**Introducción**

Microsoft además provee al usuario las herramientas necesarias para administrar la seguridad de las bases de datos dentro de Azure, así como los tipos de replicación y las auditorías para proteger la información frente a cualquier ataque externo que podría dejar consecuencias catastróficas para la base de datos. El capítulo 9 del libro de Pranab Mazumdar detalla dichas características a profundidad, así como la forma de uso respectiva.

**Conclusión**

Para la seguridad de las bases de datos, Azure posee características de seguridad tales como la georeplicación para restaurar la información en caso de fallo en el nodo primario del servidor, y las auditorías con el fin de protegerla frente a amenazas reales y/o percibidas, e incluso la posibilidad de controlar el acceso a una determinada fila de una tabla. También existe la encriptación total o parcial de los datos, sin importar si la base de datos está o no en plena ejecución, siendo la encriptación permanente la adición más reciente a las características que SQL Server ofrece.

**Capítulo 8: Base de Datos Azure SQL**

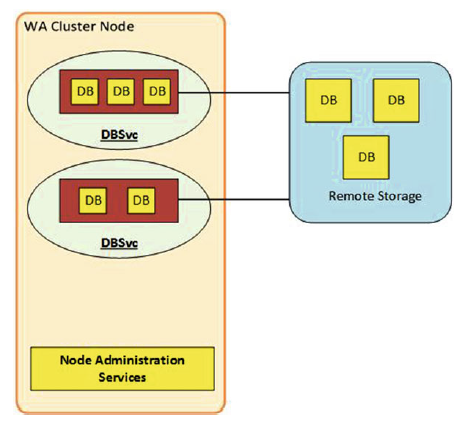
La base de datos Azure SQL es un servicio de bases de datos relacionales en Microsoft Azure, basados en el motor de Microsoft SQL Server, y con casi todas las capacidades de SQL Server para misión crítica. Azure SQL está diseñado para proveer bases de datos altamente disponibles y escalables como un servicio con rendimiento predecible, continuidad de negocios y capacidades de protección de datos.

ARQUITECTURA DE LA BASE DE DATOS AZURE SQL

Azure SQL está construido sobre el framework de Windows Azure, el cual provee administración de máquina y la funcionalidad de aplicación distribuida. Un cluster de SQL Azure está constituido por un anillo de control y varios anillos “inquilinos”, donde cada anillo equivale a un cluster físico de Windows Azure que consiste en una colección de nodos que ejecutan una o más aplicaciones.

ANILLO INQUILINO

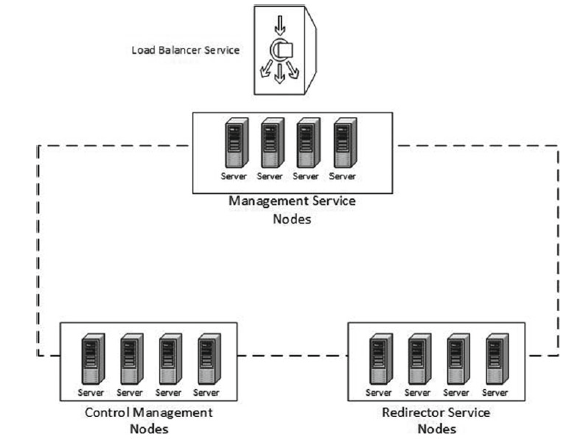
Un anillo inquilino es un cluster físico de Windows Azure, donde cada nodo del cluster está diseñado para ejecutar una aplicación. Dicha aplicación es de tipo DBService, el cual es un servicio del motor de SQL Server o del motor Hekaton (procesamiento de transacciones en línea dentro de la memoria).



La arquitectura del anillo inquilino permite la creación y el funcionamiento de múltiples DBServices, perteneciendo a uno o varios clientes. Ya que cada DBService es independiente de otras aplicaciones en DBService ejecutando dentro del mismo nodo en el anillo inquilino, esta arquitectura permite al mismo nodo alojar varias bases de datos cliente.

ANILLO DE CONTROL

El anillo de control provee los servicios de administración, provisionamiento y redireccionamiento dentro del cluster físico de Azure. Además, determina la ubicación de las bases de datos en los anillos inquilinos y el enrutado hacia las bases de datos inquilinas apropiadas.



