INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

LICENCIATURA EN CIENCIA DE DATOS

UNIDAD DE APRENDIZAJE

BASES DE DATOS AVANZADAS

AZURE COSMOS DB

NOMBRE DE LOS ALUMNOS:

DE LUNA OCAMPO YANINA

HUGO LÓPEZ MIGUEL

PROFESOR:

BOTELLO CASTILLO ALEJANDRO

GRUPO:

4CDV1

FECHA DE ENTREGA:

10 de junio de 2022

ÍNDICE

Introducción 3
Características principales 3
- Distribución global 3
- Múltiples modelos de datos y API's 4
- Rendimiento y almacenamiento 5
- Alta disponibilidad y tiempo de respuesta 5
- Cinco modelos de consistencia 5
- Latencia 5

Conceptos básicos 6
- Distribución global 6
  - Escribir y leer regiones 6
- Modelos de consistencias 6
  - Alcance de la coherencia 7
  - Modelo de consistencia fuerte 7
  - Modelo de consistencia eventual 7
  - Modelo de consistencia de estancamiento acotado 8
  - Modelo de coherencia de las sesiones 8
  - Modelo de consistencia de prefijo 8
  - Consistencia para las consultas 8

Implementación de Seguridad 9
- Cifrado en reposo 9
- Firewall Support 9
- Protección de acceso a los datos 9
  - Master Keys 9
  - Tokens de recursos 10

Protocolo de transferencia 10
Transacciones 10
APIs compatibles 11
- Azure Cosmos DB REST API 11
- API DocumentDB 12
- API de MongoDB 12
- Gráfico API 12
- Tabla API 12

Propuesta para negocios 13
Particiones 14
Resumen/Conclusión 15

Introducción

Azure Cosmos DB empezó en 2010 como un proyecto interno de Microsoft llamado "Project Florence", su objetivo era añadir algunos de los problemas que los trabajadores habían tenido que enfrentar con aplicaciones a gran escala de Internet. En 2015, el proyecto fue disponible para desarrolladores externos en Microsoft Azure y nació con el nombre de "Document DB". Finalmente en la conferencia de Microsoft Build 2017, Azure Cosmos DB fue lanzado con las capacidades existentes de DocumentDB como las distribuciones globales y la escala horizontal con poca latencia y un alto rendimiento.

Lo que resaltó de este es que soporta de forma nativa múltiples datos, como: key-value, documentos, grafos, columnar y muchas más que se están desarrollando actualmente. Esto, da la libertad de trabajar con tu data en la forma que mejor la describa. Asimismo, soporta múltiples APIs para acceder a los datos incluyendo DocumentDB SQL, MongoDB, Apache Cassandra, Graph y Table. [3]

Es un modelo global distribuído, escalable masivamente, y multi-modelo.

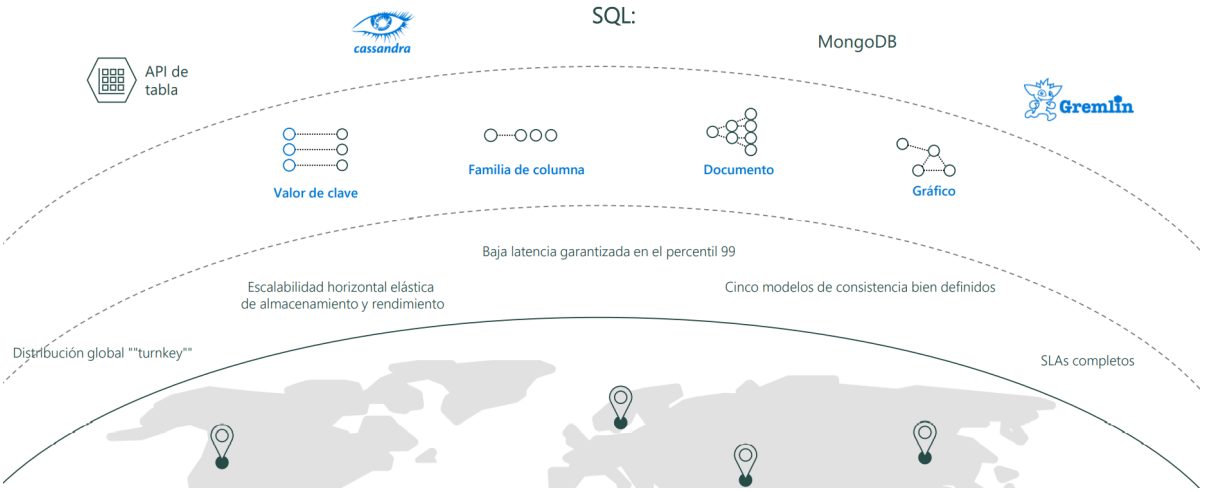


Imagen tomada de [10]

