1	CodVacante	LISTADO	1	random(1, 140)
2	ISFAS	LISTADO	1	partial(0, "XXXXX", 4)
3	Email	LISTADO	1	default()

EMAIL

Tras mostrarle los progresos al Capitán, estima que es interesante que el usuario de tropa pueda ver los correos, de modo que nos ordena retirarle la encriptación.

```

ALTER TABLE Perso.LISTADO
ALTER COLUMN Email DROP MASKED;
GO

```

```

EXEC ElZorro
GO

```

```

EXEC ElZorro
GO

```

100 %

	name	table_name	is_masked	masking_function
1	CodVacante	LISTADO	1	random(1, 140)
2	ISFAS	LISTADO	1	partial(0, "XXXXX", 4)

Podemos comprobar que la columna de email deja de estar cifrada.

Al día siguiente a haber realizado los cambios, el Mando nos informa de que sí requiere cifrar la columna email, puesto que ya se confeccionó el listado con todos los correos corporativos.

```
SELECT nombre, apellido, email INTO Perso.EMAIL FROM Perso.LISTADO;
```

```
ALTER TABLE Perso.LISTADO  
ALTER COLUMN email ADD MASKED WITH (FUNCTION = 'email()')  
GO
```

```
EXECUTE AS USER='tropa';  
GO
```

```
SELECT *  
FROM Perso.LISTADO;  
GO
```

```
REVERT  
GO
```

	CodVacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	36	Juan Jose	Gomez Fernandez	XXXXX2560	Activo	aXXX@XXXX.com	2004-04-24
2	51	Emesto	Grial Facundo	XXXXX4456	Activo	jXXX@XXXX.com	2021-04-12
3	133	Lucia	Garote Lopez	XXXXX0385	Activo	fXXX@XXXX.com	1959-01-01

Esta vez hemos utilizado la función email, que lo que hace es enmascarar el correo electrónico mostrando solamente la primera letra del email. Tiene la ventaja de que el que disponga de 2-3 correos suelen empezar por letra distinta, pero fuera de eso es muy similar a emplear la de por defecto.

Por último, se han filtrado datos nuevamente y el Mando ordena retirar todos los permisos a todos los usuarios hasta que se aclare.

	CodVacante	Nombre	Apellido	ISFAS	Situacion	Email	Alta_Cuartel
1	5	Juan Jose	Gomez Fernandez	XXXXX2560	Activo	aXXX@XXXX.com	2004-04-24
2	74	Emesto	Grial Facundo	XXXXX4456	Activo	jXXX@XXXX.com	2021-04-12
3	102	Lucia	Garote Lopez	XXXXX0385	Activo	fXXX@XXXX.com	1959-01-01

Por último, el Alto Mando nos encarga eliminar los usuarios y la tabla tras las recientes filtraciones.

```
DROP USER subof;  
DROP USER tropa;  
USE MASTER;  
DROP DATABASE Personal_GFJJ;
```

Row Encryption:

El capitán de la Compañía de Plana Mayor y Servicios nos solicita que creamos una tabla en donde se pueda registrar el Estadillo diario de la Compañía y que cada Sección pueda subir datos pero solamente pueda ver los datos que introduce. La oficina debería de poder ver todos los datos. Para ello vamos a aplicar seguridad a nivel de Fila. Comenzamos creando la tabla.

```
DROP DATABASE IF EXISTS PLMS;
CREATE DATABASE PLMS;
USE PLMS;

CREATE TABLE ESTADILLO (
    ID int NOT NULL IDENTITY PRIMARY KEY,
    Nombre nvarchar(20) NULL,
    Apellido nvarchar(20) NULL,
    Situacion varchar(20) CONSTRAINT DFLTSituacion DEFAULT ('Presente')
        CONSTRAINT CHKSituacion
        CHECK (Situacion in ('Presente', 'Guardia', 'Ausente', 'Vacaciones')),
    fecha date NOT NULL,
    usuario VARCHAR);
GO
DROP USER IF EXISTS Oficina;
CREATE USER Oficina WITHOUT LOGIN;
DROP USER IF EXISTS Despensa;
CREATE USER Despensa WITHOUT LOGIN;
DROP USER IF EXISTS Jardin;
CREATE USER Jardin WITHOUT LOGIN;
GO
DROP USER IF EXISTS Secretaria;
CREATE USER Secretaria WITHOUT LOGIN;
GO
GRANT SELECT, INSERT ON ESTADILLO TO Despensa, Jardin, Secretaria
GRANT SELECT ON ESTADILLO TO Oficina
GRANT EXECUTE TO Despensa, Jardin, Secretaria
GO
```

Creamos la política de seguridad y el filtro a aplicar. En este caso, filtra para que solamente veamos las filas que fueron creadas por el mismo usuario. Los usuarios dbo y Oficina deberían de poder ver todas las filas.

```
DROP SECURITY POLICY IF EXISTS dbo.ESTADILLO_SecurityPolicy ;
GO
DROP FUNCTION usuario$SecurityPredicate;
CREATE FUNCTION dbo.usuario$SecurityPredicate (@usuario AS sysname)
    RETURNS TABLE
WITH SCHEMABINDING
AS
    RETURN (SELECT 1 AS usuario$SecurityPredicate
        WHERE @usuario = USER_NAME()
            OR (USER_NAME() IN ('Oficina', 'dbo')));
GO
CREATE SECURITY POLICY dbo.RLSGFJJ
ADD FILTER PREDICATE dbo.usuario$SecurityPredicate(usuario) ON dbo.ESTADILLO
WITH (STATE = ON);
```

Para hacer las inserciones más automatizadas y sencillas, vamos a hacer un procedimiento almacenado, que a su vez nos aporta un pelín más de seguridad. Vamos a hacer que coja automáticamente la fecha, para evitar que se pueda trampear y fichar cuando no corresponde.

```
DROP PROCEDURE IF EXISTS Nuevo_estadillo;
CREATE OR ALTER PROCEDURE Nuevo_estadillo
    @nombre AS VARCHAR(20),
    @apellido AS VARCHAR(20),
    @situacion AS varchar(20)
AS
```

```

BEGIN
DECLARE @fecha AS DATE;
SET @fecha = (SELECT CONVERT (date, GETDATE()));
INSERT INTO dbo.ESTADILLO (nombre, apellido, situacion, fecha, usuario) VALUES (@nombre,@apellido,@situacion,@fecha,NULL);
END;

```

Vamos a comprobar que funciona correctamente. Para ello nos conectamos como despensa y damos la novedad de dos compañeros:

```

EXECUTE AS USER = 'despensa';
GO

```

```

EXEC dbo.Nuevo_estadillo 'Alberto', 'Sampaio Lago', 'Vacaciones';
EXEC dbo.Nuevo_estadillo 'Lucia', 'Sanchez Fernandez', 'Guardia';
GO
SELECT * FROM ESTADILLO;
REVERT;

```

```

EXECUTE AS USER = 'despensa';
GO

EXEC dbo.Nuevo_estadillo 'Alberto', 'Sampaio Lago', 'Vacaciones';
EXEC dbo.Nuevo_estadillo 'Lucia', 'Sanchez Fernandez', 'Guardia';
GO
SELECT * FROM ESTADILLO;
REVERT;

```

ID	Nombre	Apellido	Situacion	fecha	usuario	
1	5	Alberto	Sampaio Lago	Vacaciones	2021-05-26	Despensa
2	6	Lucia	Sanchez Fernandez	Guardia	2021-05-26	Despensa

Podemos comprobar que en teoría van 6 registros pero solamente vemos 2. Vamos a probar con otro usuario. Con secretaría:

```

EXECUTE AS USER = 'secretaria';
GO

```

```

EXEC dbo.Nuevo_estadillo 'Marta', 'Picallo Peña', 'Ausente';
EXEC dbo.Nuevo_estadillo 'Brais', 'Fustes Santiago', 'Guardia';
GO
SELECT * FROM ESTADILLO;
REVERT;

```

```

EXECUTE AS USER = 'secretaria';
GO

EXEC dbo.Nuevo_estadillo 'Marta', 'Picallo Peña', 'Ausente';
EXEC dbo.Nuevo_estadillo 'Brais', 'Fustes Santiago', 'Guardia';
GO
SELECT * FROM ESTADILLO;
REVERT;

```

ID	Nombre	Apellido	Situacion	fecha	usuario	
1	9	Marta	Picallo Peña	Ausente	2021-05-26	Secretaria
2	10	Brais	Fustes Santiago	Guardia	2021-05-26	Secretaria

Por último, nos queda comprobar que el usuario Oficina puede ver absolutamente todos los registros. No le dimos permiso de Ejecución ni inserción, de modo que solamente debería de poder consultar datos. Vamos a intentar hacer un registro:

```
EXECUTE AS USER = 'Oficina';
GO
EXEC dbo.Nuevo_estadillo 'Antonio', 'Resines', 'Ausente';
REVERT;
```

Msg 229, Level 14, State 6, Procedure dbo.Nuevo_estadillo, Line 1 [Batch Start Line 3]
The EXECUTE permission was denied on the object 'Nuevo_estadillo', database 'PLMS', schema 'dbo'.

Como era de esperar no nos deja. Ahora vamos a consultar los datos, debería de dejarnos

```
EXECUTE AS USER = 'Oficina';
GO
SELECT * FROM ESTADILLO;
REVERT;
```

```
EXECUTE AS USER = 'Oficina';
GO
SELECT * FROM ESTADILLO;
REVERT;
```

Results

ID	Nombre	Apellido	Situacion	fecha	usuario
4	Alberto	Sampaio Lago	Vacaciones	2021-05-26	Despensa
5	Lucia	Sanchez Fernandez	Guardia	2021-05-26	Despensa
6	Marta	Picallo Peña	Ausente	2021-05-26	Despensa
7	Brais	Fustes Santiago	Guardia	2021-05-26	Despensa
8	Marta	Picallo Peña	Ausente	2021-05-26	Secretaria
9	Brais	Fustes Santiago	Guardia	2021-05-26	Secretaria

Podemos consultar los datos de varios usuarios, como era previsto. La seguridad a nivel de filas nos permite en una misma tabla ocultar la información a otros usuarios. Es muy válida para tablas en las que distintos organismos tengan que introducir/manipular datos pero no puedan insertar/ver los datos de otros usuarios.

Allways Encrypted:

El Equipo Operativo de Seguridad número 2 está realizando maniobras conjuntas con la Fragata Álvaro de Bazán. Se quiere llevar un registro de las actividades realizadas cada día por el Equipo Operativo, pero no se puede enviar de forma segura así que nos piden que puedan almacenarlo de forma segura en un ordenador en el Buque y traerlo posteriormente al acuartelamiento, en donde se podrá descifrar y manipular la información. Para ello vamos a utilizar la funcionalidad de SSMS de Allways Encrypted.

Allways Encrypted es una característica diseñada para cifrar información confidencial en aplicaciones cliente y nunca revelar las claves de cifrado en Motor de base de datos (SQL Database o SQL Server). Como resultado, Always Encrypted proporciona una separación entre aquellos que poseen los datos y pueden verlos, y aquellos que los administran, pero que no deberían tener acceso. Al asegurar que los administradores de base de datos local, los operadores de base de datos en la nube y otros con usuarios con privilegios elevados no autorizados no pueden obtener acceso a los datos cifrados, Always Encrypted permite a los clientes almacenar información confidencial de forma segura fuera de su control directo.

Always Encrypted admite dos tipos de cifrado:

1. El **cifrado determinista** genera siempre el mismo valor cifrado para cualquier valor de texto no cifrado concreto.
2. El **cifrado aleatorio** usa un método que cifra los datos de una manera menos predecible. El cifrado aleatorio es más seguro, pero evita las búsquedas, la agrupación, la indexación y la combinación en columnas cifradas.

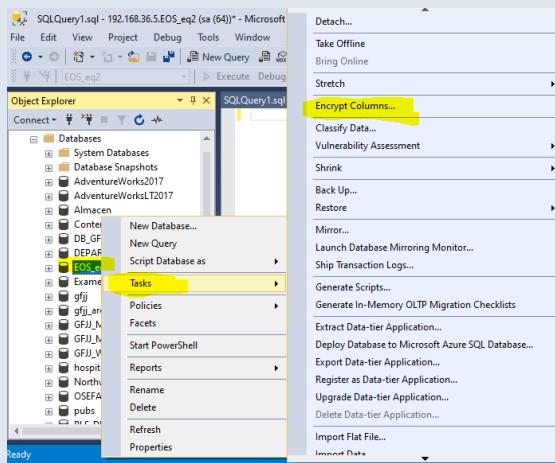
Comenzamos creando la BD, la tabla y el contenido.

```
DROP DATABASE IF EXISTS EOS_eq2
CREATE DATABASE EOS_eq2
GO
USE EOS_eq2
GO
DROP TABLE IF EXISTS F102
GO

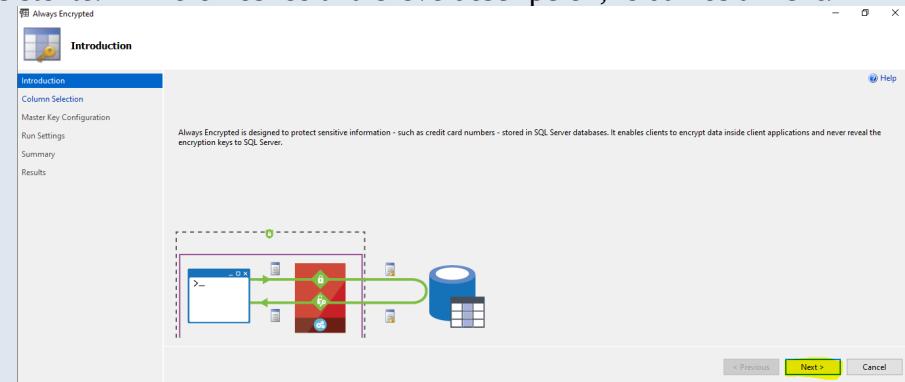
-- Tenemos que utilizar el tipo de Collate Latin1_General_BIN2 porque si no no funciona.
CREATE TABLE F102
(
    ID int identity primary key,
    Fecha date NOT NULL,
    Cod_Sdo varchar(100) COLLATE Latin1_General_BIN2 NOT NULL,
    Actividad varchar(100) COLLATE Latin1_General_BIN2 NOT NULL
)
GO

INSERT INTO F102 ( Fecha, Cod_Sdo, Actividad)
VALUES ('2020-03-24','E02128Y', 'Tiro Nocturno'),
       ('2020-03-24','E02194L', 'Tiro de Precision'),
       ('2020-03-24','E027500', 'Fast-Rope'),
       ('2020-03-25','E02194L', 'Boarding'),
       ('2020-03-25','E027500', 'Abordaje'),
       ('2020-03-26','E02128Y', 'Fast-Rope')
GO
```

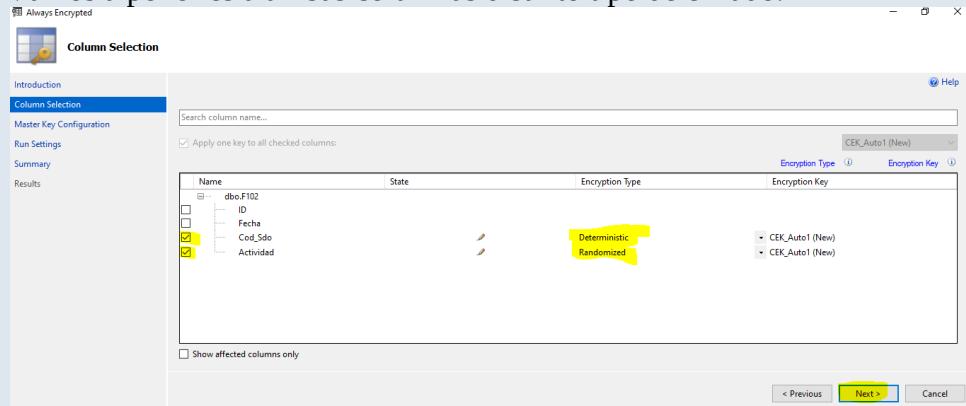
Ahora en entorno gráfico hacemos click derecho sobre la BD y dejamos el cursor en **Tasks → Encrypt Columns** como vemos en la imagen.



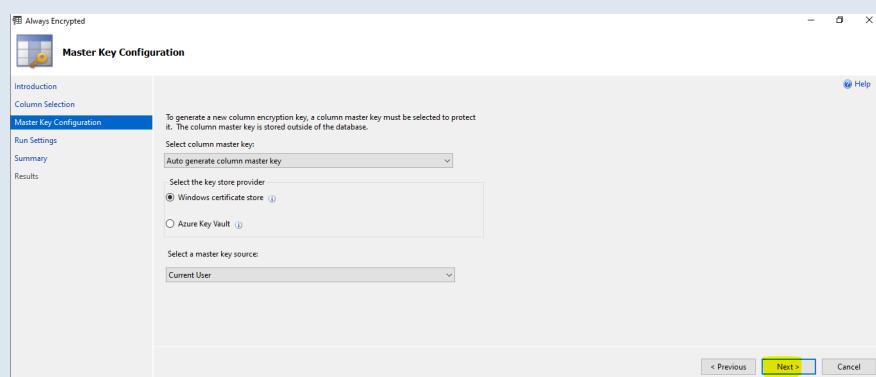
Nos abre un asistente. Primero nos lee una breve descripción, le damos a **Next**.



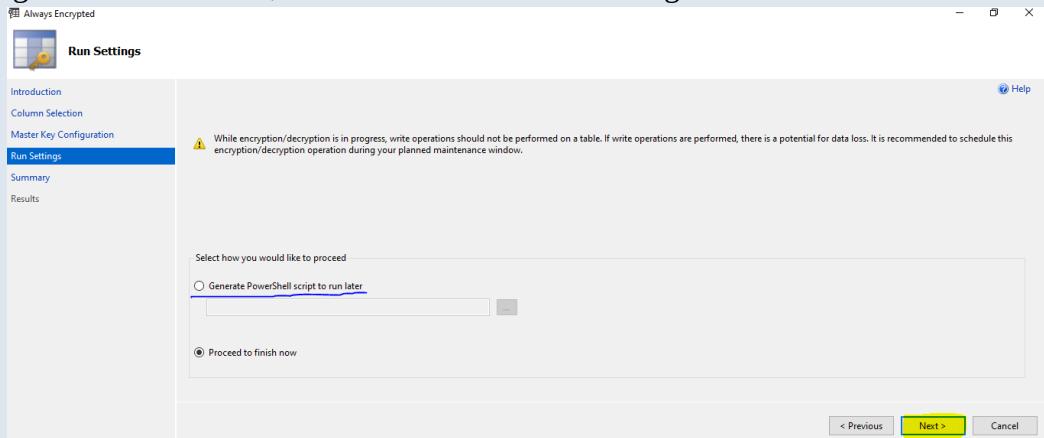
Ahora tenemos que escoger qué columnas queremos cifrar y con qué tipo de cifrado. En este caso vamos a cifrar el Código Militar (que identifica a cada miembro del equipo) así como la Actividad que realiza. Vamos a ponerles a ambas columnas distinto tipo de cifrado.



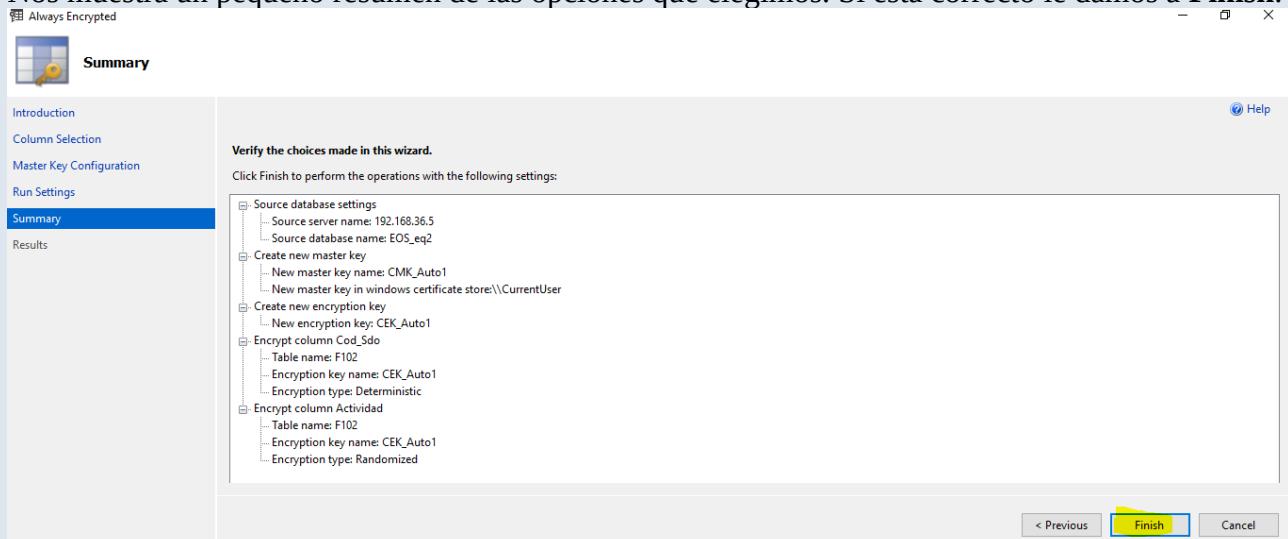
Nos permite seleccionar la clave maestra que queremos seleccionar, así como si es gestionada a través de los certificados de Windows o de Azure. Vamos a dejar en nuestro caso el de Windows.



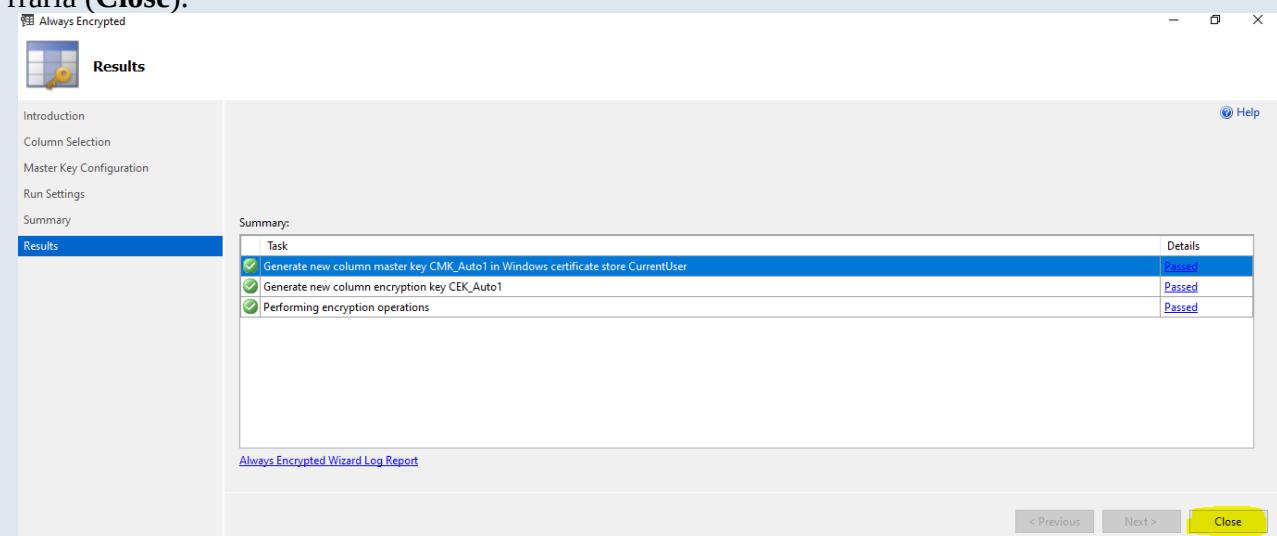
Ahora nos informa de que podemos generar un script de PowerShell para realizar las operaciones en la tabla durante el mantenimiento, lo recomiendan por motivos de seguridad para evitar pérdida de datos. En nuestro caso como sería un operador que introduce manualmente esos datos y no se harían múltiples gestiones en la tabla, no deberíamos de tener incongruencia de datos. Le damos a Next.



Nos muestra un pequeño resumen de las opciones que elegimos. Si está correcto le damos a **Finish**.



Después de unos segundos, si todo está correcto nos muestra la ventana en verde y nos permite cerrarla (**Close**).