NODO DE ADMINISTRACION DEL CONTROL

Provee los servicios de administración interna del cluster, los cuales proveen capacidades como administración de la capacidad, del anillo inquilino, migración, entre otros. Uno de los componentes clave del nodo de administración del control es el almacenamiento de metadatos del cluster, el cual es el punto de todos los metadatos relacionados con los clusters de Azure SQL (estado del cluster, de los recursos en ejecución, e información adicional que asegura el buen funcionamiento del cluster).

NODO DE LOS SERVICIOS DE ADMINISTRACIÓN

Se encarga de alojar los componentes administrativos que proveen APIs REST para que los usuarios finales puedan administrar la base de datos Azure SQL. Dichos servicios se ejecutan como servicios sin estado en múltiples nodos de administración activos. Si un nodo falla, el servicio se reinicia en otro y la operación continúa a partir del punto en donde se dio el fallo.

NODO DEL SERVICIO DE REDIRECCIÓN

Este nodo permite la redirección TDS a la base de datos Azure SQL.

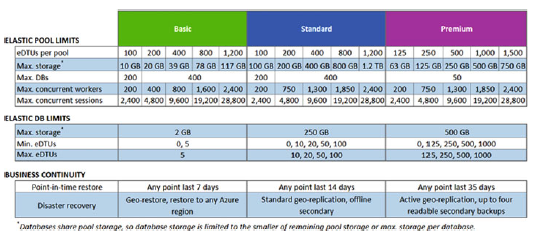
La arquitectura utiliza además otros servicios para lograr una alta disponibilidad, balance de carga y gobernación de los recursos.

ESLABONES DE SERVICIO DE LA BASE DE DATOS AZURE SQL

La base de datos Azure SQL está disponible en tres eslabones de servicio con múltiples niveles de rendimiento, disponibles bajo cada uno de ellos. Cada nivel de rendimiento contiene un conjunto creciente de recursos del sistema para proveer niveles crecientes de tasas de transferencia efectiva. Los recursos disponibles bajo cada nivel de rendimiento se expresan en términos de DTU (unidades de tasas de transferencia efectiva en la base de datos). Los DTU describen la cantidad relativa de potencia, memoria y tasas de E/S efectiva requeridas para completar una transacción de la base de datos.

POOL ELÁSTICO DE LA BASE DE DATOS

La base de datos Azure SQL también permite la creación y la administración de varias bases de datos en un pool elástico de la base de datos. Éste provee un conjunto de DTUs compartidos o eDTUs asociados al pool, los cuales pueden ser utilizados por las bases de datos en el pool.



ESLABONES DE SERVICIO: CAPACIDADES Y LÍMITES

* Tamaño máximo de la base de datos: Especifica el límite máximo para el tamaño de la base de datos.
* Respaldo automático y restauración de punto en el tiempo: Azure SQL provee características para el respaldo automático. Estos respaldos pueden ser restaurados en cualquier punto del tiempo dentro del período de retención especificado por los eslabones de servicio.
* Almacenamiento máximo de OLTP en memoria: Especifica el límite máximo de almacenamiento para el guardado de objetos OLTP en memoria. Su desventaja es que es solamente aplicable a los eslabones de alta categoría.
* Máximo de solicitudes concurrentes: Es el número máximo de solicitudes concurrentes por parte del usuario o la aplicación que se pueden ejecutar sobre la base de datos.
* Máximo de inicios de sesión concurrentes: Representa el límite en el número de usuarios o aplicaciones permitidos para ingresar a la base de datos de forma simultánea. Este límite no es aplicable al pool elástico de la base de datos.

HERRAMIENTAS DE ADMINISTRACIÓN

La administración de la base de datos Azure SQL puede ser hecho a través de las siguientes herramientas:

* Azure Portal
* SQL Server Management Studio
* Herramientas de datos en SQL Server
* Utilidades de línea de comandos y APIs REST

AZURE PORTAL

Azure Portal es una aplicación basada en web que tiene las capacidades de crear, eliminar, restaurar, y administrar la base de datos Azure SQL y su servidor lógico asociado. Además, posee las capacidades de monitorear el rendimiento de la base de datos, configurar la seguridad y alta escalabilidad, y la capacidad de modificar el eslabón de servicio de la base de datos.