Características principales

Hay muchas características importantes dentro del producto, pero las mencionadas son las que han impulsado la implementación. Son lo que los desarrolladores de productos más tenían en mente.

La mayoría de estas características estaban presentes desde DocumentDB; sin embargo, con la evolución del producto, se introdujeron nuevas características, haciendo Azure Cosmos DB lo que es ahora, añadiendo que muchas más características están en desarrollo con el paso del tiempo. [2]

*** Distribución global**

Significa que sus bases de datos pueden ser distribuidas a través de diferentes regiones de Microsoft Azure, se puede almacenar y lo más importante, accesible para sus clientes, esta funcionalidad tiene un alto grado de automatización y rendimiento, ya que no hay necesidad de manejar complejas configuraciones, tiempo de inactividad de replicación, alta latencia o problemas de seguridad.

Usando el portal de Microsoft Azure, todo lo que se necesita hacer es seleccionar las regiones donde se distribuirá la base de datos y el portal hará el resto.

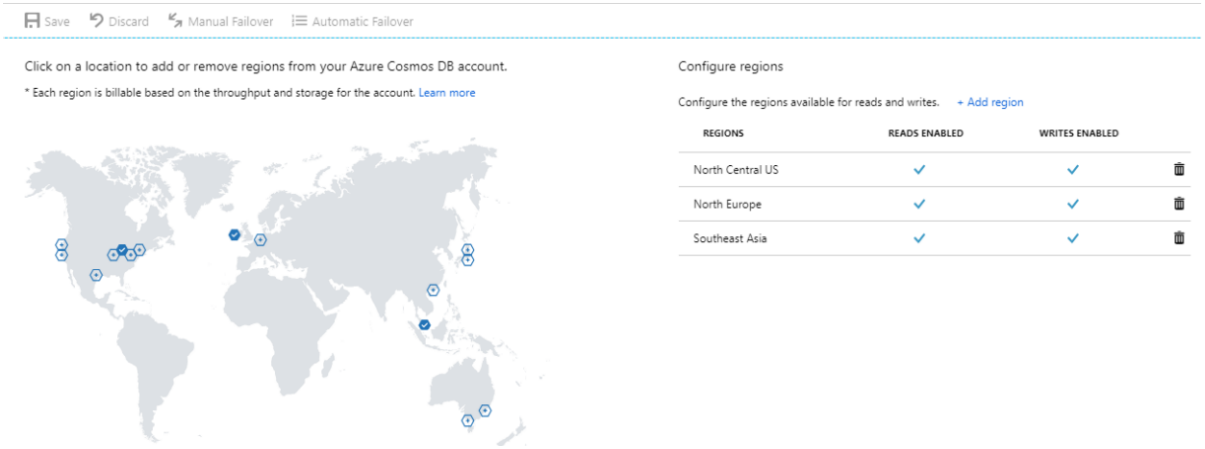


Imagen tomada de [10]

*** Múltiples modelos de datos y API's**

Puedes seleccionar el modelo de datos que mejor representa a tus datos, no hay necesidad de pensar en términos de una estructura rígida para los datos. Ilustrando de mejor forma lo dicho anteriormente, presentamos los siguientes ejemplos para ser claro en la idea planteada:

1. Suponga que desea almacenar la configuración del usuario, puede utilizar un modelo de datos de valor clave.
2. Ahora, si desea trabajar con pedidos, productos y pagos, puede utilizar un modelo de datos del documento.