Ahora vamos a comprobar que están encriptadas las columnas:

```
SELECT * FROM F102;
```

ID	Fecha	Cod_Sdo	Actividad
1	2020-03-24	0x016F799FE5F4A3B3BEF3318AF128FA8DE74037BEB9646F...	0x015C303F52C4C16941B401A03ED264F5CD2FE0B3F143E98...
2	2020-03-24	0x01DCF51742E01EF1D28E20ECE57343565EBF92413AA4612...	0x01E32B52D796905393D987B586D3AD64F715558B5E8A11...
3	2020-03-24	0x01B9265FD8CA4A9606CFBCD7195DE49A6339C9FA32591...	0x0124E4A52AB932EDF59EF5F615244A9B06F022C1839A7A6...
4	2020-03-25	0x01DCF51742E01EF1D28E20ECE57343565EBF92413AA4612...	0x01A5DA1B830471C042719E87FCDC2F0621C905750B6DB...
5	2020-03-25	0x01B9265FD8CA4A9606CFBCD7195DE49A6339C9FA32591...	0x01815BF1A6722F5E52829A784ED321BF322D50C4C5B6...
6	2020-03-26	0x016F799FE5F4A3B3BEF3318AF128FA8DE74037BEB9646F...	0x01FAE092A4F569634065DE73929B8CA03C9E5D4CA65883...

Como podemos comprobar en la imagen, ambas columnas están cifradas. Vamos a fijarnos en el tipo de cifrado.

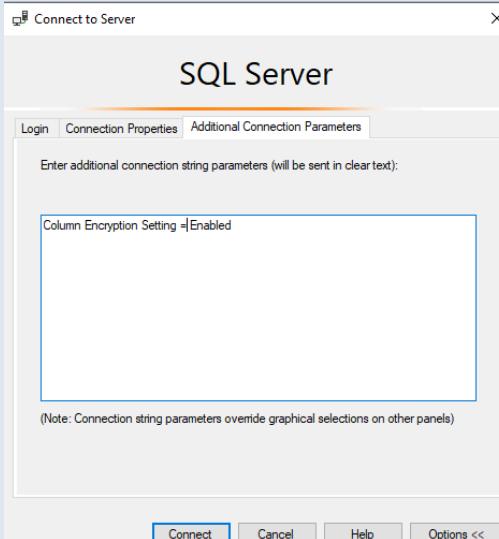
La columna Cod_Sdo tiene puesta un cifrado determinista. Genera una serie de valores aleatorios, pero ... Si nos fijamos bien, los registros 1 y 6 de esa columna tienen el mismo código. Al ser cifrado determinista genera un código aleatorio, pero al ser campos idénticos expone el mismo código aleatorio en ambas columnas. Tiene la ventaja de que podemos presuponer con la práctica (y con pocos campos, claro) a quién/qué corresponde ese valor. También nos valdría para saber cuántos registros con el mismo valor tenemos. Por contra, aunque no sepamos cual es el contenido real sí podemos determinar cuantas veces se repite ese registro.

La columna Actividad tiene puesto un cifrado aleatorio. Como podemos observar, los registros 3 y 6 tienen el mismo valor, pero sin embargo se les ha generado un valor aleatorio a ambas columnas, de modo que a simple vista todos los registros son distintos. No podríamos calcular cuantos registros idénticos muestra la tabla.

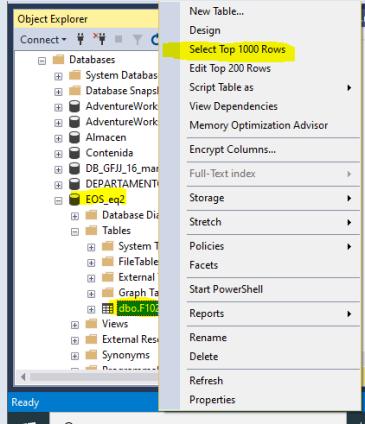
DATABASE=Contenida;

Ahora vamos a cerrar la sesión actual en SSMS y abrimos otra... Pero antes de conectarnos le damos a Options → Additional Connection Parameters y ponemos:

Column Encryption Setting = Enabled



Hecho eso le damos a Connect. La primera consulta que hagamos tiene que ser por entorno gráfico. Hacemos click derecho en la tabla y le damos a **Select Top 1000 Rows**.



Como vemos, nos muestra las columnas sin enmascarar:

A screenshot of the SSMS Query Results window. It displays a table with the following data:

ID	Fecha	Cod_Sdo	Actividad
1	2020-03-24	E02128Y	Tiro Nocturno
2	2020-03-24	E02194L	Tiro de Precision
3	2020-03-24	E027500	Fast-Rope
4	2020-03-25	E02194L	Boarding
5	2020-03-25	E027500	Abordaje
6	2020-03-26	E02128Y	Fast-Rope

Sin embargo, si nos conectamos nuevamente pero sin poner el “Column Encryption Setting = Enabled” en las opciones, nos sigue apareciendo cifrado el contenido:

SQLQuery7.sql - 192...Contenida (sa (72)) + X

```
***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [ID]
      ,[Fecha]
      ,[Cod_Sdo]
      ,[Actividad]
  FROM [EOS_eq2].[dbo].[F102]
```

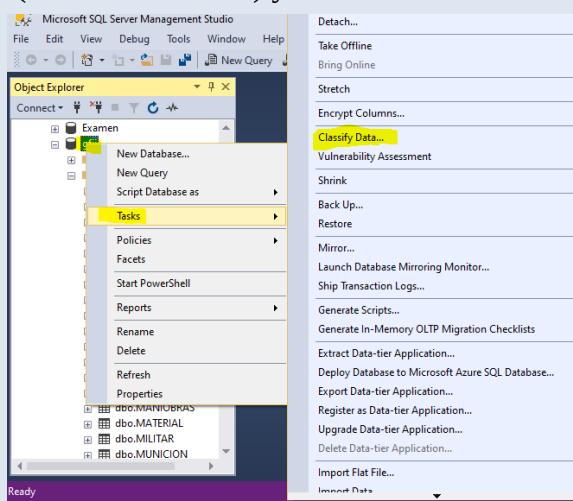
100 %

Results Messages

	ID	Fecha	Cod_Sdo	Actividad
1	1	2020-03-24	0x016F799FE5F4A3B3BEBF3318AF128FA8DE74037BEB9646F...	0x015C303F52C4C16941B401A03ED264F5CD2FE0B3F143E98...
2	2	2020-03-24	0x01DCF51742E01EF1D28E20ECE57343565EBF92413AA4612...	0x01E32B52D796905393D987B5B6D3AD64F71555885E8A11...
3	3	2020-03-24	0x01B9265FDF8CA4F9606CFBCD7195DE49A6339C9FA32591...	0x0124E4A52AB932EDF59EF5F615244A9B06F022C1839A7A6...
4	4	2020-03-25	0x01DCF51742E01EF1D28E20ECE57343565EBF92413AA4612...	0x01A5DA1B830471C042719E87FCDC2F0621C905750B6DB...
5	5	2020-03-25	0x01B9265FDF8CA4F9606CFBCD7195DE49A6339C9FA32591...	0x01815BF1A6722F5E52829A784EED321BF322DD50C4C5B6...
6	6	2020-03-26	0x016F799FE5F4A3B3BEBF3318AF128FA8DE74037BEB9646F...	0x01FAE092A4F569634065DE73929B8CA03C9E5D4CA65883...

SSMS:

Tras las filtraciones de datos mencionadas anteriormente, el Comandante encargado de la Oficina de Seguridad Física nos solicita una clasificación que contenga la confidencialidad que debería de tener cada columna de la BD original. Para gestionarlo eficientemente vamos a utilizar la herramienta de **clasificación y detección de datos integrada** en SSMS. Dentro de SSMS, hacemos click derecho en la BD a analizar (en este caso GFJJ) y le damos a **Tasks → Classify Data...**

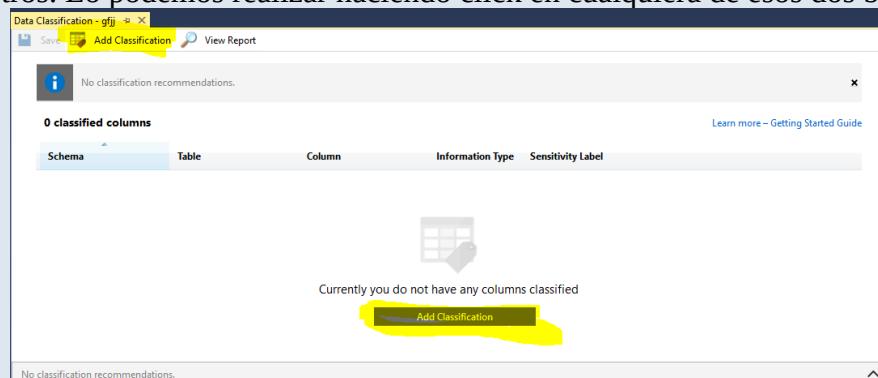


Al darle, pasan unos segundos y analiza la BD y nos muestra (si procede) las recomendaciones a seguir para etiquetar/clasificar las columnas de la BD. En este caso, la BD de mi proyecto no tiene recomendaciones así que las vamos a cargar manualmente. Pero vamos a coger la BD de AdventureWorks un momento para verificar cómo saldría la recomendación.

A screenshot of the "Data Classification" tool within SSMS. The left pane shows the "Object Explorer" with the "AdventureWorksLT2017" database selected. The main pane displays a table titled "0 classified columns" with a message "45 columns with classification recommendations (click to minimize)". Below this, a section titled "Accept selected recommendations" shows a list of 45 recommendations. The table has columns: Schema, Table, Column, Information Type, and Sensitivity Label. Most recommendations are for columns in the "DIRECCIONES" table, with "AddressLine1", "AddressLine2", "City", and "PostalCode" all set to "Contact Info" with a sensitivity label of "Confidential - GDPR". There is also one recommendation for the "UserName" column in the "ErrorLog" table, labeled as "Credentials" with a sensitivity label of "Confidential".

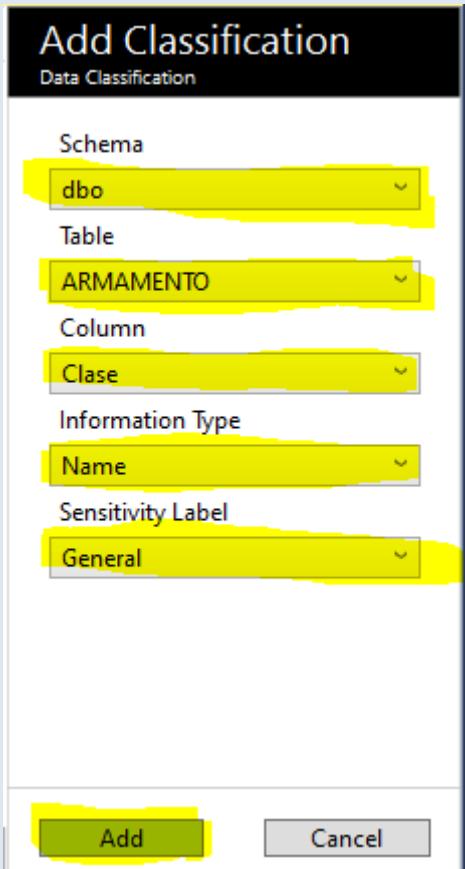
Como vemos, nos salen 45 recomendaciones de etiquetación de columnas. Seleccionaríamos en el recuadro las que queremos (o en el primer recuadro marcamos las 45), y si quisieramos cambiar alguna iríamos al menú desplegable y las cambiaríamos. Cuando estén, pulsaríamos en **Accept Selected Recommendations**.

Volviendo a nuestra BD, como no nos figura ninguna recomendación tenemos que cargar manualmente los registros. Lo podemos realizar haciendo click en cualquiera de esos dos botones.



Al darle a Add Classification nos abre un pequeño formulario con menús desplegables en donde tenemos que poner:

- a) El schema que queremos elegir (en este caso solamente trabajé con dbo, pero podríamos tener varios)
- b) La tabla que vamos a seleccionar. Hay que ir introduciendo una por una.
- c) La columna de la tabla que queremos etiquetar.
- d) El tipo de información, dentro de una lista predefinida que tiene la herramienta.
- e) La clasificación de seguridad. En este caso sería General/Pública, puesto que sería la clase de armamento y no revela información clasificada. Si fuera por ejemplo el número de cuenta de alguien o el de la SS sí habría que etiquetarlo con CONFIDENCIAL o la que estimásemos oportuna.



Le damos a add y tenemos la clasificación añadida para esa columna.

The screenshot shows the 'Data Classification - gfjj' interface. At the top, there are buttons for Save, Add Classification, and View Report. A message says: 'There are pending classification updates. Please save.' Below is a table with the following data:

Schema	Table	Column	Information Type	Sensitivity Label
dbo	ARMAMENTO	Clase	Name	General

A message at the bottom says: 'No classification recommendations.'

Como vemos, ahora nos muestra los cambios que hemos implementado, y nos dice que no están clasificados porque no le dimos a guardar. Procedemos a darle a **Save**.

No nos muestra (de momento) ninguna recomendación, así que vamos a ir introduciendo la clasificación de TODAS las columnas de la BD.

Hemos clasificado la información de 63 columnas.

The screenshot shows the 'Data Classification' interface with a success message: 'The classification changes have been updated successfully.' Below this, it says 'No classification recommendations.' and lists '63 classified columns' in a table. The table has columns: Schema, Table, Column, Information Type, Sensitivity Label, and a delete icon. The data includes:

Schema	Table	Column	Information Type	Sensitivity Label	
dbo	DESTINO	Compañía	Name	Public	
dbo	DESTINO	Destino	Contact Info	General	
dbo	DESTINO	Encargado	Contact Info	Confidential	
dbo	MUNICION	Cant_Mun	Financial	Confidential	
dbo	MUNICION	Mun_MM	Name	Public	

No classification recommendations.

Nos nos asigna ninguna recomendación a realizar, pero vamos a resumir brevemente en qué consisten las dos casillas con opciones.

El menú de tipo de información nos permite clasificar de entre unos genéricos qué tipo de información almacena. Por ejemplo, si es el número de cuenta sería información de tipo "Banking". El nombre sería de tipo "Name", el dni de tipo "ID". Es bastante genérico, pero sirve para que nos hagamos una pequeña idea sobre qué tipo de datos contiene la columna.

El nivel de sensibilidad varía desde el público (información que puede saber absolutamente todo el mundo), que en la captura vemos por ejemplo que el campo Mun_MM es información pública. Ese campo concretamente es el tipo de munición que utiliza un arma concreta. Es pública, porque se puede consultar perfectamente en Internet, y si se deslizara esa información no supondría nada.

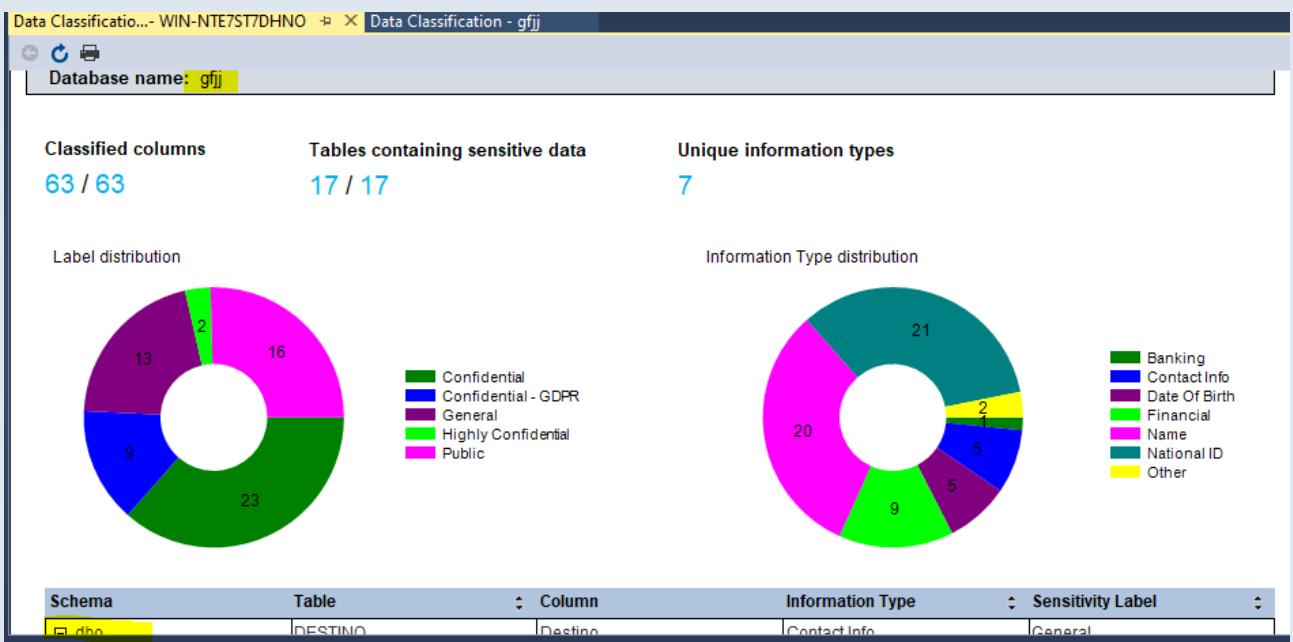
El siguiente escalón sería General. Sería que cualquiera en la empresa podría tener acceso a esa información, pero no se puede hacer uso público de ella. En la captura vemos el campo Destino, que dentro del cuartel todo el mundo puede saber en donde está destinado cada uno, pero no se puede difundir fuera del ámbito militar.

El siguiente grado es Confidencialidad (GDPR o no), que sería información que no está al alcance de todo el mundo dentro de la empresa. Por ejemplo, los códigos de Operación, Números de cuenta del personal(GDPR), números de DNI (GDPR).

Por último tenemos la información secreta o de máxima confidencialidad. Ésta sería información que tendrían acceso el mínimo de personas necesario y no se debería de extender más allá. He clasificado así las fechas de unas maniobras (porque exponerlas, según la maniobra, podría poner en peligro a los miembros implicados) o por ejemplo las cuantías y fechas de las sanciones (que solamente las sabría el Jefe de Unidad, el Jefe de Personal y el interesado).

Por último, si le damos a **View Report** podemos generar y ver un informe sobre la clasificación de la información.

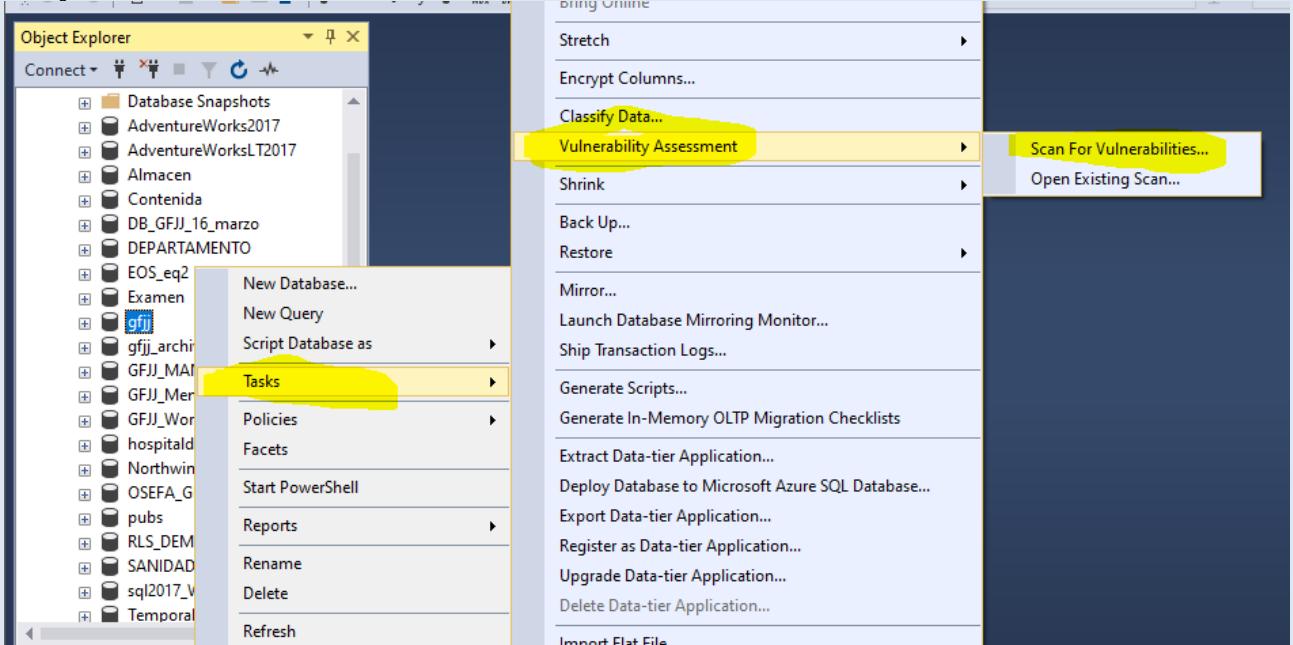
The screenshot shows the 'Data Classification' interface with a success message: 'The classification changes have been updated successfully.' Below this, there is a 'View Report' button highlighted with a yellow box.



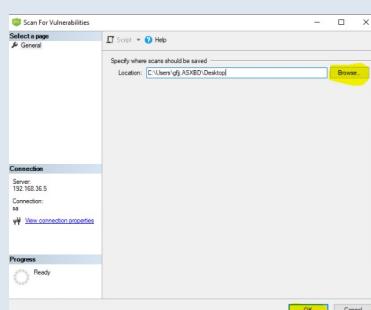
Podemos exportar el informe haciendo click derecho Export → Excel/PDF/Word.

EVALUACIÓN DE VULNERABILIDADES DE SQL SERVER

Vamos a efectuar la evaluación de vulnerabilidades de nuestra BD, a través de la herramienta propia de SQL SERVER. Para ello hacemos click derecho en nuestra BD y le damos a **Tasks → Vulnerability Assessment → Scan For Vulnerabilities**



Primero nos pregunta a dónde queremos exportarlo, tenemos que seleccionar la carpeta de destino y darle a **OK**. En unos segundos ejecuta el análisis y nos genera un fichero.



Simultáneamente nos muestra el resultado del análisis de seguridad.

ID	Security Check	Category	Risk	Additional Information
VA1265	Auditing of both successful and failed login attempts for contained DB authentication should be enabled	Auditing and Logging	Medium	
VA1143	'dbo' user should not be used for normal service operation	Surface Area Reduction	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
VA1054	Excessive permissions should not be granted to PUBLIC role on objects or columns	Authentication and Authorization	Low	

Como vemos, nos da 5 fallos de seguridad (2 de ellos leves) y 49 chequeos que están correctos. Si hacemos click en uno de los que nos informa que da fallo, por ejemplo en el de abajo de todo, nos muestra una breve descripción del riesgo.

✓ Approve as Baseline ✘ Clear Baseline

Name	VA1054 - Excessive permissions should not be granted to PUBLIC role on objects or columns
Risk	Low

En este caso, el riesgo es bajo y es por tener demasiados permisos concedidos de carácter público en roles/columnas que por su precedencia deberían tener permisos más estrictos. Esto es debido a que por ejemplo le pusimos permiso público a columnas que señalamos como etiqueta "Nombre". Realmente sí que son permisos públicos, puesto que a mi entender son figuras/representaciones que pueden ser públicas y de hecho figuran en la página oficial de la Armada, así como en la página del acuartelamiento. Así que le doy a "Approve as Baseline" para marcarlo como falso positivo. Si ahora hago un nuevo escaneo podemos comprobar que efectivamente ya no me figura como una incidencia.

El siguiente riesgo de tipo bajo nos indica que los usuarios que no sean de sistema deberían de tener revocados los permisos en las tablas y vistas de sistema.

✓ Approve as Baseline ✘ Clear Baseline

Name	VA1069 - Permissions to select from system tables and views should be revoked from non-sysadmins
Risk	Low

Vamos a ignorarlo tambien, puesto que voy a trabajar con el usuario dbo y si luego requiero de consultar la tabla por lo que fuera/fuese, es altamente probable que no recuerde que le revoqué los permisos y me vuelva loco. No estoy del todo seguro, pero creo que podríamos revocarlo con:

```
REVOKE SELECT ON sys.database_permissions FROM USUARIO;
```

The screenshot shows the SQL Server Security Center interface. At the top, it displays "Total security checks: 54" with a shield icon, "Total failing checks: 3" with a red X icon, and a risk distribution bar: High Risk (0), Medium Risk (3), and Low Risk (0). There are links to "Learn more", "SQL Security Center", and "Best Practices for SQL Security". Below this, a table lists security checks. The first row, VA1219, is highlighted in blue and shows "Transparent data encryption should be enabled" under "Security Check", "Data Protection" under "Category", and "Medium" under "Risk". Below the table are buttons for "Approve as Baseline" and "Clear Baseline". A tooltip for VA1219 says "VA1219 - Transparent data encryption should be enabled". On the right, there are buttons for "Activar" and "Ve a Com".