SQL SERVER MANAGEMENT STUDIO

Las bases de datos Azure SQL soportan tanto la autenticación de SQL como la autenticación de Windows. Cuando se va a conectar a una base de datos Azure, se debe especificar el nombre del servidor SQL lógico que es obtenido como parte de la creación de la base de datos.

Se puede utilizar SSMS para una gran cantidad de actividades dentro de la base de datos, tales como:

* Monitoreo y administración usando los eventos extendidos
* Creación, diseño y desarrollo de la base de datos
* Administrar la seguridad del servidor SQL.

SQL SERVER DATA TOOLS

SSDT es una herramienta gratuita que es usada para construir bases de datos relacionales en SQL Server y Azure SQL, paquetes SSIS, modelos de datos SSAS y reportes SSRS. SSDT es un ambiente de desarrollo que se puede utilizar para diseñar la base de datos Azure SQL.

UTILIDADES DE LINEA DE COMANDOS Y REST API

Las utilidades de línea de comandos como PowerShell pueden ser empleados para crear y administrar bases de datos Azure SQL. Además, Azure contiene un conjunto de REST APIs que también son utilizados.

AZURE SQL VS SQL SERVER EN LA MÁQUINA VIRTUAL DE AZURE

Azure SQL funciona muy bien en escenarios donde es necesario la provisión y la administración de varias bases de datos, en donde toda la administración y los gastos de parcheo están a cargo del vendedor, el cual ayuda a las organizaciones y los usuarios concentrarse solamente en el diseño y deploy de la base de datos. Sin embargo, no provee características periféricas como la replicación, agente de SQL Server, entre otros.

El uso de SQL Server en una máquina virtual de Azure es la opción apropiada para casos donde una organización busca extender sus despliegues bajo premisa en la nube. Con esta ventaja, el equipo de tecnología de información de la organización tiene el control administrativo absoluto sobre las máquinas virtuales.

REPLICACIÓN DE TRANSACCIONES

La replicación de transacciones es una característica de replicación de datos o de esquemas de SQL Server. Ésta permite a los usuarios configurar las bases de datos de Azure SQL como suscriptores. Durante la etapa inicial, la base de datos suscriptora se sincroniza haciendo uso de una copia de la base de datos del publicador. Después de este proceso, todos los cambios que se realicen en la base de datos primaria serán capturados por logRead.exe y almacenados en la base de datos de distribución.

La base de datos Azure SQL no podrá ser configurado como distribuidor o publicador para una configuración de replicación transaccional.

**Capítulo 9: Continuidad de los negocios y la seguridad con Azure SQL**

La continuidad del negocio es el asegurar que las aplicaciones críticas son resilientes a caídas de servicio (planificadas o no) que podrían resultar en pérdidas temporales o permanentes de las funcionalidades de negocio. Su meta es el diseño y deploy del negocio crítico de manera que las caídas de servicio tendrán minimizado su impacto en el negocio. Sus puntos fuertes de discusión son:

* Objetivo del tiempo de recuperación: El máximo tiempo de caída permitido para la aplicación, en el cual el negocio puede sufrir pérdidas monetarias.
* Objetivo del punto de recuperación: La máxima cantidad permitida de pérdida de datos antes de estar disponible para la aplicación.
* Tiempo estimado de recuperación: La duración estimada para que la base de datos vuelva a estar disponible después de una recuperación o failover.

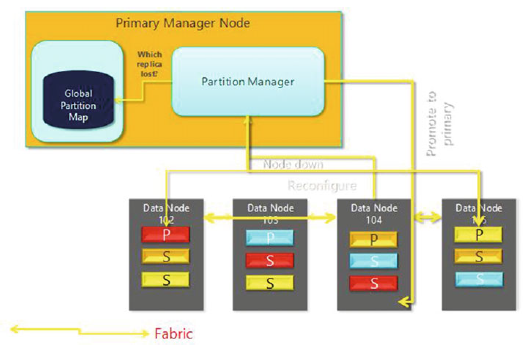
Cuando se diseña aplicaciones para la continuidad del negocio, deben considerar los tipos de caídas de servicio que provocan el fallo de la aplicación. Estos son los errores humanos, las caídas del sitio de información e incluso los procesos de actualización y mantenimiento.