Si sus datos son los mejores descritos como relaciones entre entidades, luego use un modelo de datos de gráfico. La API DocumentDB proporciona capacidades de consulta SQL conocidas o si tienes una aplicación construida en MongoDB, puedes utilizar la API MongoDB; en muchos casos no hay necesidad de reescribir la aplicación, solo cambiar la cadena de conexión. Para bases de datos de valor de clave, puede usar la API de tabla, que proporciona la misma funcionalidad que el almacenamiento de tabla de Azure pero con los beneficios del motor Azure Cosmos DB. Con la API Graph, puede usar el lenguaje de travesía de grafos de Apache TinkerPop, Gremlin, o cualquier otro sistema de gráficos compatible con Tinker Pop como Apache Spark GraphX.

Nos ayuda para escalar automáticamente los datos requeridos a todo el mundo.

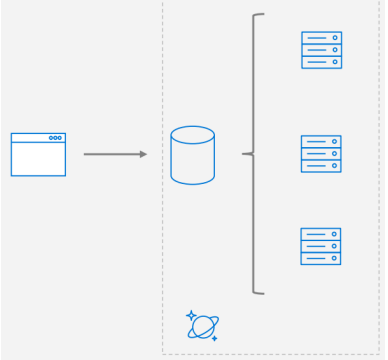


Imagen tomada de [13]

*** Rendimiento y almacenamiento**

Este se puede configurar en peticiones por segundo en función de los requisitos de su aplicación, asimismo teniendo la posibilidad de cambiar la configuración en cualquier momento.

Puede usar todo el almacenamiento que necesite, no hay ningún conflicto sobre los datos que puede almacenar. Además, escalar bases de datos sucede de forma automática en función de la configuración que establezca para la cuenta.

*** Alta disponibilidad y tiempo de respuesta**

Este tiene un SLA permanente de 99,99% de disponibilidad y una latencia en el percentil 99, independientemente de la región, proporcionando una garantía de rendimiento y consistencia.

*** Cinco modelos de consistencia**

Ofrece cinco modelos de consistencia diferentes, de fuerte consistencia tipo SQL a consistencia eventual tipo NoSQL. Todo depende sobre lo que su negocio o aplicación necesita. [4]

*** Latencia**

Baja latencia para alto rendimiento.