ID	Security Check	Category	Risk	Additional Information
VA1219	Transparent data encryption should be enabled	Data Protection	Medium	

El siguiente error es un error de gravedad media. Nos informa que deberíamos de tener activado el certificado TDE (como vimos previamente). Lo solventaríamos realizando:

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

CREATE CERTIFICATE TDE_Cert
WITH
SUBJECT='Database_Encryption';
GO
USE GFJJ;
GO
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE TDE_Cert;
GO
ALTER DATABASE gfjj
SET ENCRYPTION ON;
GO
```

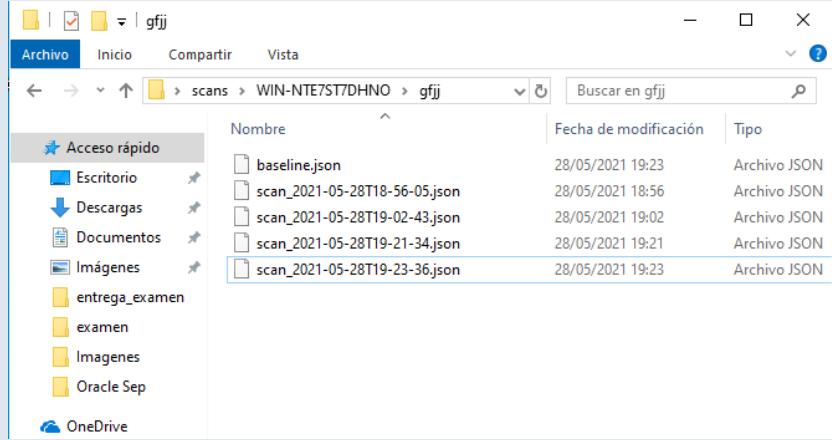
Tendríamos otro error de gravedad media por estar utilizando el usuario DBO. Esto se solventaría simplemente creando un usuario y asignándole los permisos de SELECT, INSERT, UPDATE WITH GRANT OPTION para que tambien pueda asignarlos.

The screenshot shows the SQL Server Security Center interface. At the top, it displays "Total security checks: 54" with a shield icon, "Total failing checks: 1" with a red X icon, and a risk distribution bar: High Risk (0), Medium Risk (1), and Low Risk (0). There are links to "Learn more", "SQL Security Center", and "Best Practices for SQL Security". Below this, a table lists security checks. The first row, VA1265, is highlighted in blue and shows "Auditing of both successful and failed login attempts for contained DB authentication should be enabled" under "Security Check", "Auditing and Logging" under "Category", and "Medium" under "Risk". Below the table are buttons for "Approve as Baseline" and "Clear Baseline". A tooltip for VA1265 says "VA1265 - Auditing of both successful and failed login attempts for contained DB authentication should be enabled". On the right, there are buttons for "Activar" and "Ve a Com".

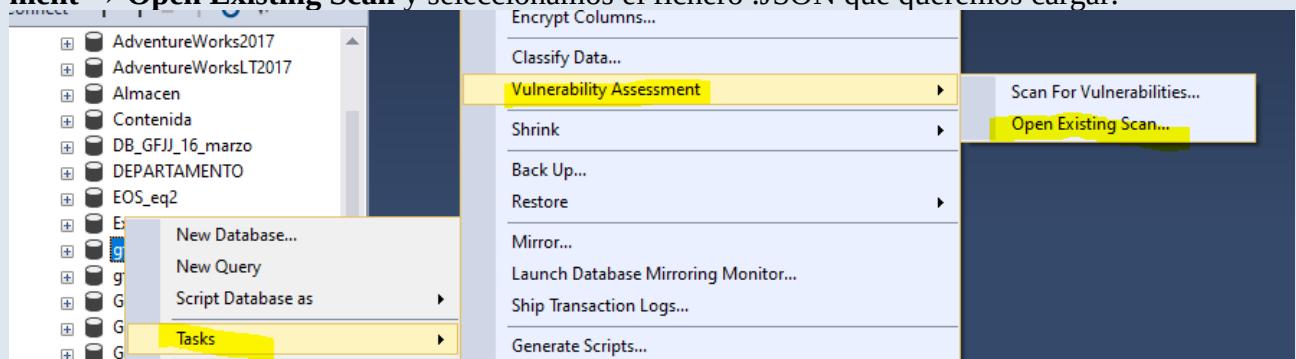
ID	Security Check	Category	Risk	Additional Information
VA1265	Auditing of both successful and failed login attempts for contained DB authentication should be enabled	Auditing and Logging	Medium	

Por último nos arroja ese error VA1265. Si vamos al manual de SQL nos informa que permite a los administradores realizar un seguimiento de los usuarios que inician sesión en instancias de SQL Server de las que son responsables. Esta regla comprueba que la auditoría está habilitada para los intentos de inicio de sesión correctos y erróneos de autenticación de bases de datos independientes. No sabría cómo solventarlo así que tendríamos 54 chequeos correctos y 1 erróneo.

Si vamos a la carpeta que nos ha creado con el escaneo, vemos que nos ha creado un fichero .JSON que contiene el informe de seguridad.



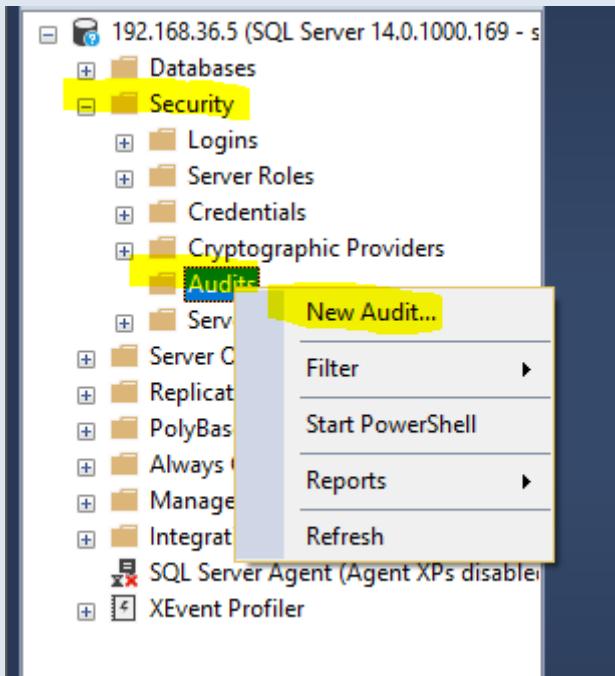
Para abrir un informe hacemos click derecho en la BD, vamos a **Tasks** → **Vulnerability Assessment** → **Open Existing Scan** y seleccionamos el fichero .JSON que queremos cargar.



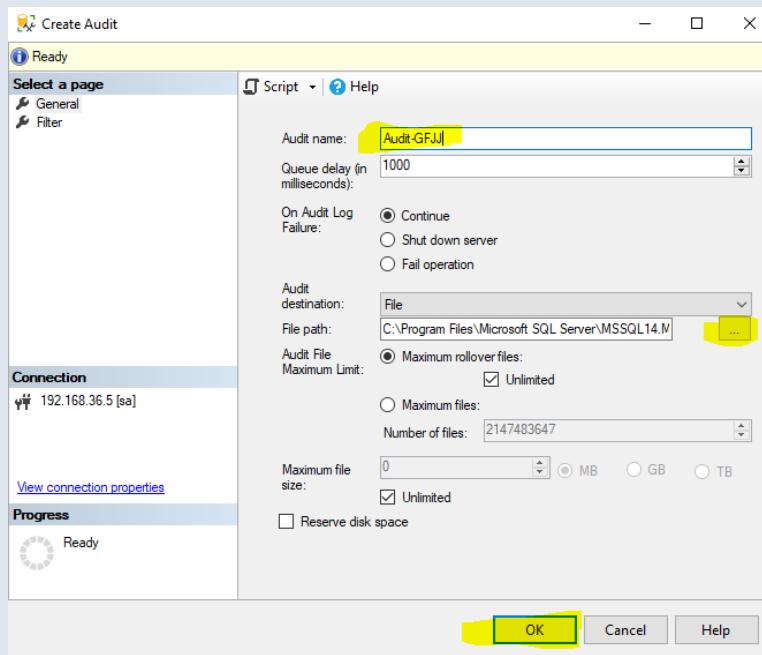
Auditoría de Servidor y Especificaciones de Servidor:

Una auditoría en SQL SERVER permite el seguimiento y registro de los eventos que se producen en el sistema. SQL Server Audit es una herramienta que recopila una única instancia de acciones de nivel de servidor o BD para su supervisión. Es posible tener varias auditorías por cada instancia de SQL Server. Se puede crear una especificación de auditoría de BD para cada BD de SQL Server.

Vamos a crear una auditoría en SQL SERVER. Se puede realizar en entorno gráfico, haciendo click derecho en **Security** → **Audits** → **New Audit**

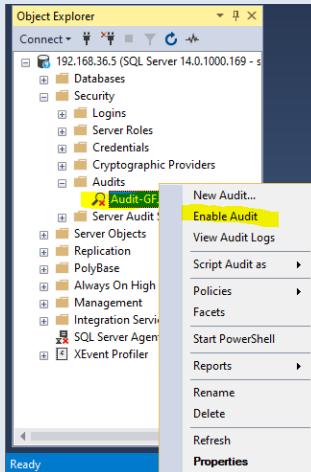


Nos abre un asistente que nos solicita que le asignemos el nombre del fichero, así como en dónde lo vamos a guardar.

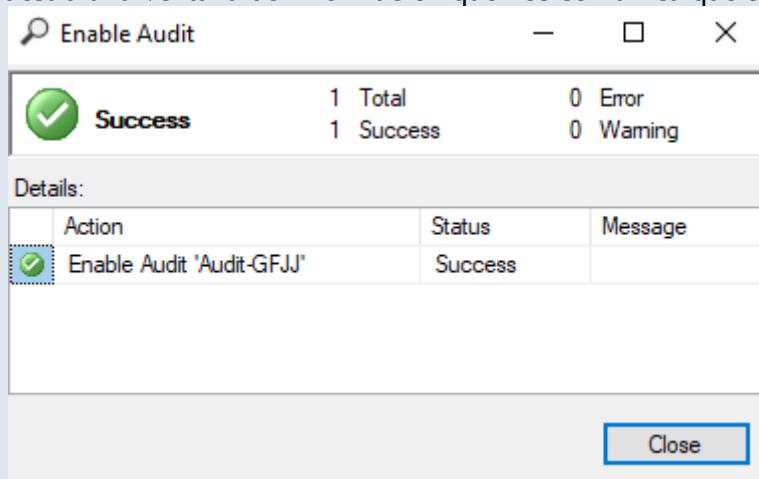


Podemos personalizar tambien el tamaño máximo del fichero, así como seleccionar qué acción debe de hacer cuando registremos un error. Por defecto sale continuar (registrar el error pero no hacer nada). Podemos seleccionar apagar el servidor (eso obviamente tiene que usarse en casos MUY puntuales y potencialmente críticos) o rechazar la operación (que tambien es peligroso, en función a lo que queramos auditar).

Una vez creado, por defecto está habilitado. Para habilitarlo hacemos click derecho sobre la auditoría y sale la opción de **Enable Audit**



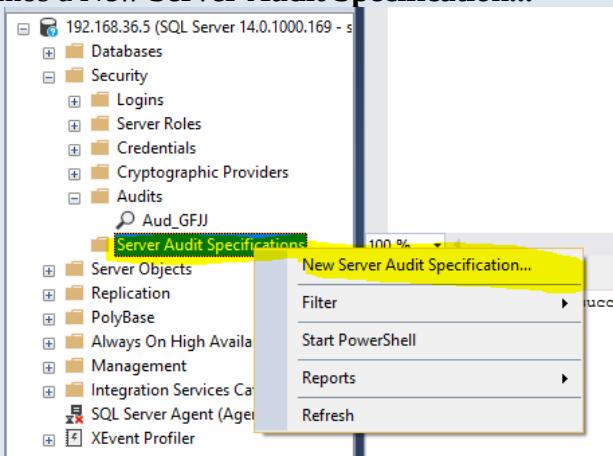
Al habilitarla nos muestra una ventana de información que nos comunica que está habilitado.



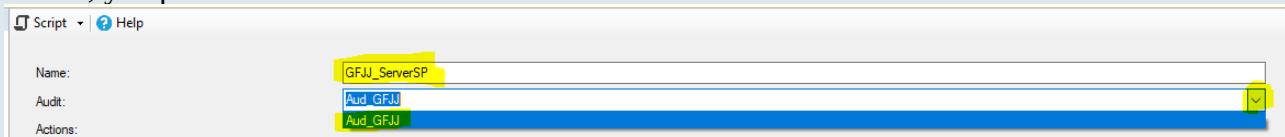
Esta misma operación se podría hacer mediante las siguientes instrucciones en una New Query de SQL SERVER.

```
USE master ;
GO
CREATE SERVER AUDIT Aud_GFJJ
    TO FILE ( FILEPATH =
'C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA' ) ;
GO
ALTER SERVER AUDIT Aud_GFJJ
WITH (STATE = ON) ;
```

Para crear una especificación de Servidor vamos a hacer click derecho en **Security → Server Audit Specifications** y le damos a **New Server Audit Specification...**

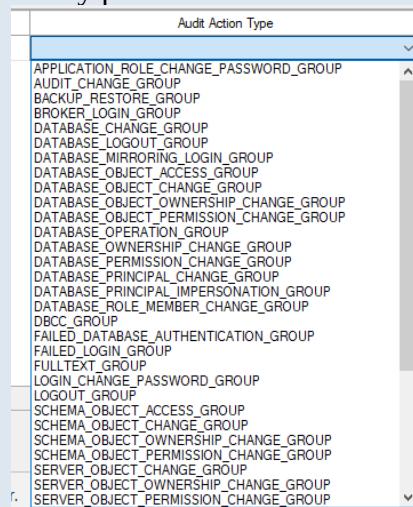


Al darle nos abre un asistente que nos requiere primero el nombre que le vamos a dar a la Especificación, y a qué Auditoría se va a hacer efecto.

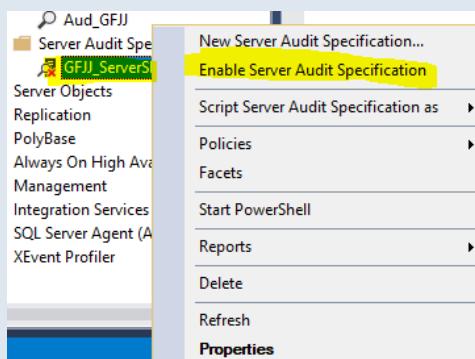


En mi caso solamente dispongo de una Auditoría, pero podríamos tener distintas con distintas especificaciones.

Lo siguiente es escoger qué tipo de acción vamos a Auditar. Es un desplegable con muchas opciones, ojo con eso que podemos liarla muy parda.

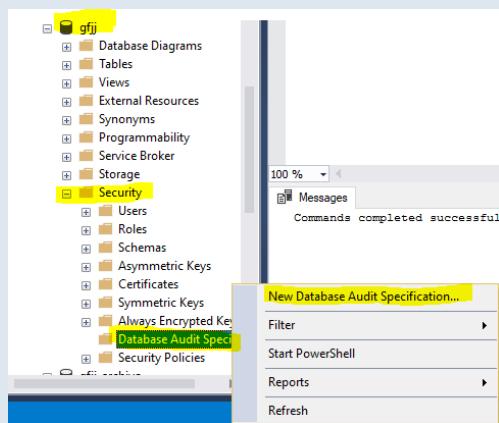


En mi caso elijo la opción de auditar el fallo de logeo en un grupo. Por defecto está deshabilitada, habría que habilitarla igual que la Auditoría, click derecho sobre ella y habilitamos la especificación.

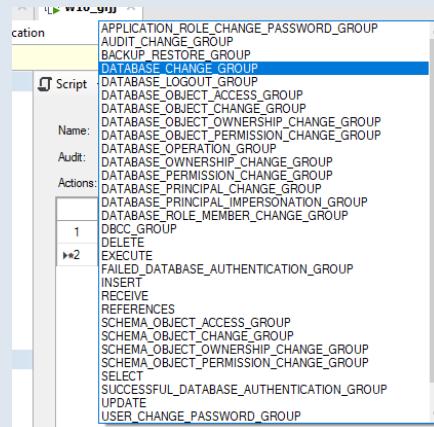


Especificaciones de Bases de Datos:

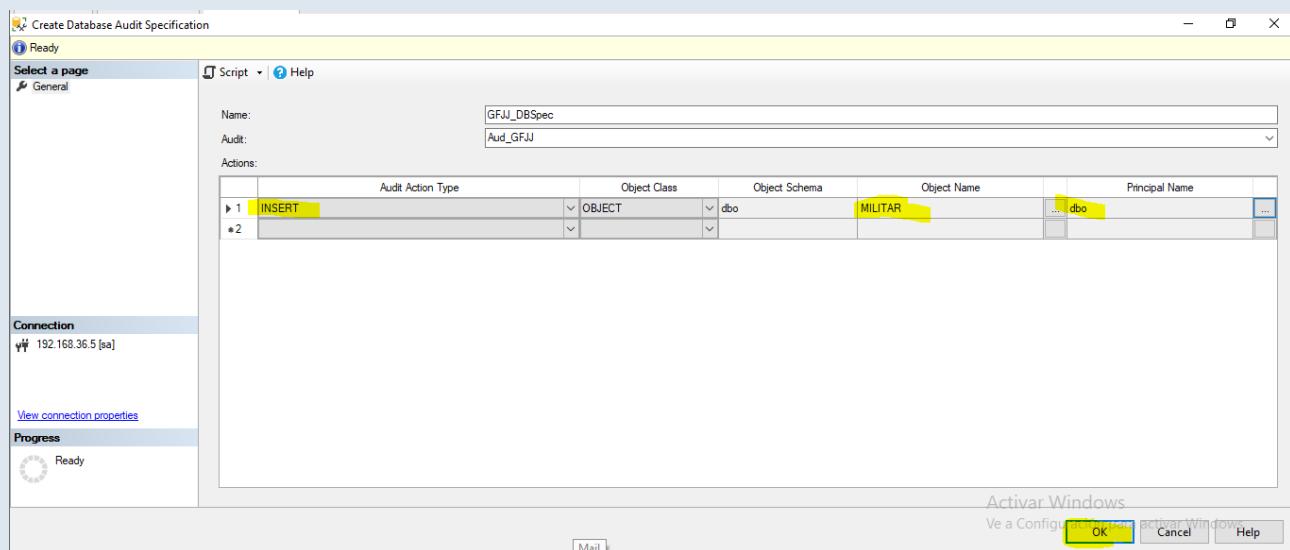
Las especificaciones de BD funcionan de manera similar a las especificaciones de Servidor, pero éstas auditán solamente registros que afectan a la BD. Para crear una especificación de BD vamos a nuestra BD → **Security** → **Database Audit Specifications** (click derecho) → **New Database Audit Specification**.



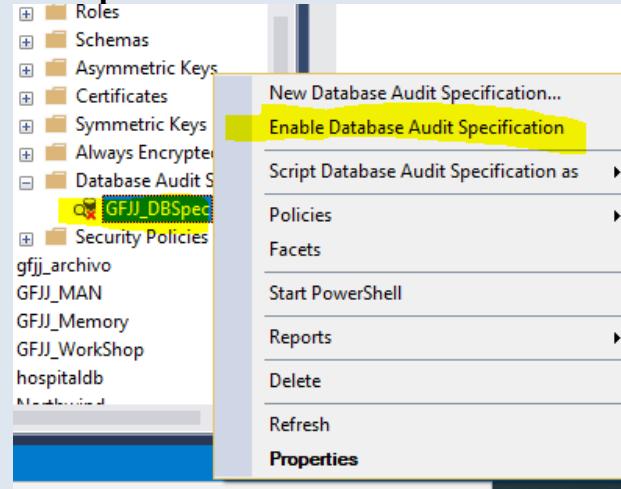
Nos sale la misma pestaña que en la auditoría de Servidor, pero nos muestra otros registros muy distintos.



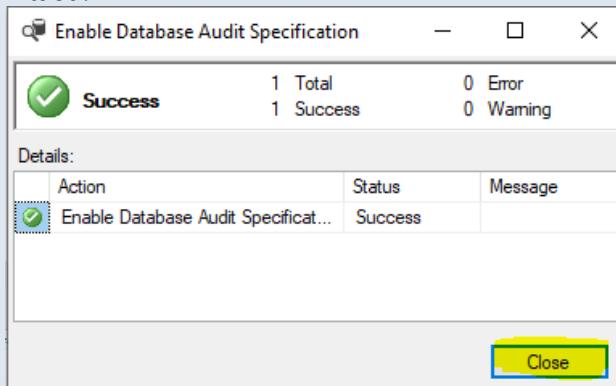
En este caso vamos a seleccionar que audite cuando se inserta datos en la tabla MILITAR por parte del usuario dbo.



Habilitamos la auditoría del mismo modo que hicimos en la del Servidor. Click derecho en la especificación y **Enable DB Audit Specification**.



Nos informa que está habilitada.



Vamos a introducir un registro en la tabla MILITAR para comprobar que lo audita:

```
USE gfjj;
INSERT INTO MILITAR (DNI, NOMBRE, NS_Arm, Destino, Empleo) VALUES ('32480603C', 'Roberto Araujo Suarez', '5649G', 'PN', 'Soldado');
```

Comprobamos que efectivamente lo está auditando si ponemos la siguiente instrucción. Ojo, muy importante poner la ruta/XXXX.sqlaudit