CONTINUIDAD DEL NEGOCIO Y RECUPERACIÓN POSTERIOR AL DESASTRE

REDUNDANCIA LOCAL

Por defecto, Azure SQL provee dos copias secundarias de la base de datos dentro del mismo centro de datos. Estas copias se sincronizan con la copia primaria de la base de datos, y además todas las operaciones de lectura/escritura se realizan sobre la copia primaria, para posteriormente ser replicada a las copias secundarias.

Azure presenta una copia lógica transparente de la base de datos para los usuarios finales. Si una de las copias falla, Azure verifica que otra copia es creada para mantener las tres copias de esa base de datos. Azure hace uso del administrador de particiones y del mapa de particiones globales para dicho fin, en cualquier punto del tiempo.



Lo que se ilustra en la figura anterior es un caso de fallo del nodo que contiene la copia primaria, en donde el administrador de particiones de Azure inicializa un algoritmo de failover en el cual una de las copias secundarias se convierte en una nueva copia primaria. Posteriormente, se repone la copia secundaria a través de la sincronización con su copia primaria.

RESTAURACIÓN EN UN PUNTO DEL TIEMPO

El servicio de Azure SQL DB provee capacidades de respaldo automático para todas las bases de datos. La retención de estos respaldos depende del eslabón de servicio en el que se ejecuta la base de datos. La base de datos puede ser restaurada en cualquier punto del tiempo durante el período de retención del respaldo.

Considere estos puntos clave respecto a la operación de restauración:

* Restore crea una nueva base de datos en el mismo servidor SQL lógico y con el eslabón de servicio, el cual fue el medio utilizado durante el punto de restauración.
* El tiempo de restauración depende de múltiples factores, tales como el tamaño de la base de datos, el punto de recuperación, el número de respaldos a ser restaurados, entre otros.
* Una vez que la base de datos es restaurada, será cargada al completo de acuerdo al eslabón de servicio y nivel de rendimiento usados.

Si una base de datos fue eliminada, el último respaldo de esa base de datos se retiene de acuerdo a las políticas de retención. Una base de datos eliminada puede restaurarse al punto en donde se dio el eliminado de la misma.

GEORESTAURACIÓN

La funcionalidad GeoRestore de Azure permite que una base de datos pueda ser restaurada a partir de la copia georedundante. Azure realiza respaldo automatizado de la base de datos que va al almacenamiento georedundante, y GeoRestore utiliza esas copias con el fin de restaurar la BD en caso de algún desastre que impacte el sitio primario de información.

GEOREPLICACIÓN

Provee la habilidad de crear réplicas secundarias geográficamente disparatadas de la base de datos primaria. A diferencia de las tecnologías de alta disponibilidad sobre premisa, como AlwaysOn en SQL Server, la georeplicación es por su naturaleza siempre asíncrona. Las transacciones en la copia primaria se envían a la segunda copia de forma asíncrona.

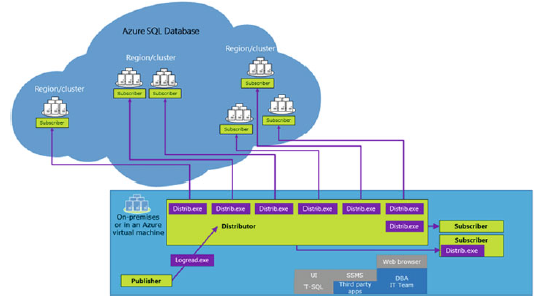
La georeplicación estándar crea una réplica secundaria sin atributos de lectura a una región DR Pair diseñada por Microsoft. Se requiere un failover manual para que sea accesible al usuario.

La georeplicación activa, mientras que se emplea la misma metodología de su contraparte estándar, crea réplicas secundarias con atributos de lectura, además de que cada centro de datos tiene capacidad para un máximo de cuatro réplicas secundarias, sin importar la región DR Pair a la que pertenece.

REPLICACIÓN EN SQL SERVER

La replicación transaccional de SQL Server se da entre SQL Server sobre premisa o ejecutándose sobre una máquina virtual de Azure, y una base de datos Azure SQL. Existen consideraciones clave a tomar en cuenta a la hora de establecer el proceso de replicación entre el publicador de SQL Server y el suscriptor de Azure SQL:

* El publicador y el distribuidor deben ser ya sea instancias de SQL Server sobre premisa o de SQL Server sobre una máquina virtual de Azure.
* El suscriptor (base de datos de Azure SQL) debería estar en una suscripción push, lo que significa que el agente de distribución se ejecutará en el servidor de distribución.
* Todo el monitoreo y administración del proceso de replicación debe llevarse a cabo desde el servidor publicador.