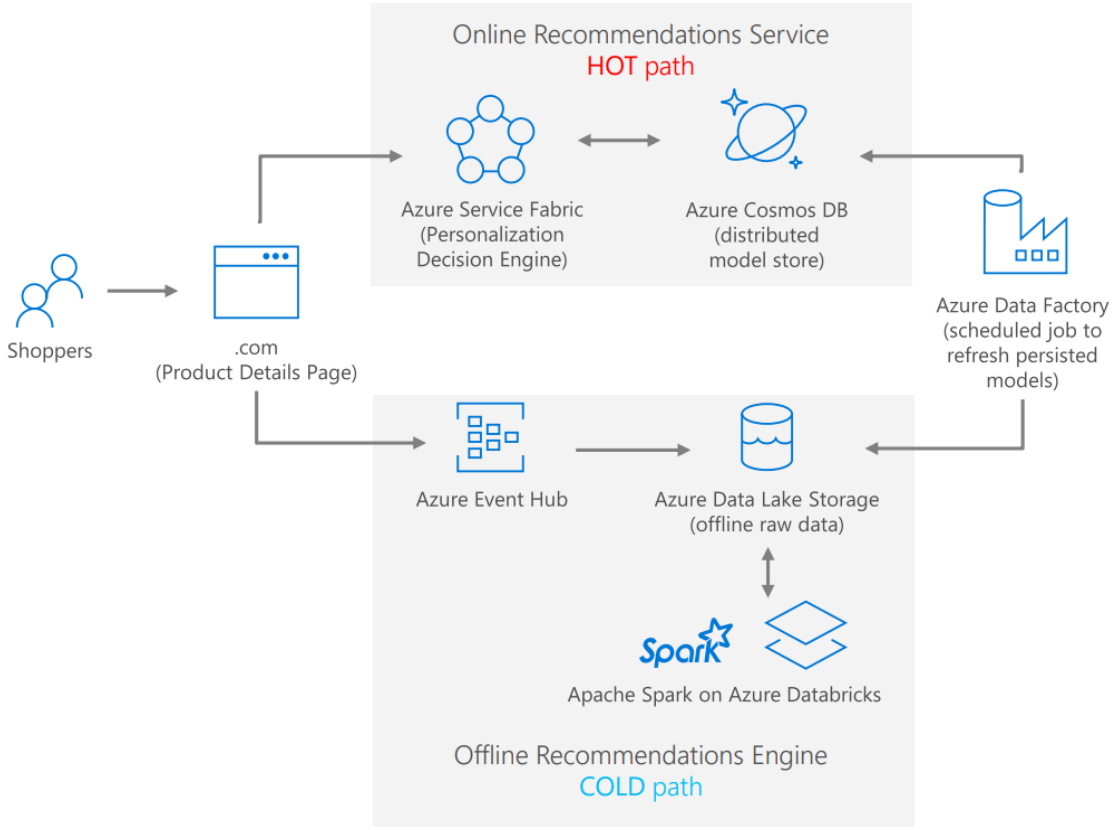


Imagen tomada de [13]

Conceptos básicos

Es muy importante entender varios conceptos clave sobre los internos de la servicio para lograr implementar y utilizar correctamente una base de datos en este servicio. La comprensión de estos conceptos es la mejor manera de aprovechar todo el potencial y capacidades de Azure Cosmos DB. Es importante saber lo que puedes hacer y que entiendas por qué las cosas funcionan de una cierta manera para implementar de forma correcta sus diferentes usos.

*** Distribución global**

Microsoft Azure está disponible a nivel mundial en más de 30 regiones, que son todas las regiones existentes y está constantemente creciendo, pero debido a que se clasifica internamente como un *Ring 0 Azure Service*, será disponible en cualquier nueva región por defecto.

Azure Cosmos DB en cuanto a las bases de datos, puede ser distribuido en las 30 regiones mencionadas para proporcionar una mayor disponibilidad, escalabilidad y rendimiento.

La distribución global es un concepto comparable a lo que representa la replicación bases de datos relacionales; la diferencia es que todo es manejado por Azure y no necesita configuraciones complejas, ya sea a nivel de base de datos o el nivel de aplicación.

**** Escribir y Leer Regiones**

Cuando la base de datos se creó por primera vez, se basaba en una sola región, que aceptaba operaciones de lectura y escritura. Cuando distribuye la base de datos a más regiones, las nuevas regiones se convierten automáticamente en regiones de lectura.

Cuando ésta nueva configuración está en su lugar, habilita la conmutación por error característica. Por defecto, ésta ocurre manualmente, lo que significa que tendrá que iniciar sesión en el portal de Azure y cambiar las lecturas a una región diferente si la región de lectura designada no está disponible. Ésta también puede ocurrir automáticamente, en cada región tiene una prioridad en la lista de regiones leídas. Si por alguna razón la región de lectura designada no está disponible, Azure cambiará a la siguiente región de lectura disponible basada en las prioridades definidas.

No se puede configurar la base de datos para que tenga más de una región, no es una característica disponible en este momento. Una configuración conocida como multi-maestro puede ser implementada, pero requiere dos bases de datos y es el más cerca de tener más de una región de escritura. Normalmente, se preferiría implementar una configuración multi-maestro para permitir escrituras en regiones donde los usuarios que crean contenido están más cerca, proporcionando una latencia aún menor.

Al tener múltiples regiones de escritura y/o lectura se necesita entender el concepto de consistencia, explicado en el siguiente punto. [8]

*** Modelos de consistencia**

La coherencia define las normas en virtud de las cuales cuando se dispone de nuevos datos en una base de datos distribuida, el modelo de consistencia

determina cuando los datos están a disposición de los usuarios para su lectura. La mayoría de los modelos de consistencia propuestos tratan de resolver solo un problema muy específico o escenario.

Azure Cosmos DB implementa cinco modelos de consistencia diferentes, con estos cinco modelos, se podrá ser capaz de determinar el modelo más adecuado para la aplicación en función de la disponibilidad y la latencia.
Al decidir qué modelo de consistencia utilizar, es necesario entender que todos ellos están vinculados a elementos tales como el rendimiento y latencia. En un extremo es fuerte consistencia, que proporcionará mayor latencia garantizando lecturas consistentes en todas las regiones. En el otro extremo, la consistencia eventual, que proporcionará la latencia más baja a un costo de una alta probabilidad de no mostrar los últimos datos al leer de diferentes regiones.
Los otros tres modelos de consistencia proporcionan valores entre estos extremos para la latencia y el rendimiento, dependerá de lo que su aplicación necesita.