```
SELECT event_time,action_id, statement, database_name, server_principal_name
  FROM fn_get_audit_file( 'C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\*.sqlaudit' , DEFAULT , DEFAULT );
```

The screenshot shows the results of a query against the audit file. The query is: 'SELECT event_time,action_id, statement, database_name, server_principal_name FROM fn_get_audit_file('C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA*.sqlaudit' , DEFAULT , DEFAULT)'. The results table has columns: event_time, action_id, statement, database_name, and server_principal_name. There are five rows of data, with the fifth row highlighted in yellow. The fifth row corresponds to the 'INSERT INTO MILITAR' query from the previous step.

	event_time	action_id	statement	database_name	server_principal_name
1	2021-05-28 16:46:03.7696009	AUSC			sa
2	2021-05-28 16:47:41.4571244	AUSC			sa
3	2021-05-28 16:47:41.4571244	AUSC			sa
4	2021-05-28 16:50:04.0667939	AUSC			sa
5	2021-05-28 17:37:51.9322566	IN	INSERT INTO MILITAR (DNI, NOMBRE, NS_Arm, Destin... gfjj		sa

Como vemos en la captura, está auditando correctamente. Está registrando el Insert que realizamos en la tabla MILITAR.

Es una herramienta MUY interesante cuando se trabaja con múltiples usuarios en la misma tabla y se quiere cotejar quien hace cada modificación y a qué hora. Así si alguien introduce un registro y otro usuario lo cambia “por error” queda reflejado.

Por contra, hay que tener en cuenta que a mayor fiscalización de los registros, va a ralentizar mucho cada proceso, así como aumentar progresivamente el coste de almacenamiento de la BD.

Si estamos almacenando pocos registros diarios no se va a notar. Pero si almacenamos miles de registros es MUY factible tanto que colapse el sistema como que se sature el fichero, creando desestabilidad y/o posible pérdida de la información.

General Data Protection Regulation :

El 25 de mayo de 2018 entró en vigor la ley GDPR (Reglamento General de Protección de Datos) en Europa. Regula directamente el almacenamiento, procesamiento, acceso, transferencia y divulgación de los registros de datos de un individuo y afecta a cualquier organización a nivel mundial que procese datos personales de personas de la Unión Europea.

El no cumplimiento del GDPR puede acarrear sanciones de una cuantía muy elevada (hasta un 2 a un 4% de la economía global de la empresa).

A grandes rasgos, las características que regula la GDPR serían:

Privacidad personal

Las personas tienen derecho a:

- I. Acceder a sus datos personales
- II. Corregir errores en sus datos personales
- III. Eliminar sus datos personales
- IV. Denegar el procesamiento de su información personal
- V. Exportar sus datos personales

Controles y notificaciones

La organización debe:

- I. Proteger los datos personales con las medidas de seguridad adecuadas
- II. Notificar a las autoridades de las filtraciones de datos
- III. Obtener el consentimiento del titular para procesar tus datos
- IV. Mantener registro de los procesos que afecten a los datos personales

Transparencia

La organización debe:

- I. Informar con claridad de sus actividades de captura de datos.
- II. Explicar adecuadamente la finalidad del procesamiento de los datos personales y los casos de uso.
- III. Definir políticas de retención y eliminación.

TI y formación

La organización debe:

- I. Formar a empleados y contratar en las obligaciones de privacidad.
- II. Auditarse y actualizar sus políticas de gestión de datos.
- III. Designar un responsable DPO (Data Protection Officer) cuando corresponda.
- IV. Incorporar cláusulas de cumplimiento en sus contratos con terceros.

FASE DE DESCUBRIMIENTO:

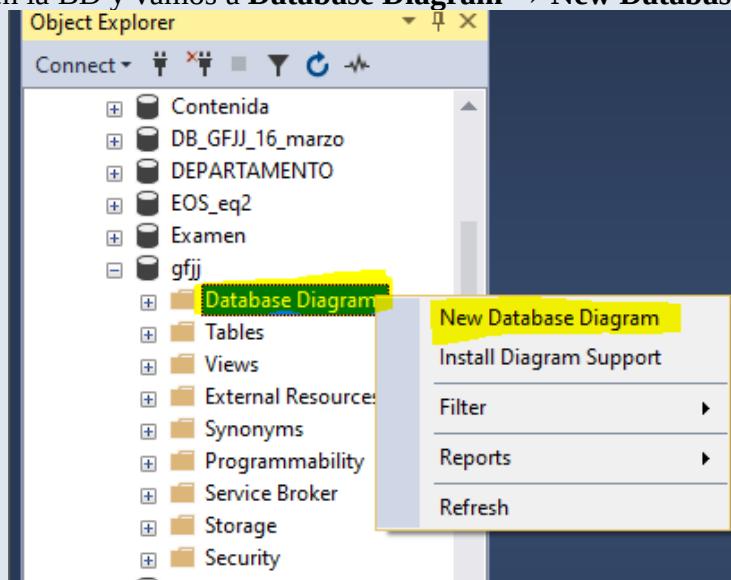
Para cumplir el RGPD, lo primero que tenemos que tener claro es **cuántos equipos/servidores de SQL Server** tenemos. En nuestro caso tenemos un servidor y un equipo, así que no tiene ciencia/misterio. En una empresa mediana o grande tendríamos que llevar un registro/control de los equipos que usamos. Se podría utilizar la herramienta de [Microsoft MAP](#), que permite inventariar y registrar los equipos que estamos utilizando, almacenando todos sus datos.

Lo siguiente que tenemos que hacer es **descubrir qué datos personales tengo en las tablas**. Eso lo hemos hecho [anteriormente aquí](#), mediante la clasificación y auditoría de datos en SQL Server.

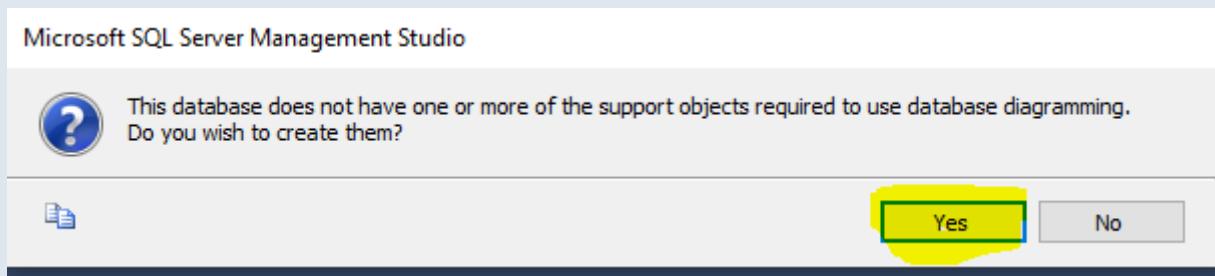
Como vimos en ese momento, nos permite llevar un control de los datos para saber, en caso de robo de información, desde un primer momento verificar qué datos hemos perdido y cuánto de grave puede ser esa filtración.

Si no queremos utilizar la auditoría de SQL Server (aunque la veo interesante), podríamos hacer un SELECT * FROM para cada tabla de la BD, y anotar (en un .doc, o cualquier formato común) el listado de columnas de la tabla (nombre de la columna y lo que representa).

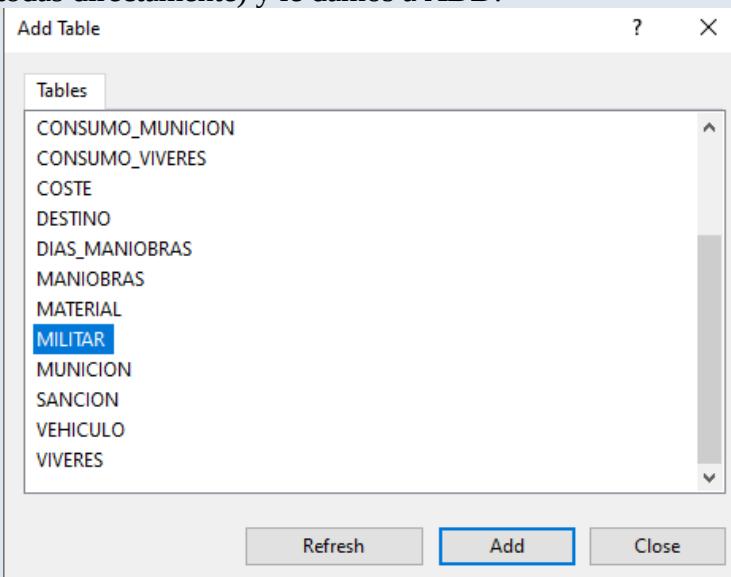
Nos indica tambien que debemos hacer un **diagrama de los flujos de los datos**. Para ello vamos a hacer click derecho en la BD y vamos a **Database Diagram → New Database Diagram**



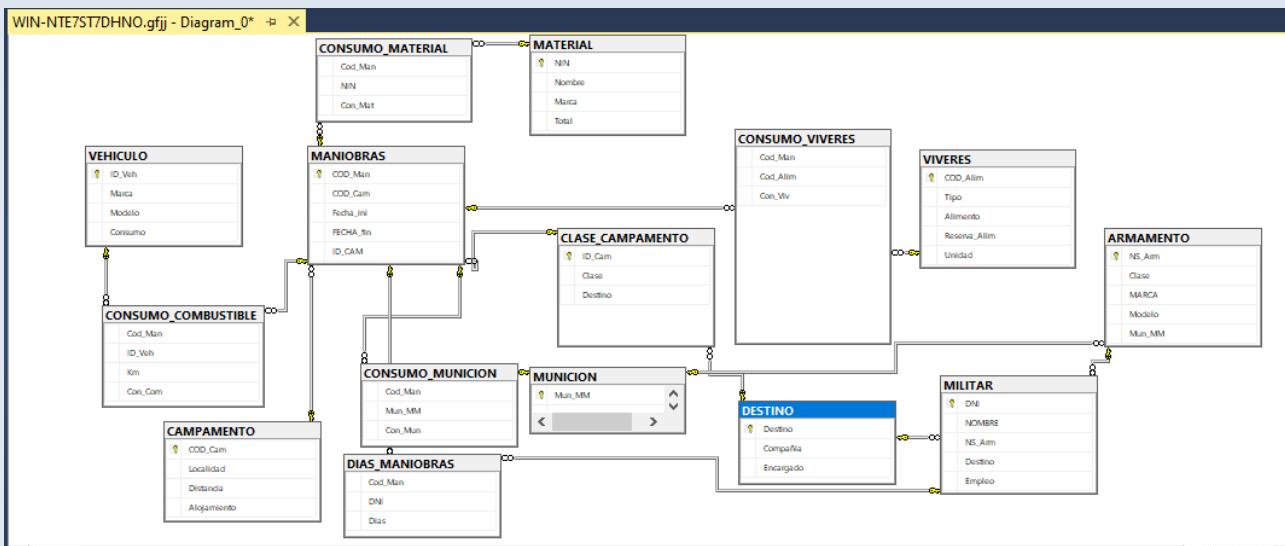
Nos informa que la BD necesita uno o más objetos que no tiene la BD, y si queremos crearlos. Le decimos que **SÍ**.



Ahora tenemos que elegir las Tablas con las que queremos hacer el diagrama. Seleccionamos todas las que queramos (o todas directamente) y le damos a **ADD**.



Tenemos una representación gráfica creada con el propio SQL Server tal que así:



Hay que recordar que siempre es mejor utilizar Data Modeler, como hemos hecho en el anterior proyecto, si bien podemos utilizar los diagramas de SQL Server.

Para cumplir con el RGPD, nos recomiendan en SQL Server deshabilitar las siguientes características:

➤ XP_CMDSHELL

```
sp_configure 'show advanced options', '1'
RECONFIGURE
sp_configure 'xp_cmdshell', '0'
RECONFIGURE
```

➤ CLR (Ya la teníamos desactivada)

➤ Filestream (que la activamos anteriormente)

```
sp_configure 'show advanced options', '1'
RECONFIGURE
sp_configure filestream_access_level, 0;
RECONFIGURE
```

➤ OLE Automation

```
sp_configure 'show advanced options', 1;
RECONFIGURE;
sp_configure 'Ole Automation Procedures', 0;
RECONFIGURE;
```

➤ External Scripts

```
sp_configure 'show advanced options', 1;
RECONFIGURE;
sp_configure 'external scripts enabled', 0;
RECONFIGURE;
```

➤ ADHOC Distributed Queries

```
sp_configure 'show advanced options', 1;
RECONFIGURE;
sp_configure 'Ad Hoc Distributed Queries', 0;
RECONFIGURE;
```

➤ TrustWorthy

```
USE MASTER;
ALTER DATABASE GFJJ SET TRUSTWORTHY OFF;
```

FASE DE GESTIÓN:

Aquí nos recomiendan asegurar la BD con [Vulnerability Assessment](#), como ya hemos realizado. Nos encomiendan a utilizar cifrado de datos, ya sea [RLS](#), [DDM](#) o ambas. Este apartado lo tendríamos hecho anteriormente.

Por último, nos encomiendan a configurar correctamente las cuentas de usuario en las instancias, de modo que se otorguen la menor cantidad posible de permisos (sólo los estrictamente necesarios), así

como evitar el uso de usuarios administrador o con cadena de privilegios.

Por último nos recomiendan:

1. Autenticarse en SQL Server mediante la Autenticación de Windows
2. Usar cuentas separadas para autenticar usuarios y aplicaciones, así como que cada usuario emplee su propia cuenta y no haya cuentas genéricas, para poder auditar cualquier robo/pérdida de información.

FASE DE PROTECCIÓN:

Lo primero que nos recomiendan en la fase de protección es encriptar los datos mediante TLS (Por defecto está implementada en Windows Server 16 así que no tenemos que tocar nada), [TDE](#) y/o [Allways encrypted](#).

Nos recomiendan tambien hacer auditorías de seguridad, tanto a nivel [de servidor](#) como [de BD](#).

En esta fase, debemos evaluar la implementación de un Grupo de disponibilidad Allways On.

Un grupo de disponibilidad admite un entorno replicado de un conjunto discreto de bases de datos de usuario, conocido como bases de datos de disponibilidad. Se puede crear para alta disponibilidad (HA) o para escalado de lectura.

1. Un **grupo de disponibilidad para alta disponibilidad** es un grupo de bases de datos que realizan la commutación por error conjuntamente.
2. Un **grupo de disponibilidad para escalado de lectura** es un grupo de bases de datos que se copian en otras instancias de SQL Server para cargas de trabajo de solo lectura.

Un grupo de disponibilidad admite un conjunto de bases de datos principales y entre uno y ocho conjuntos de las bases de datos secundarias correspondientes.

En [esta guía](#) nos indican cómo se haría un grupo de disponibilidad en SQL Server, si bien en nuestro caso no sería necesario realizarlo. El de alta disponibilidad es MUY importante en una tienda, y a tener en cuenta en cualquier BD que tenga que estar 24 horas funcionando. En nuestro caso, que no vaya la BD implica un café y esperar a que funcione, no existe esa inmediatez.

El grupo de escalado de lectura es interesante para BD que requieran de realizar muchas consultas, que en principio la nuestra sería de consultas locales e individuales y no tendría necesidad.

Una alternativa al grupo de escalado de lectura sería el [Log Shipping](#), que consiste en disponer de al menos dos instancias idénticas y en las que paulatinamente se va restaurando el log de cambios, de modo que los cambios que figuren en uno figuran en otro. La gran ventaja del Log Shipping es que permite tener una instancia para insertar/editar y otra sólo para lectura.

En base a lo que hemos estado implementando a través de las órdenes del Alto Mando, se cumpliría en gran medida con las instrucciones reflejadas en el RGPD. Quedaría reflejar esto realizando un documento de respaldo de copias de seguridad (dónde se guardarán, cada cuanto) así como entregar a los usuarios el documento de consentimiento de tratamiento de datos así como su acceso a la revocación de los mismos.

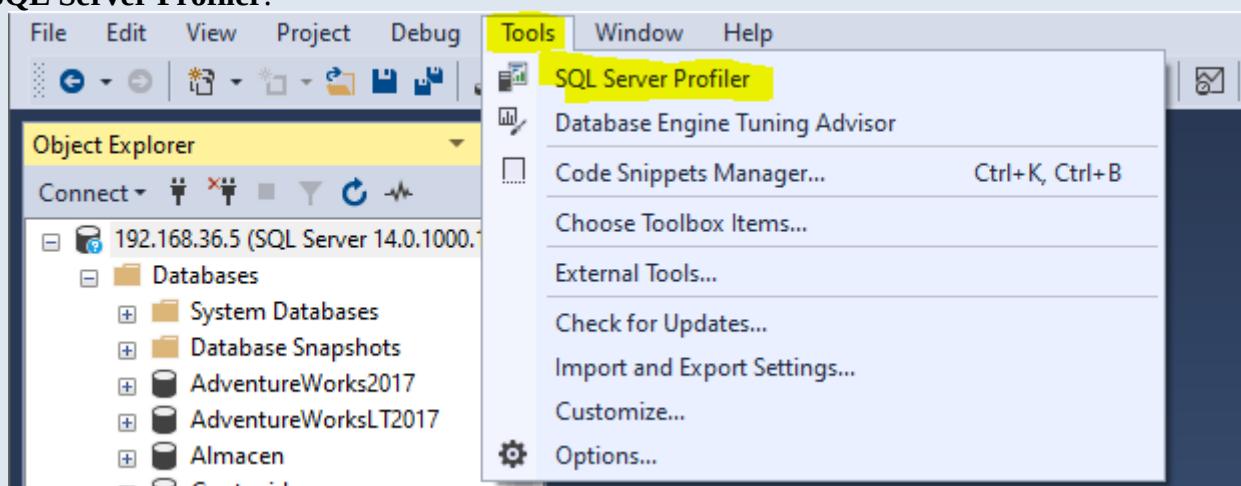
DDOS:

Un ataque DDOS (Denegación del Servicio) tiene como objetivo inhabilitar un servidor, un servicio o una infraestructura. Existen diversas formas de ataque DDoS: por saturación del ancho de banda del servidor para dejarlo inaccesible, o por agotamiento de los recursos del sistema de la máquina, impidiendo así que esta responda al tráfico legítimo.

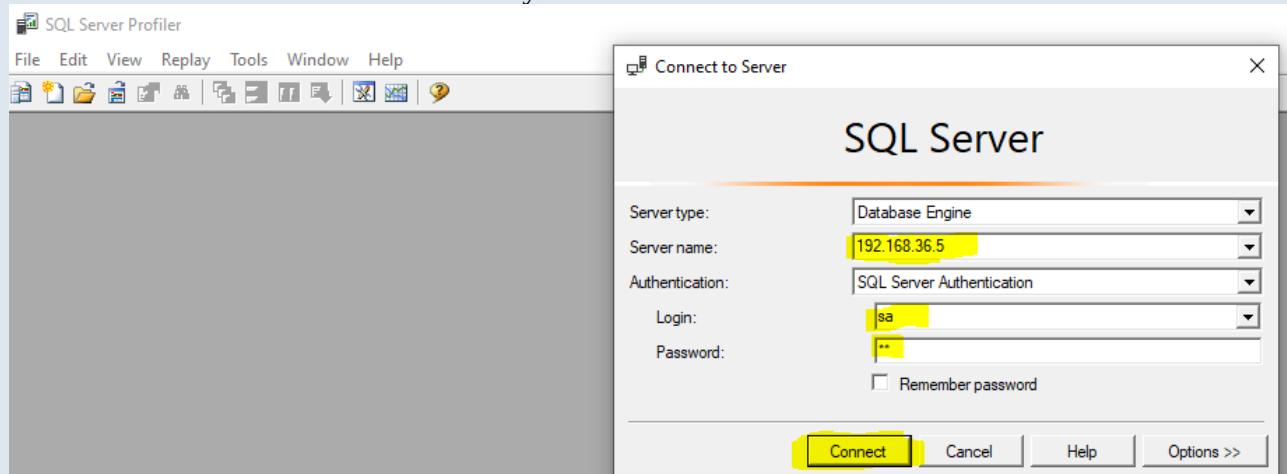
En muchas ocasiones nos encontramos con entornos de producción donde “demasiada gente” tiene acceso. En muchos casos son accesos de “solo lectura” aunque eso no los hace inofensivos de cara a la estabilidad del sistema. Pero podemos ir más allá, vamos a ver como con un login, con rol public, sin acceso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

El Comandante de Seguridad Física nos informa que recientemente la BD quedó deshabilitada temporalmente y no saben el motivo. Desconfiamos de un posible Ataque DDoS , así que nos piden que reflejemos la posibilidad de recibir un ataque DDoS.

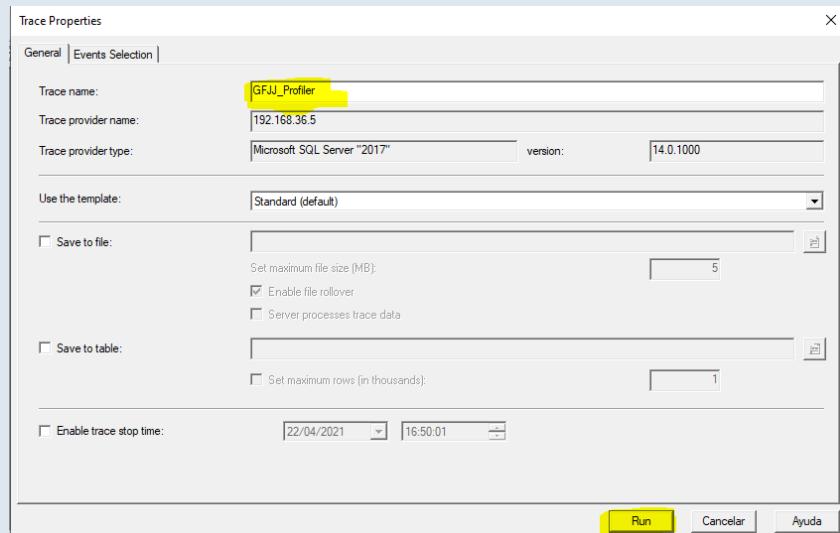
Vamos a utilizar la herramienta de SQL Server Profiler. Para ello vamos a **Tools** y seleccionamos **SQL Server Profiler**.



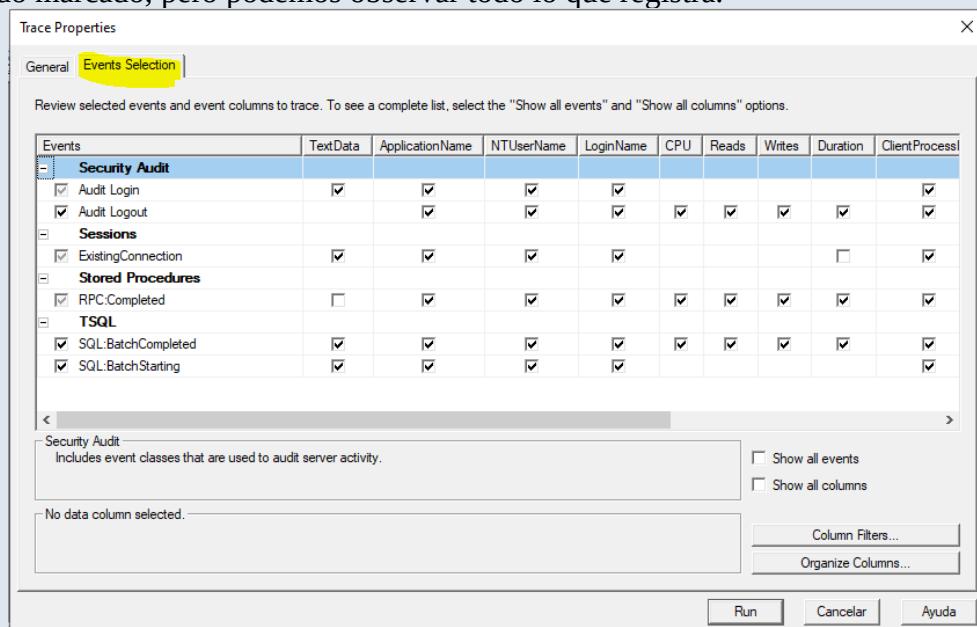
Pasados unos segundos nos abre una instancia nueva a través de la aplicación de SQL Server Profiler. Introducimos los datos de conexión y vamos a **Connect**.



Nos abre una ventana que permite seleccionar el nombre del fichero, así como establecer donde se guardaría la copia (si queremos). Automáticamente nos refleja la versión de SQL Server que tenemos.



Es interesante echar un vistazo antes de eso a la pestaña **Events Selection**. Por defecto viene prácticamente todo marcado, pero podemos observar todo lo que registra.



Vemos que empieza a capturar el tráfico. Me recuerda bastante al programa de Wireshark o cualquier sniffer común de red.

EventClass	TextData	ApplicationName	NTUserName	LoginName	CPU	Reads	Writes	Duration	ClientProcessID	SPID	Start
Trace Start											202
ExistingConnection	-- network protocol: LPC set quote... SQLServerCEIP WIN-NT... ASXBD\...									2756	58 202
ExistingConnection	-- network protocol: TCP/IP set qu... Microsoft SQ... sa									6560	62 202
< ----- >											
-- network protocol: TCP/IP set quoted_identifier on set arithabort off set numeric_roundabort off set ansi_warnings on --											
Trace is running.											

¿Qué ocurre si nos logeamos con una nueva instancia? Vamos a probar a ver. Abrimos otra instancia y volvemos a mirar el Profiler.

Podemos comprobar que registra bastantes movimientos, viene MUY completo.

EventClass	TextData	ApplicationName	NTUserName	LoginName	CPU	Reads	Writes	Duration	ClientProcessID	SPID	Start
Trace Start											20:
ExistingConnection	-- network protocol: LPC set quote... SQLServerCEIP WIN-NTE... ASXBD\...									2756	58 20:
ExistingConnection	-- network protocol: TCP/IP set qu... Microsoft SQ...									6560	62 20:
Audit Login	-- network protocol: TCP/IP set qu... Microsoft SQ...			sa						6560	67 20:
SQL:BatchStarting	DECLARE @edition sysname; SET @edi...	Microsoft SQ...		sa						6560	67 20:
SQL:BatchCompleted	DECLARE @edition sysname; SET @edi...	Microsoft SQ...		sa	0	0	0	0		6560	67 20:
SQL:BatchStarting	IF (@@microsoftversion / 0x01000000...	Microsoft SQ...		sa						6560	67 20:
SQL:BatchCompleted	IF (@@microsoftversion / 0x01000000...	Microsoft SQ...		sa	0	4	0	0		6560	67 20:
SQL:BatchStarting	DECLARE @edition sysname; SET @edi...	Microsoft SQ...		sa						6560	67 20:
SQL:BatchCompleted	DECLARE @edition sysname; SET @edi...	Microsoft SQ...		sa	0	0	0	0		6560	67 20:
Audit Login	-- network protocol: TCP/IP set qu... Microsoft SQ...			sa						6560	52 20:
SQL:BatchStarting	DECLARE @edition sysname; SET @edi...	Microsoft SQ...		sa						6560	52 20:
SQL:BatchCompleted	DECLARE @edition sysname; SET @edi...	Microsoft SQ...		sa	0	0	0	0		6560	52 20:
<											>
use [Contenida];											
if (db_id() = 1)											
begin											
-- contained auth is 0 when connected to master											
select 0											
end											
else											
begin											
-- need dynamic sql so that we compile this query only when we know resource db is available											
<											>
Trace is running.											

Como vemos, profiler es una herramienta perfectamente funcional para registrar cualquier ataque DDoS. No lo vamos a prevenir, pero sí permite monitorizarlo y registrarlo para ver quien lo ha efectuado y reportar a las autoridades competentes inmediatamente.

Una de las formas de ataque más comunes es realizar simultáneamente una cantidad de conexiones superior a la permitida por el servidor. Podemos establecer un límite de conexiones a través de la siguiente instrucción:

```
USE gfjj_archivo ;
GO
EXEC sp_configure 'show advanced options', 1;
GO
RECONFIGURE ;
GO
EXEC sp_configure 'user connections', 512 ;
GO
RECONFIGURE ;
GO
```

Hay que tener mucho cuidado con lo que hacemos, porque si ponemos muy pocas conexiones puede ser contraproducente. Si se supera el máximo de conexiones permitidas, el servidor por precaución se bloquea, de modo que incluso las conexiones que estaban antes del colapso quedan bloqueadas temporalmente, hasta que se reinicie el servicio.

El ataque DDoS, al igual que el Spam, era más común hace unos años, pero es conveniente activar un registro de log de usuarios o bien habilitar herramientas como SQL Profiler para monitorizar la actividad.

Otra opción a tener en cuenta si empleamos una red externa es habilitar en todo momento un firewall. El firewall nos permite tanto deshabilitar múltiples conexiones desde una sola IP, hasta el permitir solamente conexiones con una IP/MAC específicas.

Injection SQL:

La inyección de código SQL es un ataque en el que se inserta código malintencionado en cadenas que posteriormente se pasan a una instancia de SQL Server para su análisis y ejecución. Es uno de los mayores quebraderos de cabeza que tendremos como Administradores de SQL Server.

Todos los procedimientos que generan instrucciones SQL deben revisarse en busca de vulnerabilidades de inyección de código, ya que SQL Server ejecutará todas las consultas recibidas que sean válidas desde el punto de vista sintáctico. Un atacante cualificado y con determinación puede manipular incluso los datos con parámetros.