AZURE SQL: SEGURIDAD Y AUDITORÍA

Las bases de datos en Azure SQL proveen un conjunto de características de seguridad para asegurar que los datos del usuario localizados en Azure no se vean afectados por ningún ataque. La seguridad multicapa disponible en Azure SQL, la cual incluye autorización basada en roles similar a la presente en SQL Server, además de la encriptación de datos en modo activo o en reposo, enmascarado de datos para restringir su acceso y la seguridad a nivel de filas, está diseñada para proveer protección completa contra cualquier amenaza real o percibida.

ADMINISTRACIÓN DEL FIREWALL

La primera capa de seguridad la provee el firewall de la base de datos de Azure SQL, el cual bloquea todas las conexiones no autorizadas a esa base de datos. A través del portal de Azure o de PowerShell se puede configurar la dirección IP o el rango de direcciones IP desde las cuales se les permite la conexión a la base de datos de Azure SQL, restringiendo de forma automática cualquier conexión que no esté en la lista blanca.

AUTENTICACIÓN Y AUTORIZACIÓN

Azure SQL soporta autenticación tanto de SQL como del Directorio Activo de Azure, además de autorización de acceso por roles de usuario. SQL provee roles a nivel de servidor y/o de base de datos, administrables a través del portal de Azure, línea de comandos en PowerShell o de SSMS.

SEGURIDAD A NIVEL DE FILAS

La seguridad a nivel de filas (RLS) controla el acceso a cada una de las filas de la tabla. Este control de acceso se implementa con predicados de seguridad en la base de datos. Ya que la lógica del control de acceso está disponible en la base de datos, provee un mecanismo de seguridad muy confiable y robusto.

Ejemplo del uso de RLS sobre la tabla EmployeePerformanceData:

CREATE USER GeneralManager WITHOUT LOGIN;

CREATE USER Manager1 WITHOUT LOGIN;

CREATE USER Manager2 WITHOUT LOGIN;

CREATE TABLE EmployeePerformanceData

(

EmployeeID int,

EmployeeName varchar(200),

ManagerName sysname,

EmployeeRating int,

EmployeeIncrementPercent float

);

INSERT into EmployeePerformanceData values

(10, 'Employee10', 'Manager2', 1,10.00),

(11, 'Employee11', 'Manager2', 3, 6.53),

(12, 'Employee12', 'Manager1', 2, 8.71),

(13, 'Employee13', 'Manager2', 3, 6.25),

(14, 'Employee14', 'Manager1', 3, 5.87),

(15, 'Employee15', 'Manager2', 5, 0.00);

SELECT \* FROM EmployeePerformanceData;

GRANT SELECT ON EmployeePerformanceData TO GeneralManager;

GRANT SELECT ON EmployeePerformanceData TO Manager1;

GRANT SELECT ON EmployeePerformanceData TO Manager2;

-- If any of the users select data from the table at this point, they would see all 6 records

EXECUTE AS USER = 'GeneralManager';

SELECT \* FROM EmployeePerformanceData;

REVERT;

EXECUTE AS USER = 'Manager1';

SELECT \* FROM EmployeePerformanceData;

REVERT;

EXECUTE AS USER = 'Manager2';

SELECT \* FROM EmployeePerformanceData;

REVERT;

--- Implement RLS using Security Predicates and Filters

/\*

In this case we are creating a security predicate such that the managers

can only their own Employee Data and the GM can see all the employee

information.

\*/

CREATE SCHEMA Security;

GO

CREATE FUNCTION Security.fn\_securitypredicate(@ManagerName AS sysname)

RETURNS TABLE

WITH SCHEMABINDING

AS

RETURN SELECT 1 AS fn\_securitypredicate\_result

WHERE @ManagerName = USER\_NAME() OR USER\_NAME() = 'GeneralManager';

-- Tie the Security Predicate with the User Table

CREATE SECURITY POLICY SalesFilter

ADD FILTER PREDICATE Security.fn\_securitypredicate(ManagerName)

ON dbo.EmployeePerformanceData

WITH (STATE = ON);

-- Now if we execute the Queries, each manager would only see their own

employee information.