**** Alcance de la coherencia**

La granularidad de la consistencia se ajusta a una sola solicitud del usuario. Una solicitud de escritura puede corresponder a un insertar, reemplazar, actualizar o eliminar una transacción. El usuario puede ser requerido para escribir sobre un gran conjunto de resultados, que abarca múltiples particiones, pero cada transacción de lectura es a una sola página y servido desde dentro de una sola partición.

**** Modelo de consistencia fuerte**

Una cuenta de Azure Cosmos DB con un modelo de consistencia fuerte garantiza que cualquier lectura de un artículo devolverá la versión más reciente de dicho artículo. Esto es importante porque es el mismo modelo de consistencia típicamente implementado en sistemas de bases de datos relacionales.
Porque estamos trabajando en un entorno distribuido, la consistencia fuerte garantiza que una operación de escritura es visible sólo después de que la mayoría de las réplicas se han cometido de forma duradera con la escritura.
Cuando se configura una consistencia fuerte para la base de datos Azure Cosmos, lee solo tan rápido como la latencia entre todas las regiones involucradas en la escritura. Debido a esto, una cuenta con fuerte consistencia sólo puede ser asociada a una región de Azure. Se usa una consistencia fuerte cuando las aplicaciones no necesitan leer los datos al instante.

**** Modelo de coherencia eventual**

Al utilizar este modelo se garantiza que todas las réplicas eventualmente convergen para reflejar la escritura más reciente. En términos de consistencia de datos, este es un modelo muy débil porque los usuarios pueden leer valores que son más antiguos que los definidos por la escritura más reciente; sin embargo, ofrece la latencia más baja de todos los modelos de consistencia para ambas lecturas y escrituras. Baja latencia se logra por no requerir cada réplica (región) para leer el mismo valor después de cada escritura. Se produce la replicación de datos en el

fondo y se completará en algún momento; es solo que las lecturas de aplicaciones no se detienen hasta que todas las regiones se sincronizan.
La consistencia eventual se utiliza en muchas bases de datos NoSQL y relacionales sistemas. Es útil en escenarios donde las lecturas necesitan suceder tan pronto como sea posible, incluso si no muestran la versión más reciente de los datos. Esto solo garantiza que todas las réplicas serán consistentes en algún momento.

**** Modelo de Consistencia de Estancamiento Acotado**

Con este modelo, las lecturas pueden retrasarse escribiendo por lo más K operaciones o un intervalo de tiempo t. Para una cuenta con una sola región, K debe estar entre 10 y 1.000.000 de operaciones, y entre 100.000 y 1,000,000 operaciones si la cuenta se distribuye globalmente. Para t, los intervalos de tiempo permitidos son entre 5 segundos y 1 día para las cuentas en una región, y entre 5 minutos y 1 día para distribución global de cuentas.
Este modelo de consistencia es adecuado para aplicaciones que necesitan escritura con consistencia fuerte y baja latencia, y lecturas que son consistentes después de un número predecible de operaciones o intervalo de tiempo. Una cuenta de Azure Cosmos DB se puede distribuir globalmente a cualquier número de regiones Azure cuando se utiliza la consistencia acotada.

**** Modelo de coherencia de las sesiones**

El modelo se denomina así porque el nivel de consistencia es alcanzado en la sesión del cliente. Lo que esto significa es que cualquier lectura o escritura son siempre actuales dentro de la misma sesión y son monotónicos. Durante la vida de una sesión, cualquier escritura está inmediatamente disponible para leer y estará disponible para otras sesiones tan pronto como los datos se repliquen al resto de las regiones.
Este modelo proporciona un alto rendimiento de lectura y baja latencia. Este es el modelo de consistencia predeterminado para cualquier nueva base de datos de Azure Cosmos y puede distribuirlo a cualquier número de regiones de Azure.

**** Modelo de consistencia de prefijo**

Este modelo es similar al modelo de consistencia eventual; sin embargo, garantiza que las lecturas nunca van escritas fuera de orden.