```
USE MASTER;
CREATE DATABASE GFJJ_Inyector
GO
USE GFJJ_Inyector
GO
DROP TABLE INYEC;
CREATE TABLE INYEC (
ID INT IDENTITY PRIMARY KEY,
Nombre VARCHAR (50) NOT NULL);
GO

DROP PROCEDURE IF EXISTS Nuevo_Inyec;
CREATE PROCEDURE Nuevo_Inyec
    @ejemplo AS VARCHAR (50)
AS
BEGIN
INSERT INTO dbo.INYEC(Nombre) VALUES (@ejemplo);
END;
```

Ahora vamos a ejecutar el procedimiento Almacenado para dar de alta un usuario.

```
EXEC Nuevo_Inyec 'Antonio';
SELECT * FROM INYEC;
```

¿Qué ocurre si metemos en el procedimiento almacenado un trozo de código maliciosamente? Vamos a probar a decirle que elimine la tabla.

```
EXEC Nuevo_Inyec 'Alfredo; drop table INYEC--';
SELECT * FROM INYEC;
```

Podemos observar que ha eliminado la tabla!!! Esto en malas manos sería una tragedia. Tenemos varias formas de evitar que esto ocurra.

Una sería limitando el código, sustituyendo cada carácter que se introduzca por otro que no pueda ser dañino. El problema de esto es que hay bastantes caracteres así que puede ser un engorro modificar todo el código.

El comando SQL para sustituir palabras de una cadena sería REPLACE. Se ejecuta de la siguiente manera:

```
REPLACE ( CADENA_DE_PALABRAS , CARACTER_A_MODIFICAR , CARACTER_QUE_PONEMOS )
```

Un ejemplo de uso en nuestro procedimiento anterior sería:

```
CREATE PROCEDURE Nuevo_Inyec
    @ejemplo AS VARCHAR (50)
AS
BEGIN
INSERT INTO dbo.INYEC(Nombre) VALUES (REPLACE(@ejemplo, ';', '--'));
END;
```

El problema es que es muy complicado de frenar. Porque por ejemplo, si el atacante ve que al poner ; lo cambiamos a --, puede poner un GO en la cadena.

La mejor forma de atajar el problema es asignando los permisos oportunos. En este caso, he ejecuta-

do la cadena desde el usuario DBO, ya que estaba practicando. Vamos a crear un usuario que tenga sólo permiso de lectura e inserción, para probar.

```
DROP USER IF EXISTS atacante
CREATE USER atacante WITHOUT LOGIN;
GO
GRANT SELECT, INSERT ON INYEC TO atacante;
GRANT EXECUTE TO atacante;

EXECUTE AS USER = 'atacante';
EXEC Nuevo_Inyec 'Antonio';
SELECT * FROM INYEC;
```

Vemos que lo hace correctamente. Ahora a ver qué pasa si le metemos el código malicioso:

```
EXECUTE AS USER = 'atacante';
EXEC Nuevo_Inyec 'Antonio; drop table INYEC--';
SELECT * FROM INYEC;
REVERT;
```

Vemos que se ha cargado igualmente la fila! Ahora vamos tanto a revocarle permisos para borrar, como a reemplazar el texto de la cadena:

```
DENY DELETE ON INYEC TO atacante;
CREATE PROCEDURE Nuevo_Inyec
    @ejemplo AS VARCHAR (50)
AS
BEGIN
INSERT INTO dbo.INYEC(Nombre) VALUES (REPLACE(@ejemplo, ';', 'ROLLBACK'));
END;
```

ID	Nombre
1	Antonio
2	Antonio; ROLLBACK drop table INYEC--

Podemos observar que el punto y coma lo sustituye en este caso por la palabra ROLLBACK, de modo que pasa a ser inofensivo.

El lío con esto es que tendríamos que reemplazar absolutamente todo el código que vayamos a insertar. A mayores, habría que reemplazar tanto cada cadena como limitar el tamaño máximo de contenido.

Es fundamental recordar el tema de los privilegios. Si un usuario solamente tiene que hacer lecturas NUNCA le debemos permitir ingresar código.

Ransomware:

El ransomware es el mayor quebradero de cabeza de cualquier Administrador de Sistemas actual. En los últimos años ha afectado a multitud de empresas y gobiernos en mayor o menor medida. Aquí en España tenemos tanto el reciente [ataque al SEPE](#) como el famoso [WannaCry, pertrechado en 2017 a Telefónica](#).

Este software malicioso «secuestra» la información de la empresa, impidiendo el acceso a la misma generalmente cifrándola, y solicitando un rescate (en inglés ransom) a cambio de su liberación. En las empresas causa pérdidas temporales o permanentes de información, interrumpe la actividad normal, ocasiona pérdidas económicas y daños de reputación.

Este tipo de ataque está creciendo de forma exponencial debido a que es muy rentable para los delincuentes:

- a) Cada vez hay más dispositivos «secuestrables».
- b) Es más fácil «secuestrar» la información debido a los avances de la criptografía.
- c) Los ciberdelincuentes pueden ocultar su actividad para lanzar ataques masivos.
- d) Al utilizar sistemas de pago anónimo internacionales es más difícil el seguimiento del delito.

En caso de recibir un ataque de ransomware, lo más importante que nos recuerda las autoridades es NO PAGAR el rescate, debido a que no siempre se garantiza que vayan a desbloquearte el dispositivo. El caso reciente del SEPE no estaba claro si se podría desbloquear, ya que fuentes del INCIBE daban el código por extremadamente aleatorio e indescifrable incluso para los creadores del código, debido a la alta aleatoriedad que se le asignó al cifrado.

En caso de que nuestro organismo reciba un ataque por ransomware, hay que avisar a las autoridades, si bien en España se puede acudir al [INCIBE](#).

Lo más importante en caso de haber sido víctima de un ataque de ransomware, es disponer de una copia de seguridad segura y reciente.

Si hemos recibido un ataque reciente a nuestra BD y nos cifran todo el contenido, pero disponemos de una copia de seguridad de ese mismo día en la que apenas hubo cambios, el daño realizado es mínimo.

En cambio, si no disponemos de una copia de seguridad adecuada el daño [puede ser MUY elevado](#). A mayores, una vez hecha la restauración habría que cambiar todas las contraseñas de forma automática, puesto que si el atacante dispone de las credenciales de acceso, restauramos la información pero no las modificamos, es altamente probable que vuelva a realizar el mismo tipo de ataque contra nosotros.

En el caso que nos atribuye, el Alto Mando muestra su satisfacción por disponer de [copias de seguridad cifradas](#) que permitan recuperar la información en caso de un ataque por ransomware.

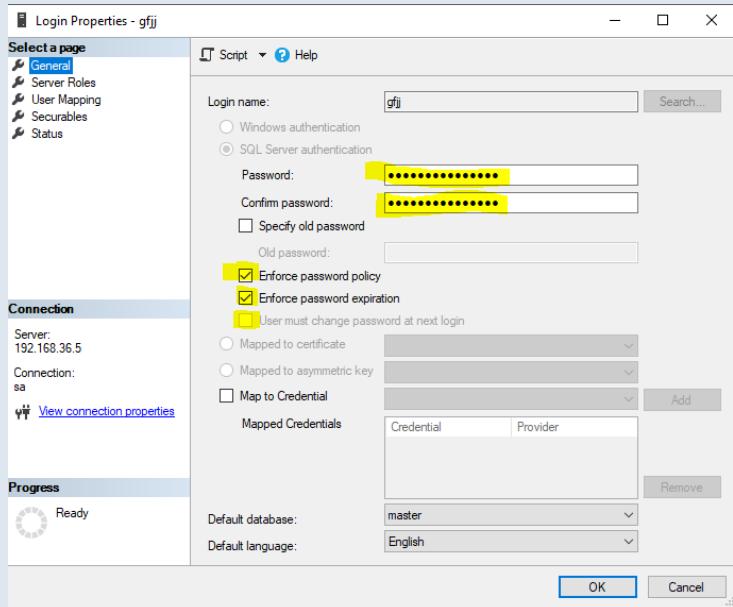
Uno de los apartados clave para evitar un ataque ransomware es evitar realizar la actividad SQL en Internet, y en cambio utilizar una Intranet propia. En el caso de ser indispensable la utilización de servicios de Internet (por ejemplo, una tienda/negocio online), es altamente recomendable conectarse al servicio a través de una VPN.

Una VPN nos permite tanto preservar el anonimato de forma segura (básicamente tú te conectas a un servidor, envías esa consulta y la devolución que llega al servidor te la retransmite a tí) como enmascarar nuestra red, de modo que cualquier ataque que se intente hacer por este medio, se lanza contra el propio servidor.

Lo idóneo como empresa es utilizar una VPN segura de pago, si bien eso requiere suscripción mensual. La alternativa gratuita podría ser [HotSpot](#), si bien para una empresa es altamente recomendable escoger una solución de pago.

Por último, se recomienda escoger contraseñas de usuario seguras. Para ello lo mejor es establecer una política de contraseñas seguras en SQL Server.

Tambien es recomendable cambiar periódicamente las contraseñas de los usuarios para minimizar la posibilidad de daños en caso de pérdida.



Es fundamental recalcar que aparte de gestionar los backups, debemos de tener una política de backups segura y confiable. Se puede automatizar la tarea, pero es fundamental disponer de la copia separada del servidor, puesto que si se cifra toda la información del servidor y tenemos la copia en el mismo equipo, realmente no tenemos copia.

Herramientas:

Herramientas para bloquear ataques DDoS

Los ataques DDoS se efectúan en gran medida a través de Internet. Es muy complicado que se efectúen en una Intranet. Podría ser efectuado por un usuario ejecutando un script que haga un bucle de conexiones, pero es poco habitual.

Lo que se recomienda si queremos de una conexión a Internet (un negocio, etc..) sería:

1. Aumentar la capacidad de entrada del servidor. A una mayor capacidad de conexiones, menor probabilidad de un ataque DDoS, si bien esto implica un coste.
2. Utilizar un firewall y/o bien utilizar ACL (Listas de Control de Acceso).

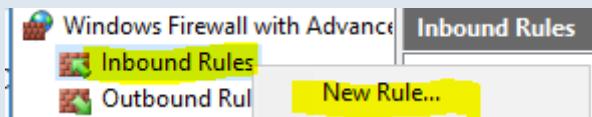
Se puede configurar el propio Firewall de Windows para estas tareas, si bien no es descartable emplear firewalls de red específicos. En un negocio/empresa mediana, lo idóneo en mi opinión sería poner un Bastión, es decir, un equipo que solamente esté para recibir el tráfico entrante/saliente y que permita/deniegue el tráfico sospechoso. Hay que recordar que el propio router de casa se puede configurar para permitir/bloquear el tráfico mediante ACLs.

Si el negocio es grande y queremos de alta disponibilidad, se le puede delegar esta tarea a un tercero. Amazon (entre otras) dispone de un servicio [ACL de red](#).

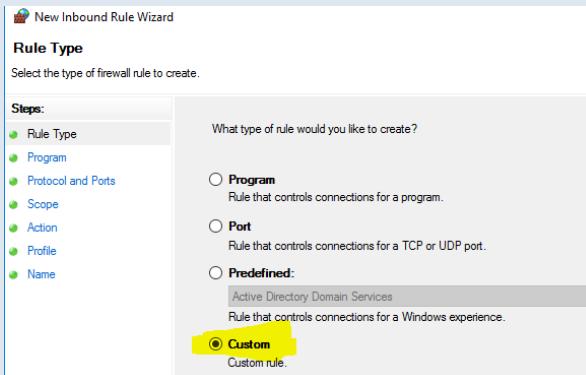
Otra recomendación sería deshabilitar el PING entrante en el servidor, ya que es una de las formas más sencillas de ataque.

Para deshabilitarlo en el Firewall de Windows abrimos el Firewall:

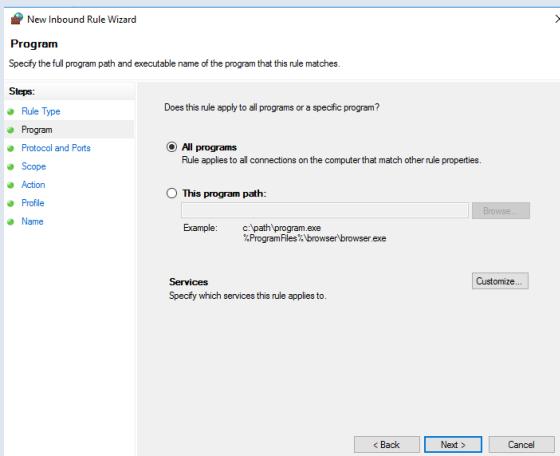
1. Hacemos click derecho en reglas entrantes y le damos a nueva Regla.



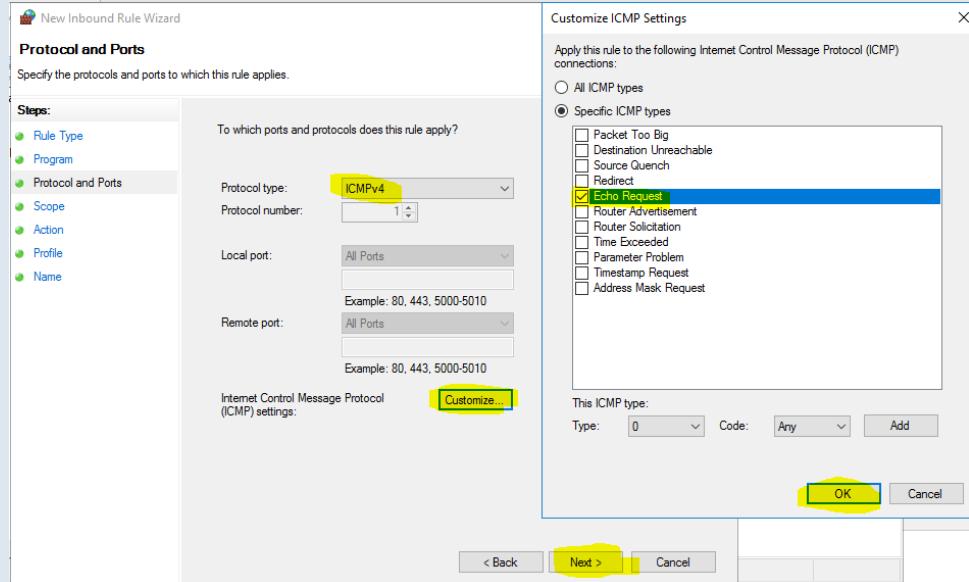
2. Regla personalizada.



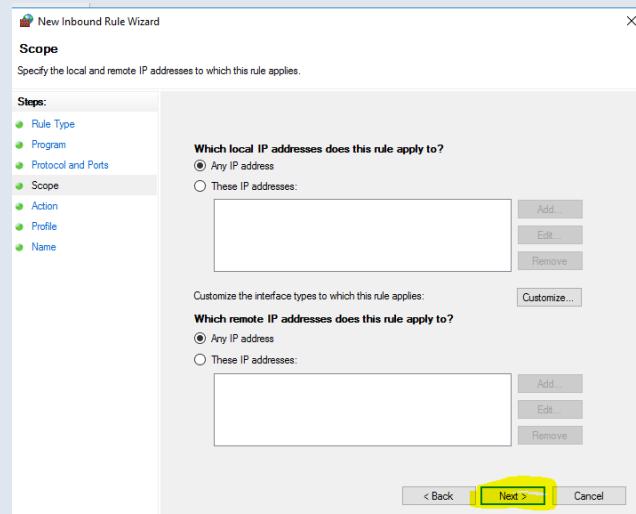
3. Dejamos por defecto y le damos a Next.



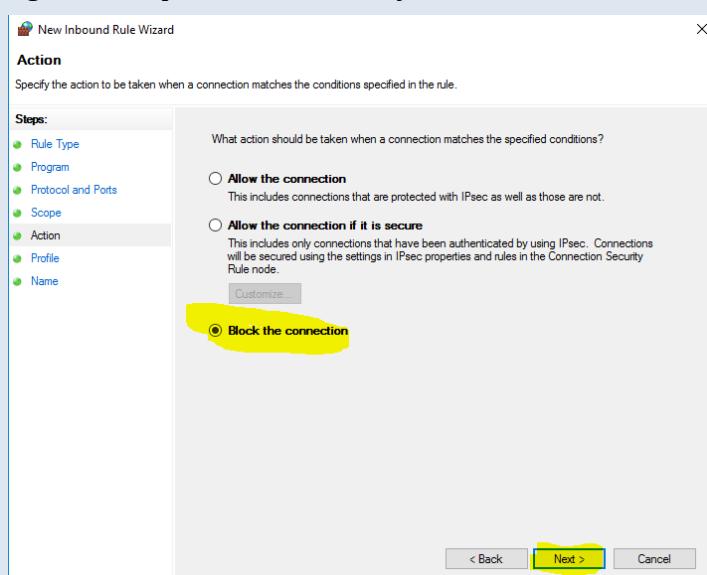
4. Ahora elegimos en el tipo de protocolo ICMPv4, y le damos a Personalizar y especificamos el servicio de petición eco. Le damos a **Ok** y **Next**.



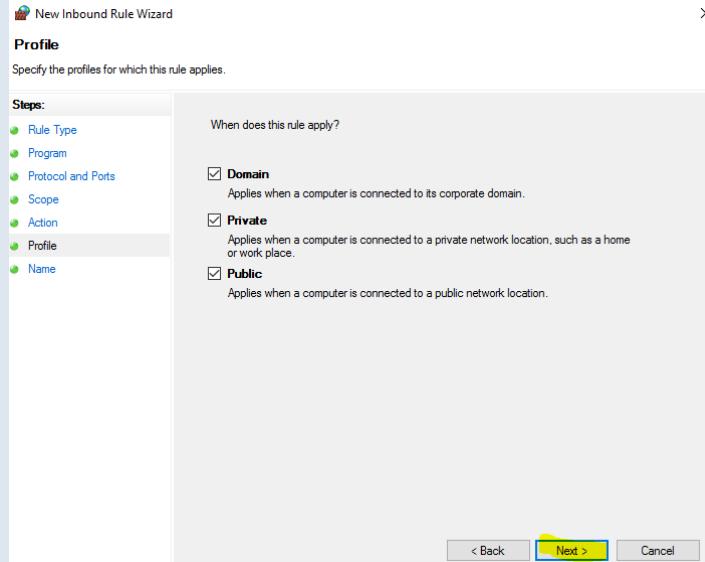
5. Aquí podemos elegir a quién aplicamos la regla. En principio a todo el mundo, luego podríamos hacer otra regla que permita a una IP concreta, si necesitamos que un admin le haga pin desde un equipo.



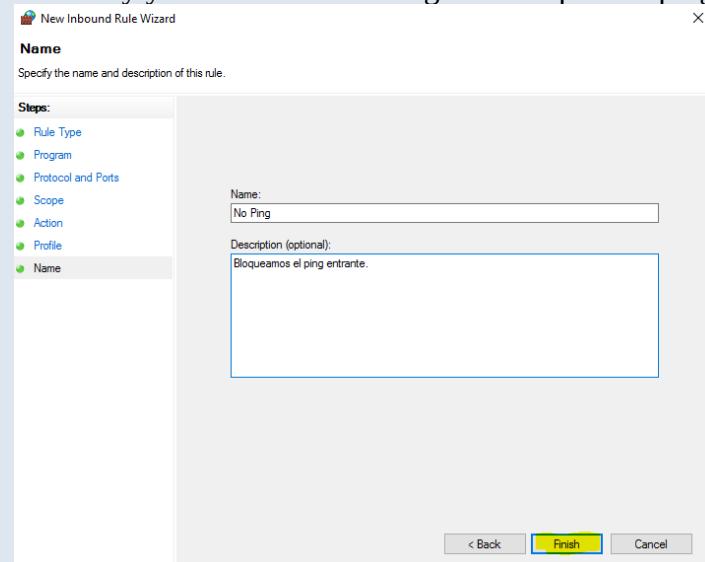
6. Ahora elegimos bloquear la conexión, y le damos a **Next**.



7. Dejamos todo por defecto (las 3 opciones marcadas) y le damos a **Next**.



8. Le ponemos el nombre y ya tenemos nuestra regla de bloqueo de ping creada.



9. Lo probamos para verificar que funciona.

```
Símbolo del sistema - ping 192.168.36.5
Microsoft Windows [Versión 10.0.18362.30]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\gfjj.ASXBD>ipconfig

Configuración IP de Windows

Adaptador de Ethernet Ethernet0:

    Sufijo DNS específico para la conexión. . . :
    Vínculo: dirección IPv6 local. . . : fe80::e1aa:1e8d:6aa8:5b5c%11
    Dirección IPv4. . . . . : 192.168.36.10
    Máscara de subred . . . . . : 255.255.255.0
    Puerta de enlace predeterminada . . . . : 192.168.36.5

C:\Users\gfjj.ASXBD>ping 192.168.36.5

Haciendo ping a 192.168.36.5 con 32 bytes de datos:
Tiempo de espera agotado para esta solicitud.
```

Otra recomendación sería utilizar una VPN, para evitar Ataques DDoS. Hay servicios gratuitos, pero para una empresa no serían buena opción.

Las mejores opciones según los expertos sería utilizar una VPN de [NordVPN](#), la de [Surfshark](#) o [ExpressVPN](#).

Herramientas para bloquear ataques SQL Injection

Lo que se recomienda encarecidamente es que todos los procesos almacenados que reciban instrucciones, contengan un REPLACE como [hemos hecho](#), pero vienen una serie de indicaciones de código bastante grandes. Podemos ver las recomendaciones en la [página oficial de Microsoft](#).

He encontrado que una solución a SQL Injection podría ser utilizar [Apex SQL Formater](#). Es una herramienta que nos permite encapsular el código, así como filtrarlo.

Soluciones contra ataques Ransomware

Si bien hay [alguna herramienta](#) que nos podría intentar descifrar el código del ransomware y recuperar nuestra BD... No es infalible, aparte de que va a tardar bastante tiempo.

Lo ideal es tener una copia de seguridad siempre a mano y segura. Es fundamental que la copia de seguridad no esté compartiendo disco en el servidor permanentemente, puesto que si nos cifran el contenido del disco, y la copia está dentro...

Para ello podemos recurrir a herramientas como [SQL Backup PRO](#), que nos realiza la copia directamente en la red. Una alternativa a esto sería gestionar las copias en red.

Para ello lo primero que tenemos que hacer es tener acceso desde el servidor a la unidad de red. Hecho esto, tendríamos que habilitar el servicio (que antes deshabilitamos porque no necesitábamos) xp_cmdshell. Sería interesante en el procedimiento almacenado del backup, habilitar el xp_cmdshell y posteriormente deshabilitarlo.