EXECUTE AS USER = 'GeneralManager';

SELECT \* FROM EmployeePerformanceData;

REVERT;

EXECUTE AS USER = 'Manager1';

SELECT \* FROM EmployeePerformanceData;

REVERT;

EXECUTE AS USER = 'Manager2';

SELECT \* FROM EmployeePerformanceData;

REVERT;

-- Disable the security Policy

ALTER SECURITY POLICY SalesFilter

WITH (STATE = OFF);

ENMASCARADO DINÁMICO DE DATOS

El enmascarado de datos previene la exposición y/o el acceso no autorizado a los datos más vulnerables antes de presentar la base de datos al usuario final. El enmascarado de datos se configura definiendo las políticas de seguridad dentro de la definición tabla/objeto y utilizando los permisos de enmascarado y desenmascarado para controlar la visibilidad de los datos. Los dueños y administradores de la base de datos tendrán acceso absoluto a los datos no enmascarados.

Ejemplo de enmascarado de datos:

CREATE TABLE Employee

(

EmployeeID int IDENTITY PRIMARY KEY,

FirstName varchar(100) MASKED WITH (FUNCTION = 'partial(1,"XXXXXXX",0)')

NULL,

LastName varchar(100) NOT NULL,

Phone# varchar(12) MASKED WITH (FUNCTION = 'default()') NULL,

Email varchar(100) MASKED WITH (FUNCTION = 'email()') NULL,

Salary float Masked with (Function='random(1,7)') Null

);

INSERT Employee (FirstName, LastName, Phone#, Email, Salary) VALUES

('Roberto', 'Tamburello', '555.123.4567', 'RTamburello@contoso.

com',100000.00),

('Janice', 'Galvin', '555.123.4568', 'JGalvin@contoso.com.co',200000.00),

('Zheng', 'Mu', '555.123.4569', 'ZMu@contoso.net',100000.00),

('Bill', 'Anderson', '555.123.4570', 'billand@contoso.net',150000.00),

('Graham', 'Scott', '555.123.4571', 'Grahamsco@contoso.net',120000.00);

SELECT \* FROM Employee;

CREATE USER AppUser WITHOUT LOGIN;

GRANT SELECT ON Employee TO AppUser;

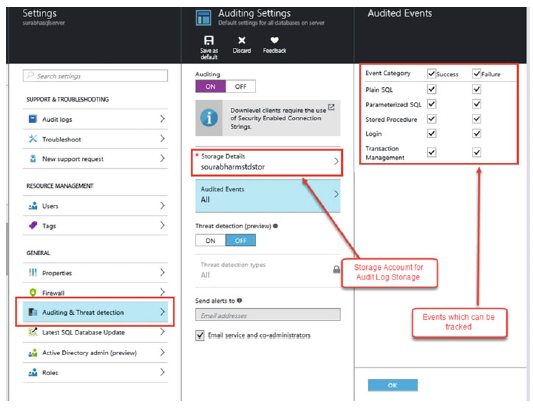
EXECUTE AS USER = 'AppUser';

SELECT \* FROM Employee;

REVERT;

AUDITORÍA DE LA BASE DE DATOS SQL

Provee la habilidad de identificar los eventos clave dentro de la base de datos y los almacena en el servicio de almacenamiento de Azure. Estos pueden ser utilizados para requerimientos regulatorios y para proveer una prueba de rendimiento de las actividades en la base de datos.



DETECCIÓN DE AMENAZAS A LA BASE DE DATOS SQL

Provee un mecanismo para detectar y responder ante las amenazas potenciales (actividades anómalas) a la base de datos SQL. Usando una combinación de detección y auditoría de amenazas, los usuarios podrán investigar y tomar acciones necesarias ante cualquier amenaza a la base de datos.