**** Consistencia para las consultas**

Cualquier recurso definido por el usuario, de forma predeterminada, tendría que ser la misma consistencia para consultas como se definió para lecturas. Esto es posible porque los índices se actualizan de forma sincrónica en la simple acción de insertar, reemplazar o eliminar en cualquier elemento de un contenedor Azure Cosmos DB.
También puede cambiar la estrategia de actualización del índice para que sea perezoso. Esto aumenta el rendimiento de la escritura, especialmente en escenarios de datos a granel importantes donde la aplicación se utiliza principalmente para lecturas. El nivel de consistencia de una consulta específica se puede ajustar en cada una usando la API. [2]

Implementación de Seguridad

La seguridad se implementa en varios niveles. Hay una capa de seguridad a nivel de almacenamiento con la implementación de una tecnología cifrada. A nivel de red, hay un firewall para habilitar acceso solo a direcciones IP o rangos de IP especificados, y los datos siempre están cifrados durante el tránsito. En el nivel de acceso a los datos, hay una configuración con teclas y tokens para autenticar a los usuarios y proporcionar acceso a los datos. Finalmente, para una estrategia de replicación asegura que los datos nunca se pierdan.

*** Cifrado en reposo**

El término cifrado en reposo comúnmente se refiere a cifrar datos en almacenamiento permanente, como unidades de estado sólido (SSD) o unidades de disco duro (HDDs). Azure Cosmos DB almacena las bases de datos primarias en discos SSD. Los archivos adjuntos de medios, réplicas y copias de seguridad se almacenan en el almacenamiento de Azure Blob, que utiliza discos duros. El cifrado en reposo se implementa en todos los niveles para que todas las bases de datos, archivos adjuntos de medios y copias de seguridad estén cifrados. Esta función está activada de forma predeterminada y no hay controles para desactivarla. Está gestionado en su totalidad por Azure y no tiene ningún impacto en el rendimiento o disponibilidad. Una consideración importante es que esta característica se incluye sin costo.

*** Firewall Support**

Azure Cosmos DB admite el control de acceso basado en políticas y en IP. Esto funciona como un cortafuegos para las conexiones entrantes donde permite que un conjunto de direcciones IP (o rangos IP) accedan a su cuenta de Cosmos DB. De forma predeterminada, esta función está desactivada, lo que significa que cualquiera puede conectarse a la cuenta Cosmos DB, pero puede activarla para limitar el acceso de los equipos a la base de datos.

De acuerdo con la documentación de Microsoft (http://bit.ly/cosmos-db-firewall), si habilita el control de acceso IP, necesitará agregar direcciones IP específicas al portal de Azure para así mantener el acceso. Si hay solicitudes de direcciones IP fuera de la lista permitida, Azure Cosmos DB devolverá una respuesta HTTP 404 Not Found con ningún detalle. Esto asegurará que las bases de datos se mantengan ocultas del acceso no autorizado.

*** Protección de acceso a los datos**

Con Azure Cosmos DB, puede usar dos claves diferentes para autenticarse como usuarios y proporcionar acceso a los datos. Son claves maestras y tokens de recursos.

**** Master Keys**

Utiliza las claves maestras para proporcionar acceso a los recursos administrativos en la cuenta, como el acceso a bases de datos, usuarios y permisos. Estos se crean automáticamente al mismo tiempo que se crea la cuenta y puede ser regenerado en cualquier momento basado en su política de seguridad o si han sido comprometidos.

Cada cuenta de Azure Cosmos DB tiene dos claves maestras: una llave primaria maestra y una llave maestra secundaria. Las llaves primarias y secundarias funcionan exactamente de la misma manera y proporcionan acceso a los mismos recursos sin ninguna diferencia en absoluto. La idea detrás de esta implementación es que el usuario pueda regenerar o rotar las teclas sin interrumpir el acceso a la cuenta o los datos.