```
EXEC sp_configure 'show advanced options', 1
GO
RECONFIGURE
GO
EXEC sp_configure 'xp_cmdshell', 1
GO
RECONFIGURE
GO
```

Ahora ejecutariamos este trozo de código:

```
SET LANGUAGE us_english
exec xp_cmdshell 'net use \IPALOJAMIENTO contraseña /user:backup'
DECLARE @Archivo AS nvarchar(100)
SET @Archivo = N'\IPALOJAMIENTO' + DATENAME(WEEKDAY, GETDATE()) + '.bak'
BACKUP DATABASE [GFJJ] TO DISK = @Archivo WITH NOFORMAT, INIT,
      NAME = N'GFJJ-Full Database Backup', SKIP, NOREWIND, NOUNLOAD,  STATS = 10;
exec xp_cmdshell 'net use \IPALOJAMIENTO /D'
```

En donde pone IPALOJAMIENTO pondríamos la dirección web donde vamos a subir el backup.

En contraseña obviamente la contraseña que tengamos en el sitio web.

Es importante recordar que el servicio xp_cmdshell nos piden deshabilitarlo, de modo que un procedimiento almacenado de backup tendría que habilitarlo primero y luego deshabilitarlo.

DOCKER:

Qué es Docker:

Docker es una herramienta que permite empaquetar una aplicación y sus dependencias en un contenedor muy ligero. Es como si tomaras una aplicación completa con absolutamente todo lo que necesita para funcionar para poder transportarla sin problema a cualquier otro servidor con Docker instalado, ya sea para seguir desarrollándola o para hacer deploy.

Docker nos permite correr nuestras aplicaciones en contenedores, cada una con su propia función, sistema operativo y recursos. Técnicamente es muy similar a emplear la virtualización por medio de máquinas virtuales, pero tiene ciertas diferencias.

- **Máquina virtual**

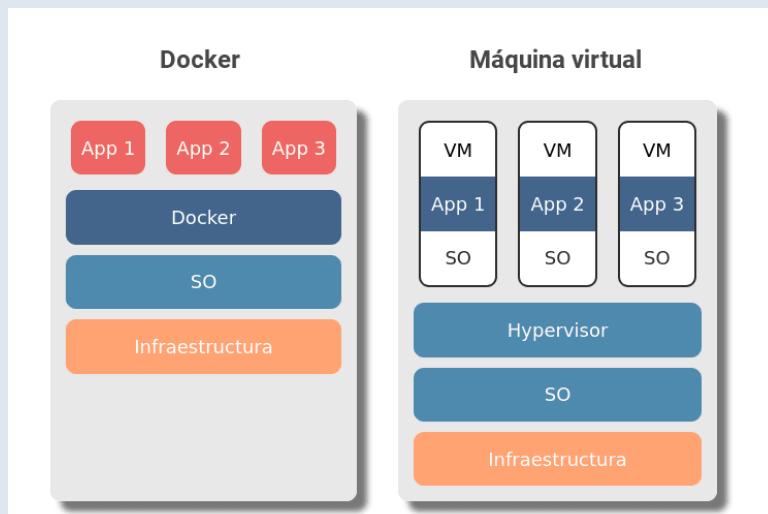
Las máquinas virtuales (VM en singular o VMs en plural) son una abstracción del hardware físico que convierten un servidor en muchos servidores. El monitor de las VMs permite que múltiples máquinas virtuales corran en una única máquina. Cada VM incluye una copia completa de un sistema operativo, la aplicación, librerías y binarios necesarios - tomando decenas de GB. Las VMs también pueden ser lentas de arrancar.

Las máquinas virtuales, en comparación con los contenedores, ocupan mucho espacio y, como deben cargar un sistema operativo completo, pueden demorar mucho más tiempo en arrancar y, a diferencia de los contenedores, virtualizan el funcionamiento de la parte del hardware.

- **Contenedores**

Los contenedores son una abstracción en la capa de aplicación que empaqueta el código y sus dependencias. Múltiples contenedores pueden correr en la misma máquina y compartir el kernel del sistema operativo con otros contenedores, cada uno corriendo como un proceso aislado en el entorno del usuario. Los contenedores ocupan menos espacio que las VMs (las imágenes de los contenedores típicamente tienen tamaños de decenas de MB), pueden manejar más aplicaciones y requieren menos VMs y sistemas operativos.

Los contenedores son muy ligeros, se encuentran aislados y virtualizan el funcionamiento de un sistema operativo.



En la primera imagen podemos apreciar que, en el caso de los contenedores, las aplicaciones interactúan directamente con Docker, y este, a su vez, con el sistema operativo.

En la segunda imagen tenemos varias aplicaciones, cada aplicación corriendo sobre su propio sistema operativo, así es, tres sistemas operativos completos monopolizando recursos. Bajo esas máquinas virtuales, el software encargado de crearlas y ejecutarlas (Hypervisor) interacciona con el sistema operativo.

Comandos Básicos:

Vamos a listar una serie de comandos básicos y lo que hacen:

COMANDOS INFORMATIVOS:

- **docker ps** muestra los contenedores en ejecución. Con ps-a vemos tambien los detenidos
- **docker logs** obtiene logs de un contenedor.
- **docker inspect** observa toda la información en un contenedor.
- **docker events** obtiene eventos de un contenedor.
- **docker port** muestra el puerto publico de un contenedor.
- **docker top** muestra los procesos corriendo en un contenedor.
- **docker stats** muestra las estadísticas de recursos usados por contenedor. Con --all muestra una lista de los contenedores que están en ejecución.
- **docker diff** muestra los archivos cambiados en el FS del contenedor.
- **docker history** muestra el historial de una imagen.
- **docker tag** etiqueta una imagen a un nombre asignado.

COMANDOS CON INSTRUCCIONES A CONTENEDORES:

- **docker create** crea un contenedor pero no lo comienza.
- **docker rename** permite renombrar al contenedor.
- **docker run** crea y comienza un contenedor en una operación.
- **docker rm** borra un contenedor.
- **docker update** actualiza los recursos limitados de un contenedor.
- **docker start** comienza un contenedor si se cayó o salió.
- **docker stop** detiene un contenedor.
- **docker restart** detiene y comienza un contenedor.
- **docker pause** pausa un contenedor corriendo.
- **docker unpause** quita la pausa de un contenedor corriendo.
- **docker wait** bloquea hasta que un contenedor corriendo se detiene.
- **docker kill** envía una SIGKILL a un contenedor corriendo.
- **docker attach** se conecta a un contenedor corriendo.
- **docker images** muestra todas las imágenes.
- **docker import** crea una imagen de un tarball.
- **docker build** crea imagen de un Dockerfile.
- **docker commit** crea imagen de un contenedor, pausándolo temporalmente si esta corriendo.
- **docker rmi** remueve una imagen.
- **docker load** carga una imagen de un archivo tar como STDIN, incluyendo imagenes y etiquetas.
- **docker save** salva una imagen a un archivo tar a STDOUT con todas las capas padre, etiquetas y versiones.

INSTALANDO DOCKER:

Para instalar Docker, primero hacemos un apt-get update.

```
gfjj@ubuntu:~$ sudo apt-get update
```

Luego apt-get install curl.

```
gfjj@ubuntu:~$ sudo apt-get install curl
```

Ahora con curl cogemos la imagen de docker.

```
curl -fsSL https://get.docker.com -o get-docker.sh
```

```
gfjj@ubuntu:~$ curl -fsSL https://get.docker.com -o get-docker.sh
```

Instalamos docker con:

```
sh get-docker.sh
```

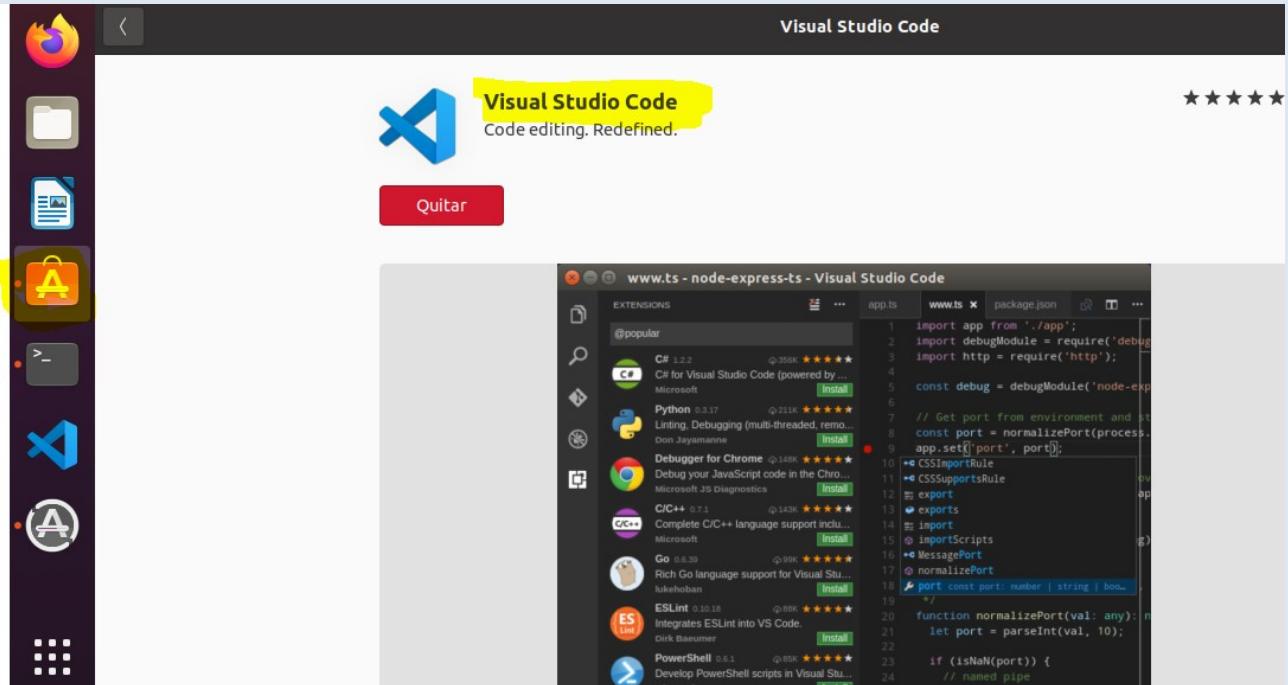
```
gfjj@ubuntu:~$ sh get-docker.sh
```

Ahora le asignamos permisos con usermod -aG docker nombre

```
gfjj@ubuntu:~$ sudo usermod -aG docker gfjj
```

```
gfjj@ubuntu:~$ sudo groups gfjj
```

Ahora tenemos que instalar VisualCode. En mi caso voy a optar por instalarlo utilizando el Software oficial de Ubuntu. Desde el menú de Ubuntu Software, con poner Visual Studio nos sale el oficial, que podemos instalar con un sólo click.



Tambien se puede instalar desde terminal, poniendo:

```
sudo snap install --classic code
```

```
gfjj@ubuntu:~$ sudo snap install --classic code
```

COMANDOS BÁSICOS:

Vamos a probar los comandos que vimos en la teoría. Primero nos ponemos en modo superusuario, ya que hay varias instrucciones que nos pueden dar error si no lo estamos. Hacemos un sudo su

```
gfjj@ubuntu:~$ sudo su  
[sudo] contraseña para gfjj:  
root@ubuntu:/home/gfjj#
```

Vamos a empezar cogiendo una imagen, para tener alguna. Vamos a coger la oficial de Ubuntu

```
root@ubuntu:/home/gfjj# docker pull ubuntu  
Using default tag: latest  
latest: Pulling from library/ubuntu  
Digest: sha256:adf73ca014822ad8237623d388cedf4d5346aa72c270c5acc01431cc93e18e2d  
Status: Image is up to date for ubuntu:latest  
docker.io/library/ubuntu:latest  
root@ubuntu:/home/gfjj#
```

Ahora vamos a comprobar la versión de Docker que tenemos.

```
root@ubuntu:/home/gfjj# docker --version  
Docker version 20.10.6, build 370c289  
root@ubuntu:/home/gfjj#
```

Vamos a arrancar el contenedor de hello-world. Ese contenedor está disponible para todo el mundo, precisamente para practicar.

```
root@ubuntu:/home/gfjj# docker run hello-world  
Unable to find image 'hello-world:latest' locally  
latest: Pulling from library/hello-world  
b8dfde127a29: Pull complete  
Digest: sha256:5122f6204b6a3596e048758cabba3c46b1c937a46b5be6225b835d091b90e46c  
Status: Downloaded newer image for hello-world:latest  
  
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/  
  
root@ubuntu:/home/gfjj#
```

Tarda unos segundos en cargar la imagen, pero vemos que nos ha cargado el contenedor.

Vamos a comprobar en qué estado está cada contenedor, con un docker ps.

```
root@ubuntu:/home/gfjj# docker ps  
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
```

Vemos que no nos muestra ninguna información relevante. Ahora vamos a ejecutar tambien el con-

tenedor de Ubuntu y vamos a comprobar el estado con un ps -a.

```
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID   IMAGE      COMMAND   CREATED      STATUS      PORTS     NAMES
4a8e76ae5895   ubuntu    "/bin/bash" 7 seconds ago  Exited (0) 6 seconds ago
9a3226c57655   hello-world  "/hello"   3 minutes ago  Exited (0) 3 minutes ago
root@ubuntu:/home/gfjj#
```

Como podemos observar, nos muestra ambas imágenes. Vamos a observar la información del contenedor hello-world. Nos viene toda la información generada por el contenedor, incluyendo metadatos e información de contenido.

```
root@ubuntu:/home/gfjj
{
  "Volumes": null,
  "WorkingDir": "",
  "Entrypoint": null,
  "OnBuild": null,
  "Labels": null
},
"Architecture": "amd64",
"Os": "linux",
"Size": 13336,
"VirtualSize": 13336,
"GraphDriver": {
  "Data": {
    "MergedDir": "/var/lib/docker/overlay2/23bf8a50d65c1ecd9a26dbbbb772ddd5146191fbeec3f4d6e5df3d4a42d1bab/merged",
    "UpperDir": "/var/lib/docker/overlay2/23bf8a50d65c1ecd9a26dbbbb772ddd5146191fbeec3f4d6e5df3d4a42d1bab/diff",
    "WorkDir": "/var/lib/docker/overlay2/23bf8a50d65c1ecd9a26dbbbb772ddd5146191fbeec3f4d6e5df3d4a42d1bab/work"
  },
  "Name": "overlay2"
},
"RootFS": {
  "Type": "layers",
  "Layers": [
    "sha256:f22b99068db93900abe17f7f5e09ec775c2826ecfe9db961fea68293744144bd"
  ]
},
"Metadata": {
  "LastTagTime": "0001-01-01T00:00:00Z"
}
}
]
root@ubuntu:/home/gfjj# docker inspect hello-world
```

Ahora vamos a crear un contenedor de mysql. Ponemos el siguiente ladrillo en Terminal.

```
docker run -d \
--rm \
--name mysqlc \
-e MYSQL_ROOT_PASSWORD=Abcd1234. \
-p 3306:3306 \
mysql
```

```
root@ubuntu:/home/gfjj# docker run -d \
> --rm \
> --name mysqlc \
> -e MYSQL_ROOT_PASSWORD=Abcd1234. \
> -p 3306:3306 \
> mysql
348d6e76c142fd4b4d2fa863cb44ec898e454b17d305e75708df203c7451c8c0
```

Creamos el volumen con la instrucción:

```
docker volume create mysql_data
```

```
root@ubuntu:/home/gfjj# docker volume create mysql_data
mysql_data
```

Lanzamos el contenedor con la siguiente instrucción:

```
docker run -d --name mysqlc -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 -v mysql_data:/var/lib/mysql mysql
```

```
root@ubuntu:/home/gfjj# docker run -d --name mysqlc -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 -v mysql_data:/var/lib/mysql mysql
da1c4219fd218ae02cda19e990d353cc2d59f8b7c43a9db57920ad9f28e0e0b3
```

Ahora hacemos tanto un docker ps como un ps -a y comprobamos que el contenedor está activo.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAME
da1c4219fd21	mysql	"docker-entrypoint.s..."	24 seconds ago	Up 24 seconds	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp	mysqlc
root@ubuntu:/home/gfjjj#	docker ps -a					
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	
da1c4219fd21	mysql	"docker-entrypoint.s..."	32 seconds ago	Up 32 seconds	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp,	
33060/tcp	mysqlc					
4a8e76ae5895	ubuntu	"/bin/bash"	12 minutes ago	Exited (0) 12 minutes ago		
ecstatic_bell						
9a3226c57655	hello-world	"/hello"	15 minutes ago	Exited (0) 15 minutes ago		
competent_yallow						
root@ubuntu:/home/gfjjj#						

Lo pausamos con la instrucción pause.

```
root@ubuntu:/home/gfjjj# docker pause mysqlc  
mysqlc
```

Comprobamos que está parado:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAME
da1c4219fd21	mysql	"docker-entrypoint.s..."	6 minutes ago	Up 6 minutes (Paused)	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp,	
33060/tcp	mysqlc					
4a8e76ae5895	ubuntu	"/bin/bash"	18 minutes ago	Exited (0) 18 minutes ago		
ecstatic_bell						
9a3226c57655	hello-world	"/hello"	21 minutes ago	Exited (0) 21 minutes ago		
competent_yallow						
root@ubuntu:/home/gfjjj#	docker ps -a					
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAME
da1c4219fd21	mysql	"docker-entrypoint.s..."	6 minutes ago	Up 6 minutes (Paused)	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp,	
mysqlc						
root@ubuntu:/home/gfjjj#						

Le quitamos la pausa con el comando unpause.

```
root@ubuntu:/home/gfjjj# docker unpause mysqlc  
mysqlc
```

Comprobamos que vuelve a estar activo:

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAME
da1c4219fd21	mysql	"docker-entrypoint.s..."	8 minutes ago	Up 8 minutes	0.0.0.0:3306->3306/tcp, :::3306->3306/tcp,	
33060/tcp	mysqlc					
4a8e76ae5895	ubuntu	"/bin/bash"	19 minutes ago	Exited (0) 19 minutes ago		
ecstatic_bell						
9a3226c57655	hello-world	"/hello"	23 minutes ago	Exited (0) 23 minutes ago		
competent_yallow						
root@ubuntu:/home/gfjjj#						

Vamos a cerrarlo con la instrucción kill. Esto mataría el proceso.

```
root@ubuntu:/home/gfjjj# docker kill mysqlc  
mysqlc
```

Podemos observar al hacer un ps -a que nos indica que fue finalizado mediante un código 137, que si vamos al manual de docker nos indica que fue con la instrucción SIGKILL, que es la que hace el comando Kill.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAME
da1c4219fd21	mysql	"docker-entrypoint.s..."	9 minutes ago	Exited (137) 2 seconds ago		mysqlc
4a8e76ae5895	ubuntu	"/bin/bash"	21 minutes ago	Exited (0) 21 minutes ago		ecstatic_bell
9a3226c57655	hello-world	"/hello"	24 minutes ago	Exited (0) 24 minutes ago		competent_yallow
root@ubuntu:/home/gfjjj#						

Lo volvemos a arrancar con el comando Start.

```
root@ubuntu:/home/gfjjj# docker start mysqlc  
mysqlc
```

Comprobamos que está activo:

```
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
NAMES
da1c4219fd21 mysql "docker-entrypoint.s..." 12 minutes ago Up 1 second 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp,
33060/tcp mysqlc
4a8e76ae5895 ubuntu "/bin/bash" 23 minutes ago Exited (0) 23 minutes ago
ecstatic_bell
9a3226c57655 hello-world "/hello" 27 minutes ago Exited (0) 27 minutes ago
competent_yallow
root@ubuntu:/home/gfjj#
```

Ahora vamos a cambiarle el nombre con el comando docker rename.

```
root@ubuntu:/home/gfjj# docker rename mysqlc gjff_mysql
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
NAMES
da1c4219fd21 mysql "docker-entrypoint.s..." 14 minutes ago Up About a minute 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp,
33060/tcp gjff_mysql
4a8e76ae5895 ubuntu "/bin/bash" 25 minutes ago Exited (0) 25 minutes ago
ecstatic_bell
9a3226c57655 hello-world "/hello" 29 minutes ago Exited (0) 29 minutes ago
root@ubuntu:/home/gfjj#
```

Con docker images podemos comprobar todas las imágenes que tenemos:

```
root@ubuntu:/home/gfjj# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
mysql latest c0cdc95609f1 2 weeks ago 556MB
ubuntu latest 7e0aa2d69a15 5 weeks ago 72.7MB
hello-world latest d1165f221234 2 months ago 13.3kB
root@ubuntu:/home/gfjj#
```

Intentamos cargarnos una imagen, pero no nos lo permite porque está en ejecución.

```
root@ubuntu:/home/gfjj# docker rmi hello-world
Error response from daemon: conflict: unable to remove repository reference "hello-world" (must force) - container 9a3226c57655 is using its referenced image d1165f221234
```

Con el comando docker logs podemos observar información de cuando se ejecutó la imagen

```
root@ubuntu:/home/gfjj# docker logs gjff_mysql
2021-05-30 14:56:17+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.25-1debian10 started.
2021-05-30 14:56:17+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2021-05-30 14:56:17+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.25-1debian10 started.
2021-05-30T14:56:17.997402Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.25) starting as process 1
2021-05-30T14:56:18.024188Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-05-30T14:56:18.474523Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-05-30T14:56:18.785989Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysql.sock
2021-05-30T14:56:18.915302Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2021-05-30T14:56:18.915589Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are now supported for this channel.
2021-05-30T14:56:18.919711Z 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.
2021-05-30T14:56:18.995678Z 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.
2021-05-30 15:08:29+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.25-1debian10 started.
2021-05-30 15:08:29+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2021-05-30 15:08:29+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.25-1debian10 started.
2021-05-30T15:08:29.965093Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.25) starting as process 1
2021-05-30T15:08:29.974441Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2021-05-30T15:08:30.253055Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2021-05-30T15:08:30.371328Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysql.sock
2021-05-30T15:08:30.389393Z 0 [System] [MY-010229] [Server] Starting XA crash recovery...
2021-05-30T15:08:30.400656Z 0 [System] [MY-010232] [Server] XA crash recovery finished.
2021-05-30T15:08:30.438991Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2021-05-30T15:08:30.439180Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are now supported
```

Al hacer un docker stats -all nos muestra la siguiente información detallada del consumo de recursos:

```
root@ubuntu:/home/gfjj
CONTAINER ID NAME CPU % MEM USAGE / LIMIT MEM % NET I/O BLOCK I/O PIDS
a1c4219fd21 gjff_mysql 0.58% 339.5MiB / 3.815GiB 8.69% 3.7kB / 0B 508kB / 13.6MB 37
a8e76ae5895 ecstatic_bell 0.00% 0B / 0B 0.00% 0B / 0B 0B / 0B 0
a3226c57655 competent_yallow 0.00% 0B / 0B 0.00% 0B / 0B 0B / 0B 0
```

Podemos ver el historial de cambios que tuvo la imagen con la instrucción docker history mysql. Esto es altamente interesante cuando se trabaja en una organización y son varias personas trabajando en el mismo proyecto.

```

root@ubuntu:/home/gfjj# docker history mysql
IMAGE      CREATED     CREATED BY
c0cdc95609f1 2 weeks ago /bin/sh -c #(nop)  CMD ["mysqld"]
<missing> 2 weeks ago /bin/sh -c #(nop) EXPOSE 3306 33060
<missing> 2 weeks ago /bin/sh -c #(nop) ENTRYPOINT ["docker-entryp...
<missing> 2 weeks ago /bin/sh -c ln -s usr/local/bin/docker-entryp...
<missing> 2 weeks ago /bin/sh -c #(nop) COPY file:345a22fe55d3e678...
<missing> 2 weeks ago /bin/sh -c #(nop) COPY dir:2e040acc386ebd23b...
<missing> 2 weeks ago /bin/sh -c #(nop) VOLUME [/var/lib/mysql]
<missing> 2 weeks ago /bin/sh -c { echo mysql-community-server m...
<missing> 2 weeks ago /bin/sh -c echo 'deb http://repo.mysql.com/a...
<missing> 2 weeks ago /bin/sh -c #(nop) ENV MYSQL_VERSION=8.0.25-...
<missing> 2 weeks ago /bin/sh -c #(nop) ENV MYSQL_MAJOR=8.0
<missing> 2 weeks ago /bin/sh -c set -ex; key='A4A9406876FCBD3C45...
<missing> 2 weeks ago /bin/sh -c apt-get update && apt-get install...
<missing> 2 weeks ago /bin/sh -c mkdir /docker-entrypoint-initdb.d
<missing> 2 weeks ago /bin/sh -c set -eux; savedAptMark="$(apt-ma...
<missing> 2 weeks ago /bin/sh -c #(nop) ENV GOSU_VERSION=1.12
<missing> 2 weeks ago /bin/sh -c apt-get update && apt-get install...
<missing> 2 weeks ago /bin/sh -c groupadd -r mysql && useradd -r ...
<missing> 2 weeks ago /bin/sh -c #(nop) CMD ["bash"]
<missing> 2 weeks ago /bin/sh -c #(nop) ADD file:7362e0e50f30ff454...
root@ubuntu:/home/gfjj#

```

Ahora vamos a cargarnos el contenedor de hello-world, para luego borrar la imagen que antes no nos dejaba borrar. Primero hacemos un ps -a para ver la ID que se le asigna al contenedor, vamos a borrarlo utilizando su nombre. Ponemos docker rm ID y podemos comprobar que lo ha borrado correctamente.