CARACTERÍSTICAS DE ENCRIPTACIÓN

* ENCRIPTACIÓN DE CONEXIONES: Azure SQL permite a los usuarios hacer uso de una conexión SSL encriptada a la base de datos.
* ENCRIPTACIÓN TRANSPARENTE DE DATOS: Provee una forma de encriptar los datos en estado de reposo. Azure SQL usa la misma tecnología para lograr un encriptado de datos cuando la base de datos se encuentra en reposo. Se habilita con el código Alter Database [MyDatabase1] set Encryption On, a través del portal de Azure o línea de comandos en PowerShell.
* ENCRIPTACIÓN A NIVEL DE CELDAS: Este tipo de encriptación se logra con la combinación de llaves simétricas y las funciones de encriptado y desencriptado disponibles en SQL.
* SIEMPRE ENCRIPTADO: Es una nueva característica disponible a partir de Azure SQL y SQL Server 2016, y permite a los clientes administrar la encriptación de datos sensibles dentro de la aplicación, sin necesidad de revelar la lógica de encriptación a la capa de la base de datos. Esta característica soporta encriptación determinística y encriptación aleatoria. La determinística generará siempre el mismo resultado encriptado para el mismo texto, mientras que la aleatoria generará valores distintos cada vez que se va a encriptar.

Ejemplo de una forma de uso:

USE [MyDatabase1]

GO

CREATE COLUMN MASTER KEY [CMKey]

WITH

(

KEY\_STORE\_PROVIDER\_NAME = N'AZURE\_KEY\_VAULT',

KEY\_PATH = N'https://<KeyVaultName>.vault.azure.net/keys/

AlwaysEncryptedkey'

)

GO

CREATE COLUMN ENCRYPTION KEY [CEKey]

WITH VALUES

(

COLUMN\_MASTER\_KEY = [CMKey],

ALGORITHM = 'RSA\_OAEP',

ENCRYPTED\_VALUE =

0x016E000001630075007200720065006E00740075007300650072002F006D0079002F0034

0066003300650037006600640037003600310031003300360034003400610036003100630

031003300640066003900370030006200320033006200370061003200620063006600390

06600380066000DEF701B5FAB3F23266DADCAAE7B448122BA75BF1841DEF7143A45C16D

37AA4AC57799D50596BA92C0406CC30A3D755D6F5D260DCCA42BB9926136985A7CCF4537

B85330DA7C1B12047048A51B04A352F6C3E71BEFEAE777019506D11AC71AF8A7AEC4DE7F

5B98ACF6EF7D56B0706E0D521533514335E500E476C6B1777212CE043BDD09B20BB97B5C7

31CB4D58BF8DDA38A7DF08EECE797DCC15A9E25B064003DE869F6D87B75A3F6625A0162

92C3B8D8F8D3876DE62DDEE57F7BC2C901E3A2097B8E050862BEA0E33EF434D2DED6D5F2

E54745D6E5C616932C5F2144B623C48B7643EDECE4CA545C31AB23DD2DFDF8067D25C05E

F1786CCBC110E005D1567B53D6E34ACCC02052F6E9AE7365DE30856EF9DB5EC4315770

D255FA76A9865204E8FBE5419AB5836480DE8345141073EB113E012CBF7132DCC22

A3A32B6E44B961DDE2B0E7F24733062412CEF9C1A0DC96976A97D48EE5DCE4F5AE12

13E680A31ADDFD9344A004ED59C6168CB7D5C8E42A22676A7D64F59A4C1687C61B5F603

49699A45D11B8EE7DC8DBB61A156AE70449483D93073497B23597A5F340A98FB7BD37D9

DC926360E32F927BB672F6BE1FFC5C01760827AF24B603E184479905BA5DFA9C23E523

182F7C5C8ABC53E5D6E6CB3806C5707EDBB7CAC3DE50DA4A2FC38D27EE65F2638FFF

37483ABC1050EEAD835919B384BB9136C0F24A6BD9489910

)

GO

CREATE TABLE dbo.EncryptedTable

(

ID INT IDENTITY(1,1) PRIMARY KEY,

LastName NVARCHAR(32) COLLATE Latin1\_General\_BIN2

ENCRYPTED WITH

(

ENCRYPTION\_TYPE = DETERMINISTIC,

ALGORITHM = 'AEAD\_AES\_256\_CBC\_HMAC\_SHA\_256',

COLUMN\_ENCRYPTION\_KEY = [CEKey]

) NOT NULL,

Salary INT

ENCRYPTED WITH

(

ENCRYPTION\_TYPE = RANDOMIZED,

ALGORITHM = 'AEAD\_AES\_256\_CBC\_HMAC\_SHA\_256',

COLUMN\_ENCRYPTION\_KEY = [CEKey]

) NOT NULL

);

GO