```

root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
da1c4219fd21 mysql "docker-entrypoint.s..." 23 minutes ago Up 10 minutes 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp,
33060/tcp gjff_mysq
4a8e76ae5895 ubuntu "/bin/bash" 34 minutes ago Exited (0) 34 minutes ago
ecstatic_bell
9a3226c57655 hello-world "/hello" 38 minutes ago Exited (0) 37 minutes ago
competent_yallow
root@ubuntu:/home/gfjj# docker rm 9a3226c57655
9a3226c57655
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
da1c4219fd21 mysql "docker-entrypoint.s..." 23 minutes ago Up 11 minutes 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 3306
0/tcp gjff_mysq
4a8e76ae5895 ubuntu "/bin/bash" 35 minutes ago Exited (0) 35 minutes ago
ecstatic_bell
root@ubuntu:/home/gfjj#

```

Ahora hacemos un docker rmi hello-world y vemos que al no tener el contenedor abierto, ya nos permite borrar la imagen.

```

root@ubuntu:/home/gfjj# docker rmi hello-world
Untagged: hello-world:latest
Untagged: hello-world@sha256:5122f6204b6a3596e048758cabba3c46b1c937a46b5be6225b835d091b90e46c
Deleted: sha256:d1165f2212346b2bab48cb01c1e39eee8ad1be46b87873d9ca7a4e434980a7726
Deleted: sha256:f22b99068db93900abe17f7f5e09ec775c2826ecfe9db961fea68293744144bd
root@ubuntu:/home/gfjj# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
mysql latest c0cdc95609f1 2 weeks ago 556MB
ubuntu latest 7e0aa2d69a15 5 weeks ago 72.7MB
root@ubuntu:/home/gfjj#

```

Por último, vamos a parar el contenedor que teníamos activo.

```

root@ubuntu:/home/gfjj# docker stop da1c4219fd21
da1c4219fd21
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
da1c4219fd21 mysql "docker-entrypoint.s..." 27 minutes ago Exited (0) 2 seconds ago
4a8e76ae5895 ubuntu "/bin/bash" 39 minutes ago Exited (0) 39 minutes ago
root@ubuntu:/home/gfjj#

```

CONTENEDOR DE MYSQL EN DOCKER:

Para poder trabajar con contenedores de mysql, primero creamos el volumen de mysql.
docker volume create mysql_data

```
root@ubuntu:/home/gfjj# docker volume create mysql_data  
mysql_data
```

Ahora creamos un contenedor utilizando la imagen de mysql.

docker run -d --name gfjj -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 -v mysql_data:/var/lib/mysql mysql

se crea con el docker run, y tenemos que tener en cuenta que:

- En --name vamos a asignarle el nombre al contenedor. Es interesante anotar cual ponemos, si bien podemos dirigirnos a él mediante la ID aleatoria que se asigna.
- En MYSQL_ROOT_PASSWORD estamos introduciendo la contraseña. Atención porque no va encomillada ni nada similar como en otras plataformas.
- Con -p establecemos el puerto. Por defecto MySQL utiliza el 3306, pero si queremos establecer otro tenemos que especificarlo aquí. Mucho cuidado con cambiarlo y no darnos cuenta.
- Por defecto, los contenedores que creamos son volátiles. Eso implica que al reiniciar o cerrar sesión perdemos todo el contenido. Si tenemos que cargar un contenedor por ejemplo para utilizarlo puntualmente y no guardar cambios (por ejemplo, para abrirlo, navegar de forma segura y cerrarlo) está genial así. Pero si vamos a trabajar en algo estable, podemos quitarle la aleatoriedad con el -y volumen:/ruta

```
root@ubuntu:/home/gfjj# docker run -d --name gfjj -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 -v mysql_data:/var/lib/mysql mysql  
54198d312c70d6180540d1de9e765b93495ec7160f5d5ee66f2020c47e970c6e  
root@ubuntu:/home/gfjj# docker ps -a  
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS  
NAMES  
54198d312c70 mysql "docker-entrypoint.s..." 3 seconds ago Up 1 second 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp  
, 33060/tcp gfjj  
4a8e76ae5895 ubuntu "/bin/bash" About an hour ago Exited (0) About an hour ago  
ecstatic_bell  
root@ubuntu:/home/gfjj#
```

Una vez creado el contenedor, vamos a acceder a él con docker exec
docker exec -it gfjj /bin/bash

```
root@ubuntu:/home/gfjj# docker exec -it gfjj /bin/bash
```

Hacemos un ls para ver el contenido de la carpeta:

```
root@54198d312c70:/# ls  
bin dev entrypoint.sh home lib64 mnt proc run srv tmp var  
boot docker-entrypoint-initdb.d etc lib media opt root sbin sys usr  
root@54198d312c70:/#
```

Para salir del contenedor ponemos exit.

```
root@54198d312c70:/# exit  
exit
```

Entramos de nuevo, y accedemos a MySQL.

```
root@ubuntu:/home/gfjj# docker exec -it gfjj /bin/bash  
root@54198d312c70:/# 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>
```

Introducimos el siguiente código para probar:

```
DROP DATABASE IF EXISTS probando;
```

```
CREATE DATABASE probando;
```

```
USE probando;
```

```
CREATE TABLE GFJJ_prueba (
    ID INT AUTO_INCREMENT PRIMARY KEY,
    Nombre VARCHAR(50) NOT NULL,
    Apellidos VARCHAR(50) NOT NULL,
    F_Nac DATE NOT NULL
);
```

```
INSERT INTO GFJJ_prueba (Nombre, Apellidos, F_Nac) VALUES('Juan Jose','Gomez Fernandez','1984-08-07');
```

```
INSERT INTO GFJJ_prueba (Nombre, Apellidos, F_Nac) VALUES('Rosa Maria','Alonso Vega','1997-05-02');
```

Comprobamos que está correcto:

```
mysql> CREATE DATABASE probando;
Query OK, 1 row affected (0.00 sec)

mysql> USE probando;
Database changed
mysql>
mysql> CREATE TABLE GFJJ_prueba (
    ->     ID INT AUTO_INCREMENT PRIMARY KEY,
    ->     Nombre VARCHAR(50) NOT NULL,
    ->     Apellidos VARCHAR(50) NOT NULL,
    ->     F_Nac DATE NOT NULL
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> INSERT INTO GFJJ_prueba (Nombre, Apellidos, F_Nac) VALUES('Juan Jose','Gomez Fernandez','1984-08-07');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO GFJJ_prueba (Nombre, Apellidos, F_Nac) VALUES('Rosa Maria','Alonso Vega','1997-05-02');
Query OK, 1 row affected (0.00 sec)

mysql> SHOW TABLES;
+-----+
| Tables_in_probando |
+-----+
| GFJJ_prueba         |
+-----+
1 row in set (0.00 sec)

mysql> ■
```

Ahora vamos a salir de MySQL y parar el contenedor. Podemos comprobar que al ser estático y no volátil, el contenedor sigue estando ahí aunque lo cerremos.

```
root@54198d312c70:/# exit
exit
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID        IMAGE       COMMAND                  CREATED             STATUS              PORTS
 NAMES
54198d312c70      mysql      "docker-entrypoint.s..."   42 minutes ago   Up 42 minutes          0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp
cp_gfjj
4a8e76ae5895      ubuntu     "/bin/bash"              2 hours ago      Exited (0) 2 hours ago
ecstatic_bell
root@ubuntu:/home/gfjj# docker stop gfjj
gfjj
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID        IMAGE       COMMAND                  CREATED             STATUS              PORTS
 NAMES
54198d312c70      mysql      "docker-entrypoint.s..."   42 minutes ago   Exited (0) 2 seconds ago
4a8e76ae5895      ubuntu     "/bin/bash"              2 hours ago      Exited (0) 2 hours ago
ecstatic_bell
root@ubuntu:/home/gfjj# ■
```

Hay que tener en cuenta que podemos crear un contenedor sin persistencia de datos, introduciendo -rm en el docker run. Esto lo que hace es que cuando se cierra la conexión, no guarde cambios y elimine el contenedor del disco. Es algo MUY a tener en cuenta en ciertos escenarios, véase por ejem-

plo en un cíber (que cuando acabe el tiempo del usuario cierre el contenedor y borre todo el contenido, preservando la seguridad de los datos del usuario y garantizando la estabilidad del sistema al no almacenar posibles ficheros dañinos) o si lo que queremos es un contenedor para consultar datos offline y no insertar datos.

Vamos a probar que funciona correctamente, creamos el contenedor:

```
root@ubuntu:/home/gfjj# docker run -d --rm --name gfjj_volatile -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 mysql
e3183cd43c03f268cab2e24a41aa36cd798580daeb4cd09c20a56651057b780f
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
e3183cd43c03 mysql "docker-entrypoint.s..." 2 seconds ago Up 2 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 3306
0/tcp gfjj_volatile
54198d312c70 mysql "docker-entrypoint.s..." 52 minutes ago Exited (0) 10 minutes ago
gfjj
4a8e76ae5895 ubuntu "/bin/bash" 2 hours ago Exited (0) 2 hours ago
ecstatic_bell
root@ubuntu:/home/gfjj# docker stop gfjj_volatile
gfjj_volatile
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
54198d312c70 mysql "docker-entrypoint.s..." 52 minutes ago Exited (0) 10 minutes ago
4a8e76ae5895 ubuntu "/bin/bash" 2 hours ago Exited (0) 2 hours ago
root@ubuntu:/home/gfjj#
```

Vemos que funciona correctamente. Nos crea el contenedor, podríamos trabajar con él sin problema ninguno... Lo paramos y al pararlo elimina el contenedor automáticamente. Si trabajamos en una empresa con contenedores volátiles sin y con persistencia de datos simultáneamente, es fundamental ponerles etiquetas MUY VISUALES a los nombres de los contenedores, puesto que si tenemos nombres similares podemos estar introduciendo datos en donde no se debe, o bien no introduciéndolos en donde deberíamos.

Docker Flag-Link

Docker tiene un sistema de vinculación que le permite vincular varios contenedores y enviar información de conexión de uno a otro. Cuando los contenedores están vinculados, la información sobre un contenedor de origen se puede enviar a un contenedor de destinatarios. Esto permite al destinatario ver los datos seleccionados que describen aspectos del contenedor de origen.

Vamos a probarlo. Creamos el contenedor con la instrucción anterior y luego creamos el link como mostramos en la imagen.

```
root@ubuntu:/home/gfjj# docker run -d --name gfjj -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 -v mysql_data:/var/lib/mysql mysql
3fe348ef1357d0f75ee541d18c8e9cec8abf79f47853adeae5fef454a6a2d32e
root@ubuntu:/home/gfjj# docker run -d --rm --link gfjj -p 8080:8080 mysql
bfb6beadcbebdb11ab99f39f33204c9140fc0cd2ef88fb19866307ef8cd9c0395
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
3fe348ef1357 mysql "docker-entrypoint.s..." 10 seconds ago Up 9 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp
cp gfjj
4a8e76ae5895 ubuntu "/bin/bash" 2 hours ago Exited (0) 2 hours ago
ecstatic_bell
root@ubuntu:/home/gfjj#
```

PHPMyADMIN EN DOCKER:

Vamos a conectar contenedores conectados a la red utilizando un flag link. Para ello, primero nos cargamos los contenedores abiertos y creamos tanto el anterior, como el nuevo con PHPMyADMIN.

NOTA: *Importante la parte de eliminar conexiones activas. Hay que recordar que solamente podemos abrir un contenedor en un puerto, así que si no nos damos cuenta y queremos crear un segundo contenedor en puerto 3306 nos va a dar un error de conexión.*

```
docker run -d \
--name gfjj \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=Abcd1234. \
-v mysql_data:/var/lib/ \
mysql
```

```
root@ubuntu:/home/gfjj# docker run -d --name gfjj -e MYSQL_ROOT_PASSWORD=Abcd1234. -p 3306:3306 -v mysql_data:/var/lib/mysql mysql
3fe348ef1357df75ee541d18c8e9cec8abf79f47853adeae5fef454a6a2d32e
root@ubuntu:/home/gfjj# docker run -d --rm --link gfjj -p 8080:8080 mysql
bf6beadcbecbdb11ab99f39f33204c9140f6cd2ef88fb19866307ef8bcd9c0395
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
 NAMES
3fe348ef1357 mysql "docker-entrypoint.s..." 10 seconds ago Up 9 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp
cp_gfjj 4a8e76ae5895 ubuntu "/bin/bash" 2 hours ago Exited (0) 2 hours ago
ecstatic_bell
root@ubuntu:/home/gfjj#
```

```
docker run -d \
--rm \
--link gfjj \
-e PMA_ARBITRARY=1 \
-p 8080:80 \
phpmyadmin/phpmyadmin
```

```
root@ubuntu:/home/gfjj# docker run -d --rm --link gfjjlink -e PMA_ARBITRARY=1 -p 8080:80 phpmyadmin/phpmyadmin
16bc4c0dc4d6a0155220972b1b4e59225edce6d0222fe1fb66d459125a0175ad
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS
 NAMES
16bc4c0dc4d6 phpmyadmin/phpmyadmin "/docker-entrypoint...." 10 seconds ago Up 9 seconds 0.0.0.0:8080->80/tcp, :::8080->80/tcp
p_brave_chebyshev b3c72249b7eb mysql "docker-entrypoint.s..." 11 minutes ago Up 11 minutes 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp
6/tcp, 3306/tcp gfjjlink
4a8e76ae5895 ubuntu "/bin/bash" 3 hours ago Exited (0) 3 hours ago
ecstatic_bell
root@ubuntu:/home/gfjj#
```

Conectando dos contenedores utilizando una user-defined bridge network:

Antes de nada, nos cargamos los contenedores con docker rm -f \$(docker ps -qa)

```
root@ubuntu:/home/gfjj# docker rm -f $(docker ps -qa)
16bc4c0dc4d6
b3c72249b7eb
4a8e76ae5895
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
root@ubuntu:/home/gfjj#
```

Creamos la red con el comando docker network create *NOMBRE*

```
root@ubuntu:/home/gfjj# docker network create netgfjj
506348a87556e507414a7fa1e340abc627bcad8d1a626e8c7ef8fbc78e730f02
root@ubuntu:/home/gfjj#
```

```
docker run -d \
--name gfjj \
--network netgfjj \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=Abcd1234. \
-v mysql_data:/var/lib/ \
mysql
```

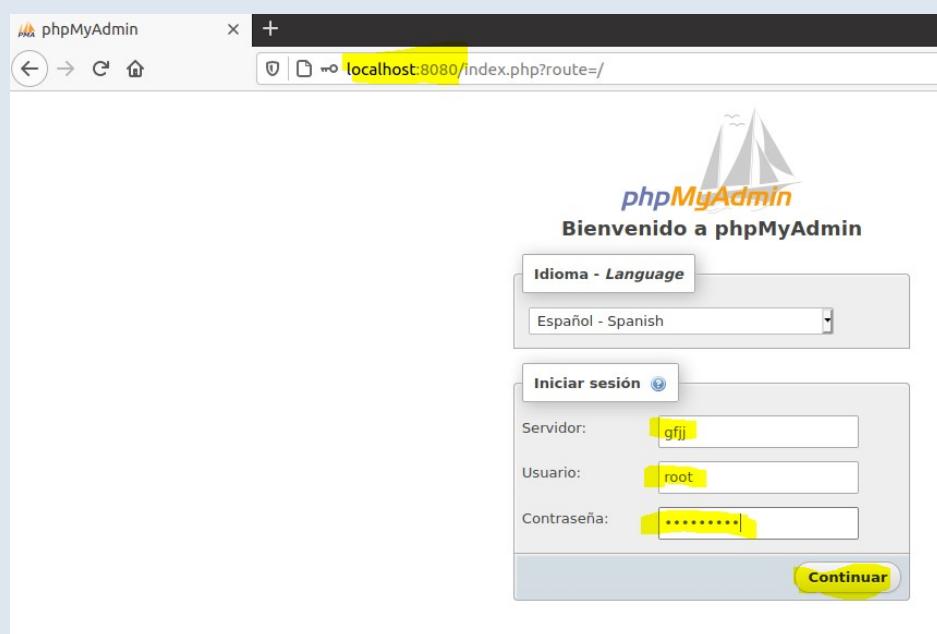
Al añadirle el --network estamos especificando la red.

```
root@ubuntu:/home/gfjj# docker run -d \
> --name gfjj \
> --network netgfjj \
> -p 3306:3306 \
> -e MYSQL_ROOT_PASSWORD=Abcd1234. \
> -v mysql_data:/var/lib/ \
> mysql
4fbdb3bea6ed483a86de84a3be78aba332a0a6ce29b06db8aadf86b9638d95103
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAME
4fbdb3bea6ed4 mysql "docker-entrypoint.s..." 28 seconds ago Up 27 seconds 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060/tcp gfjj
root@ubuntu:/home/gfjj#
```

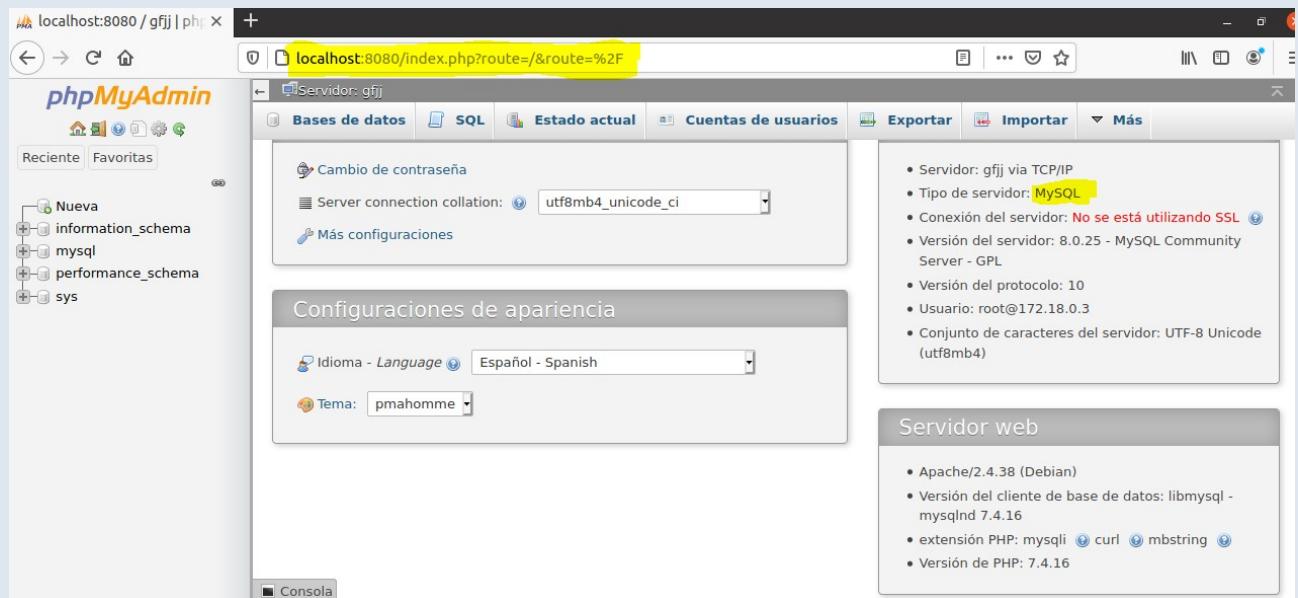
```
docker run -d \
--rm \
--network netgfjj \
-e PMA_ARBITRARY=1 \
-p 8080:80 \
phpmyadmin/phpmyadmin
```

```
root@ubuntu:/home/gfjj# docker run -d \
> --rm \
> --network netgfjj \
> -e PMA_ARBITRARY=1 \
> -p 8080:80 \
> phpmyadmin/phpmyadmin
c70b2d38809ce911d044992afca6f00a5d9257135afc30e1e7c1ae6f0581d487
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
c70b2d38809c phpmyadmin/phpmyadmin "/docker-entrypoint...." 23 seconds ago Up 22 seconds 0.0.0.0:8080->80/tcp, :::8080->80/tcp objective_leavitt
4fbdb3bea6ed4 mysql "docker-entrypoint.s..." 3 minutes ago Up 3 minutes 0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 3306/tcp gfjj
root@ubuntu:/home/gfjj#
```

Comprobamos que funciona correctamente. Abrimos Firefox y cargamos <http://localhost:8080> y nos abre el formulario de phpmyadmin. Introducimos los datos correctamente y estamos dentro:



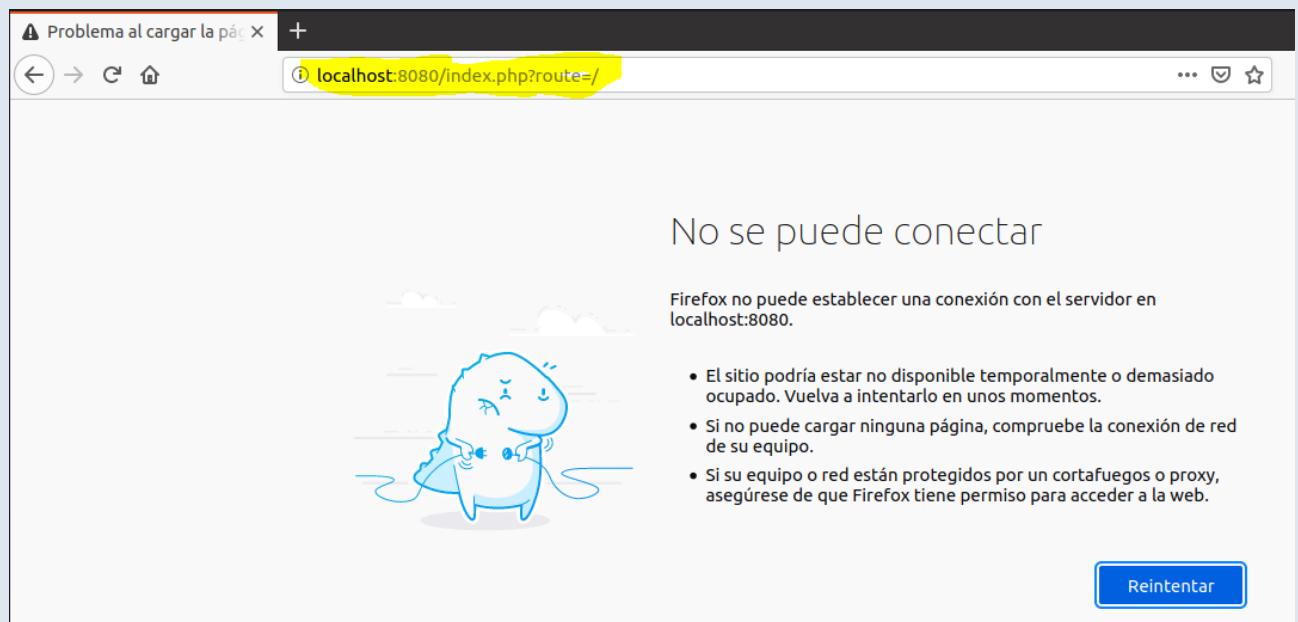
Vemos que nos ejecuta PHPMyADMIN correctamente a través del enlace que hemos fijado, por el puerto 8080.



Ahora me cargo todas las conexiones activas y la red (docker network rm NOMBRE)

```
root@ubuntu:/home/gfjj# docker rm -f $(docker ps -qa)
c70b2d38809c
4fb3bea6ed4
root@ubuntu:/home/gfjj# docker network rm netgfjj
netgfjj
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS          NAMES
root@ubuntu:/home/gfjj#
```

Acto seguido pruebo otra vez a conectarme a PHPMyADMIN y comprobamos que no podemos acceder.



LANZAR MÚLTIPLES CONTENEDORES EN DOCKER:

Cuando explicamos las características de Docker vimos que podíamos ejecutar múltiples contenedores simultáneamente. En este caso vamos a crear varios contenedores que alojen diferentes servicios en una misma red. Lo primero nos conectamos como usuarios administradores y creamos la red.

docker network create gfjjmulti

```
root@ubuntu:/home/gfjj# docker network create gfjjmulti
9ec41396b75d3a8b9b38d3725749e5dff90e90e1a0765e9e5069e280c421b251
root@ubuntu:/home/gfjj# docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
aa19cfcc4c323  bridge    bridge      local
9ec41396b75d   gfjjmulti  bridge      local
2528ed1de36a   host      host      local
b305cc45a002   none      null      local
root@ubuntu:/home/gfjj#
```

Ahora creamos un volumen específico para estos 3 contenedores.

docker volume create gfjjvol

```
root@ubuntu:/home/gfjj# docker volume create gfjjvol
gfjjvol
root@ubuntu:/home/gfjj# docker volume ls
DRIVER      VOLUME NAME
local       25bb957b96015a70c52292966238a8e7ffa2e75e5f231c38ba035c00d3c43dd7
local       211d68b05f486693d60cbf033dac547438325f7e6f95cb0613c07e19f5f22816
local       gfjjmysql_data
local       gfjjvol
local       mysql_data
root@ubuntu:/home/gfjj#
```

Ahora creamos una instancia del contenedor de MySQL en la red y volumen que acabamos de crear.

*docker run -d \
--rm \
--name gfjjmysqlbd \
--network gfjjmulti \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=root \
-e MYSQL_DATABASE=GFJJ_BD \
-e MYSQL_USER=GFJJ \
-e MYSQL_PASSWORD=Abcd1234. \
-v gfjjvol:/var/lib/mysql \
mysql:5.7.28*

```
root@ubuntu:/home/gfjj# docker run -d \
> --rm \
> --name gfjjmysql \
> --network gfjjmulti \
> -p 3306:3306 \
> -e MYSQL_ROOT_PASSWORD=root \
> -e MYSQL_DATABASE=GFJJ_BD \
> -e MYSQL_USER=GFJJ \
> -e MYSQL_PASSWORD=Abcd1234. \
> -v gfjjvol:/var/lib/mysql \
> mysql:5.7.28
29e186d8a759dfa422542e35596efe49af175c732ddebb12d7be36ab6aa7a066
root@ubuntu:/home/gfjj#
```

Ahora creamos la instancia de PHPMyADMIN, tambien utilizando la red creada para ello.

```
docker run -d \
--rm \
--network gfjjmulti \
-e PMA_ARBITRARY=1 \
-p 8080:80 \
phpmyadmin/phpmyadmin
```

```
root@ubuntu:/home/gfjj# docker run -d \
> --rm \
> --network gfjjmulti \
> -p 8080:80 \
> phpmyadmin/phpmyadmin
c28cbd5d4b7b4f3aeac1525ede512ccb708805a4c3360a8f00db3e07794f94
root@ubuntu:/home/gfjj#
```

Podemos observar que tenemos funcionando ambos contenedores.

```
root@ubuntu:/home/gfjj# docker ps
CONTAINER ID   IMAGE          COMMAND       CREATED      STATUS        PORTS
 NAMES
23fe00c52e77   phpmyadmin/phpmyadmin   "/docker-entrypoint..."   7 minutes ago   Up 7 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp
charming_mculty
4eb66b32067e   mysql:5.7.28       "docker-entrypoint.s..."   8 minutes ago   Up 8 minutes   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 3306
p_gfjjmysqlbd
root@ubuntu:/home/gfjj#
```

La gracia de esto es crear en una misma red todos los contenedores de servicios que estemos utilizando. Una gran ventaja de Docker con respecto a la virtualización habitual, es que nos permite ejecutar los contenedores por separado, de modo que estamos otorgando mayor seguridad a los servicios (solamente arrancamos los que tengamos que utilizar en cada momento), así como optimización de recursos.

Las principales características de un volumen de datos son:

- ✓ Se utiliza para guardar e intercambiar información de forma independientemente a la vida de un contenedor.
- ✓ Se utiliza para guardar e intercambiar información entre contenedores. Podemos dejar información en un volumen y acceder a ella desde distintos contenedores. Sería como una carpeta compartida.
- ✓ No se borra aunque borremos los contenedores que lo utilizan.
- ✗ Son directorios del host montados en un directorio del contenedor, aunque también se pueden montar solo ficheros. Aquí lo peligroso es mantener la cadena de privilegios para evitar que se efectúe una escalada y se alcance (mediante un ataque) al host.
- ✓ Si montamos en un directorio ya existente de un contenedor un volumen de datos , su contenido no será eliminado.
- ✗ No se pueden montar cuando creamos una imagen utilizando el fichero dockerfile

MARIADB EN DOCKER:

Vamos a crear un contenedor de docker para María DB.

```
docker run -d \
--rm \
-p 127.0.0.1:3306:3306 \
--name gfjjmari \
-e MARIADB_ROOT_PASSWORD=Abcd1234. \
-d mariadb
```

```
root@ubuntu:/home/gfjj# docker run -d \
> --rm \
> -p 127.0.0.1:3306:3306 \
> --name gfjjmari \
> -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
b846f4f4774a: Pull complete
66409f940bd2: Pull complete
82d8723e99d8: Pull complete
55edbf0f673e: Pull complete
c34793730ad6: Pull complete
8f1925a0d734: Pull complete
72904fb5fd0b: Pull complete
Digest: sha256:0c3c560359a0da112134a52122aa9b78fec5f9dd292a01ee7954de450f25f0c1
Status: Downloaded newer image for mariadb:latest
9914530c25d3c5362dda48d3ad4ed1212c815cf6d69f7128f6426a7f6f78facd
root@ubuntu:/home/gfjj#
```

Ahora probamos a arrancarlo.. si bien se me hace raro que me carga como si fuera MySQL, aunque especifica que es el motor de MariaDB.

```
root@ubuntu:/home/gfjj# docker start gfjjmari
gfjjmari
root@ubuntu:/home/gfjj# mysql -h 127.0.0.1 -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.5.5-10.5.10-MariaDB-1:10.5.10+maria-focal mariadb.org binary distribution

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> show databases;
+-----+
| Database      |
+-----+
| information_schema |
| mysql          |
| performance_schema |
+-----+
3 rows in set (0,00 sec)

mysql> exit
Bye
```

MONGODB EN DOCKER:

Para crear el contenedor de MongoDB, vamos a instalar DOCKER-COMPOSE. Primero ejecutamos la instrucción:

```
sudo curl -L "https://github.com/docker/compose/releases/download/1.26.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Le damos permisos de ejecución y verificamos que funciona:

```
sudo chmod +x /usr/local/bin/docker-compose
```

```
root@ubuntu:/home/gfjj# sudo chmod +x /usr/local/bin/docker-compose
root@ubuntu:/home/gfjj# docker-compose -v
docker-compose version 1.26.0, build d4451659
root@ubuntu:/home/gfjj# █
```

Ahora creamos la carpeta wordpress, con mkdir wordpress, entramos a ella y ejecutamos un *nano docker-compose.yml*

version: '3.1'

services:

```
wordpress:
  image: wordpress
  restart: always
  ports:
    - 80:80
  environment:
    WORDPRESS_DB_HOST: gfjjwp
    WORDPRESS_DB_USER: GFJJ
    WORDPRESS_DB_PASSWORD: Abcd1234.
    WORDPRESS_DB_NAME: gfjjdb
  volumes:
    - wordpress_db:/var/www/html
```

db:

```
image: mysql:5.7.28
restart: always
environment:
  MYSQL_DATABASE: GFJJ_DB
  MYSQL_USER: GFJJ
  MYSQL_PASSWORD: Abcd1234.
  MYSQL_RANDOM_ROOT_PASSWORD: '1'
volumes:
  - gfjjvol:/var/lib/mysql
```

volumes:

```
wordpress_db:
gfjjvol:
```

```
root@ubuntu: /home/gfjj/wordpress
GNU nano 4.8                                            docker-compose.yml

services:

  wordpress:
    image: wordpress
    restart: always
    ports:
      - 8080:80
    environment:
      WORDPRESS_DB_HOST: gfjj
      WORDPRESS_DB_USER: root
      WORDPRESS_DB_PASSWORD: Abcd1234.
      WORDPRESS_DB_NAME: gfjjdb
    volumes:
      - wordpress:/var/www/html

  db:
    image: mysql:5.7
    restart: always
    environment:
      MYSQL_DATABASE: GFJJ_BD
      MYSQL_USER: root
      MYSQL_PASSWORD: Abcd1234.
      MYSQL_RANDOM_ROOT_PASSWORD: '1'
    volumes:
      - db:/var/lib/mysql
```

Ahora montamos el contenedor de MySQL:

```
docker run -d \
--rm \
--name gfjmysqlbd \
--network gfjmulti \
-p 3306:3306 \
-e MYSQL_ROOT_PASSWORD=root \
-e MYSQL_DATABASE=GFJJ_BD \
-e MYSQL_USER=GFJJ \
-e MYSQL_PASSWORD=Abcd1234. \
-v gfjvol:/var/lib/mysql \
mysql:5.7.28
```

NOTA: No tenía ninguna imagen cargada pero aún así me daba un error porque el puerto 3306 estaba en uso. Si eso pasa tenemos que ejecutar:

```
sudo kill `sudo lsof -t -i:3306`
```

DOCKER HUB:

Docker HUB es un servicio de registro de repositorios. Nos permite extraer y enviar imágenes de la ventana acoplable hacia y desde Docker Hub. Podemos tratar esto como un GitHub, donde obtenemos y enviamos nuestro código fuente, pero en el caso de Docker Hub, descargamos o publicamos nuestras imágenes de contenedor. Es un repositorio en línea basado en la nube que almacena ambos tipos de repositorios, es decir, el repositorio público y el privado.

Para poder acceder a DockerHUB vamos a [su página web](#). Desde ahí, al no tener sesión nos aparece inmediatamente un formulario de registro, en donde incluir ID a crear, dirección de correo electrónico así como la contraseña (y cubrir el CAPTCHA).

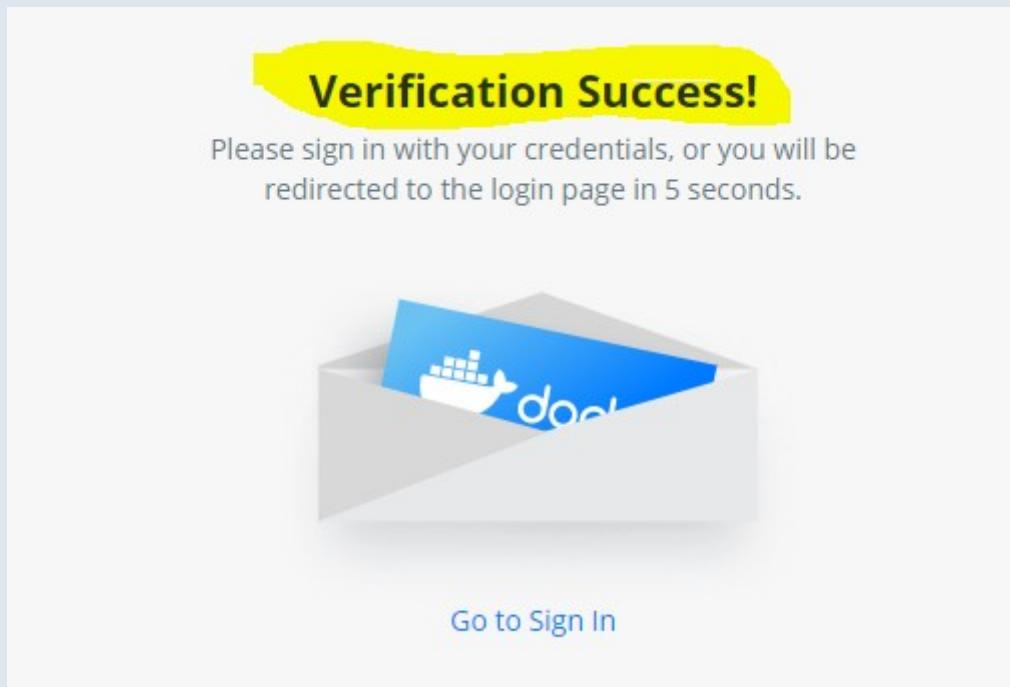
The screenshot shows the Docker Hub sign-up page. It has fields for a username ('gtjasir'), email ('awjuanjose@gmail.com'), and password ('*****'). There's a checkbox for receiving updates and a reCAPTCHA verification. A large green 'Sign Up' button is at the bottom.

Ahora nos deja elegir entre 3 planes:

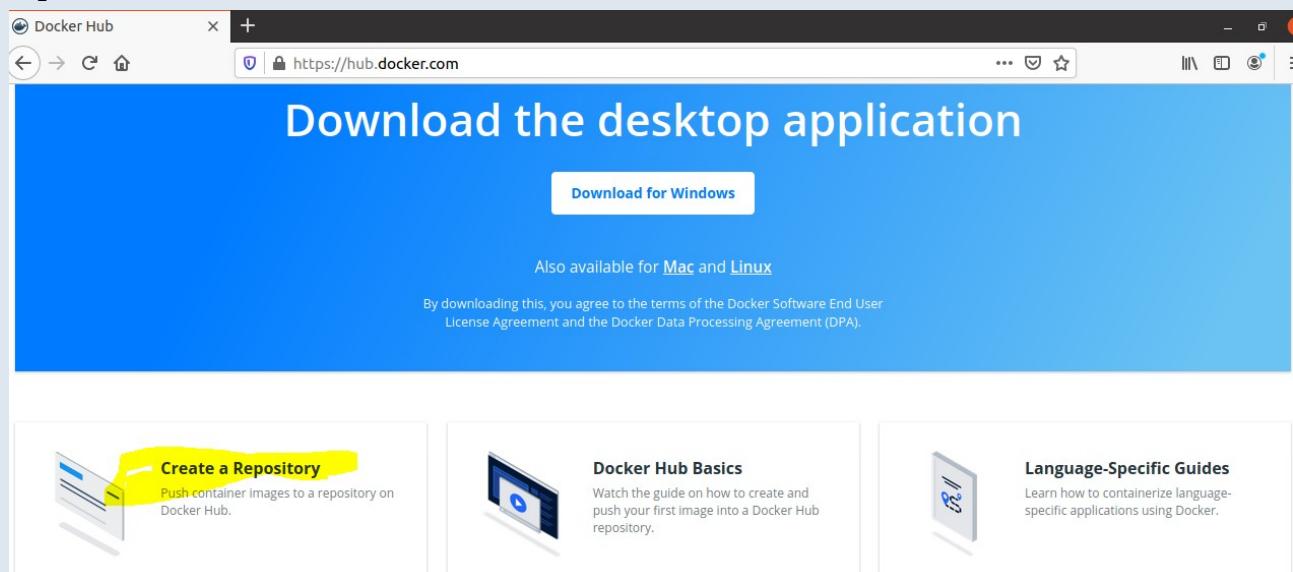
- 1) Gratuito (elegimos ese)
- 2) Pro (incluye mayor capacidad y soporte)
- 3) Equipo (para mínimo 5 usuarios)

CY DEVELOPER FAVORITE		
Free 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	Pro 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	Team 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
\$0 /month Continue with Free	\$5 /month With annual plan Buy Now	\$7 user/month Starts at \$25 for 5 users with annual plan Buy Now

Nos envían un correo de verificación. Hacemos click en el enlace y nos reenvía automáticamente a la página de inicio de DOCKER HUB.



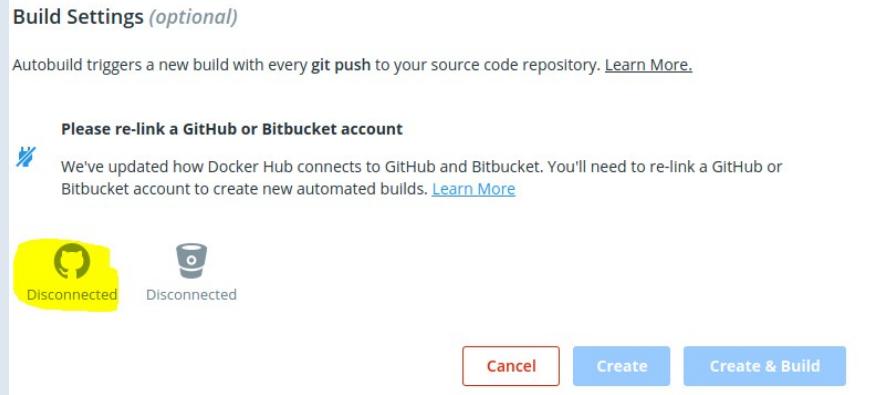
Una vez que nos logeamos, nos figura tal como vemos en la imagen. Podemos darle a “**Crear un repositorio**”



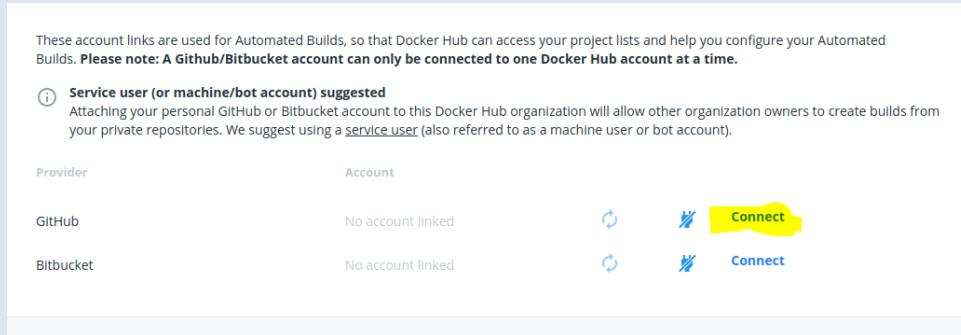
Escogemos el nombre y descripción del repositorio. Es interesante que ya nos vienen los comandos a la derecha.

A screenshot of the "Create Repository" form on Docker Hub. The repository name is "gfjasir" and the description is "Repositorio para ASXBD". On the right, there is a "Pro tip" section with the command: "docker tag local-image:tagname new-repo:tagname" and "docker push new-repo:tagname". Below this, a note says: "Make sure to change tagname with your desired image repository tag." At the bottom, there are sections for "Visibility" (Public is selected, appearing in search results) and "Build Settings (optional)".

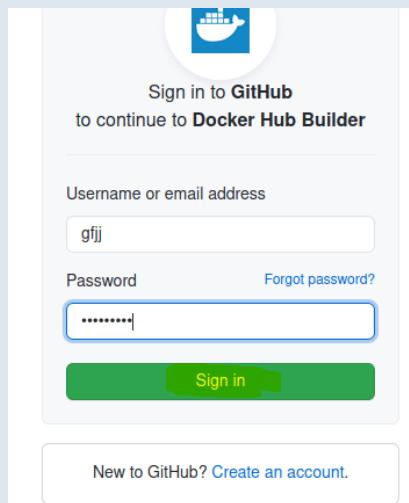
Tambien tenemos directamente la opción de vincular nuestra cuenta de GITHUB al DOCKER HUB. Es interesante. Para ello, hacemos click en el icono del gatete.



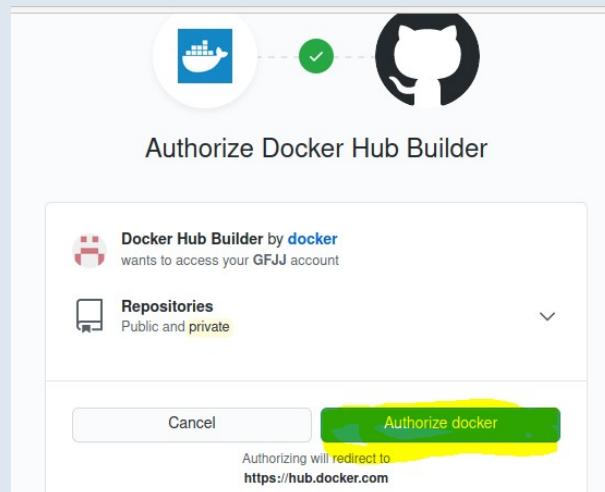
Ahora le damos a conectar y nos envía a la página de inicio de GITHUB.



Introducimos usuario y contraseña, como viene en la imagen.



Ahora le damos al botón de autorización a Docker para vincularlo con GITHUB.



Automáticamente nos manda a la ventana anterior. Pero esta vez indica que lo tenemos asociado.

The screenshot shows the Docker Hub settings page for a user account. On the left, there's a sidebar with options like General, Linked Accounts, Security, Default Privacy, Notifications, Convert Account, and Deactivate Account. The main content area has a note about the settings page for the user account. Below it, under 'Linked Accounts', there's a section titled 'Linked Accounts' with a note that these links are used for automated builds. It says 'Please note: A Github/Bitbucket account can only be connected to one Docker Hub account at a time.' There's a section for GitHub where it says 'Service user (or machine/bot account) suggested' and 'Attaching your personal GitHub or Bitbucket account to this Docker Hub organization will allow other organization owners to create your private repositories. We suggest using a service user (also referred to as a machine user or bot account).'. Below this, there's a table with two rows: GitHub (with account 'GFJJ' highlighted in yellow) and Bitbucket (with 'No account linked'). There are 'Connect' buttons for both.

Vamos a probar que funciona. Nos logueamos con
docker login -u gfjjasir

```
root@ubuntu:/home/gfjj/wordpress# docker login -u gfjjasir
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
```

Una vez hecho esto, vamos a crear una imagen para subirla. Hacemos una imagen de un contenedor que tenemos.

```
root@ubuntu:/home/gfjj# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
 NAMES
3df08eb0df43        mysql:5.7.28      "docker-entrypoint.s..."   3 minutes ago       Up 3 minutes (Paused)   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 3306
0/tcp                gfjjdockhub

root@ubuntu:/home/gfjj# docker commit 3df08eb0df43 gfjj:update
sha256:0e9f3d02943bd048219afbd4753a542e374889a2c5edb1f58bc14259ad50bb3
root@ubuntu:/home/gfjj# docker images
REPOSITORY          TAG           IMAGE ID            CREATED             SIZE
gfjj               update        0e9f3d02943    2 seconds ago     437MB
mariadb             latest        efff629089685   6 days ago        408MB
wordpress           latest        0adda6ed742f    6 days ago        551MB
mysql               5.7          2c9028880e58   2 weeks ago       447MB
mysql               latest        c6cdcc95609f1   2 weeks ago       556MB
mongo               latest        07630e791de3   2 weeks ago       449MB
ubuntu              latest        7e0aa2d69a15   5 weeks ago       72.7MB
mongo-express       latest        51fc3f2af7a1   6 weeks ago       128MB
phpmyadmin/phpmyadmin latest        72000eb04892   2 months ago      477MB
mysql               5.7.28       db39680b63ac   17 months ago     437MB
root@ubuntu:/home/gfjj#
```

Nos logeamos (otra vez, que me había salido) en dockerhub.

```
root@ubuntu:/home/gfjj# docker login -u gfjjasir
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/gfjj#
```

Con el comando docker tag etiquetamos la imagen, es fundamental.

`docker tag idImagen usuarioDockerHub/repositorioDockerHub`

Después la subimos con un `docker push nombre-etiquetadeltag`

```
root@ubuntu:/home/gfjj# docker tag 0e9f33d02943 gfjjasir/gfjj:e1
root@ubuntu:/home/gfjj# docker push gfjjasir/gfjj:e1
The push refers to repository [docker.io/gfjjasir/gfjj]
0cba077cf62e: Pushed
ab91c8a5ef0a: Pushed
ce4569bf481c: Pushed
17d2117d1ff3: Pushed
400dd8938406: Pushed
2a60eb850753: Pushed
cf6a13051478: Pushed
fef9e518b701: Pushed
955b4c88a6e8: Pushed
61cb1c0dec27: Pushed
25575e327c84: Pushed
814c70fd62: Pushed
e1: digest: sha256:899e2bc652446233f27380f3adfe17c6f264e9ed4034c3b6298e7b5d7a257a29 size: 2829
root@ubuntu:/home/gfjj#
```

Ahora comprobamos que está ok:

The screenshot shows the Docker Hub repository page for the user `gfjjasir` and the repository `gfjj`. The URL is `https://hub.docker.com/repository/docker/gfjjasir/gfjj/general`.

Tags and Scans
This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
e1		2 minutes ago	2 minutes ago

[See all](#)

VULNERABILITY SCANNING - DISABLED
[Enable](#)

Docker commands
To push a new tag to this repository,
`docker push gfjjasir/gfjj:tag`

Recent builds
Link a source provider and run a build to see recent builds.