**** Tokens de recursos**

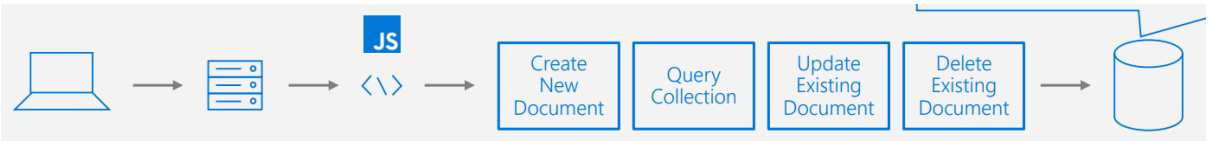
Los tokens de recursos proporcionan acceso a recursos dentro de la base de datos, como claves de partición, documentos, archivos adjuntos y procedimientos almacenados; son particularmente útil cuando se desea proporcionar acceso a un cliente que no puede con una clave maestra. Se crean cada vez que se concede a un usuario permisos a un recurso específico y recreado cuando un permiso toma acción por una petición POST, GET o PULL.

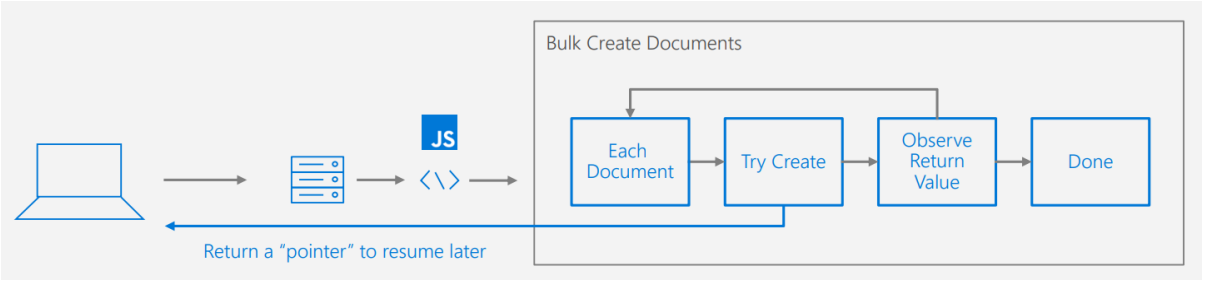
A diferencia de las teclas, los tokens de recursos no se pueden administrar en el portal de Azure. Solo pueden ser administradas mediante la API de la BD de Azure Cosmos o las bibliotecas de cliente. Este tiene un período de validez que por defecto es de una hora. Este período de validez se puede ajustar hasta cinco horas. Utiliza un token de hash específicamente diseñado para el usuario y permiso. [8]

Protocolo de transferencia

Imagen tomada de [14]

Transacciones





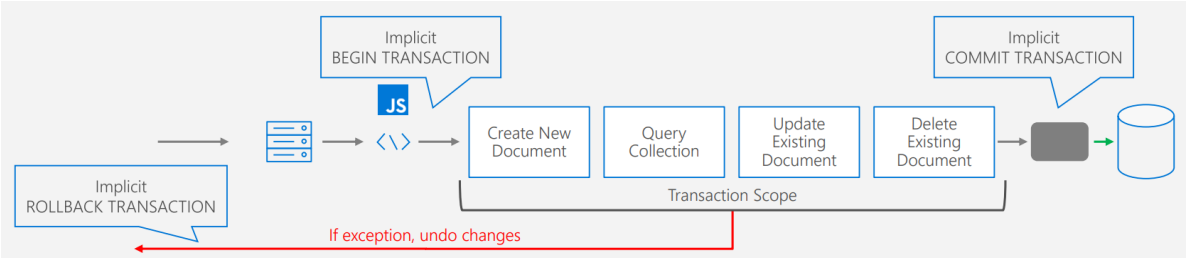


Imagen tomada de [13]

APIs compatibles

Como ya se había mencionado previamente, Azure Cosmos DB admite varias API para recursos, datos gestión y varios kits de desarrollo de software (SDK) que encapsulan la funcionalidad para ellos. En su núcleo está la API REST, que proporciona una base para todas las acciones que se pueden realizar contra una cuenta de Azure Cosmos DB. También hay otras API como DocumentDB, Mongo DB, Apache Cassandra, Tabla y Gráfico. A continuación veremos algunas de las más importantes, las no mencionadas no significa que no aporten algo, simplemente por el tipo de trabajo, se mencionan 5.

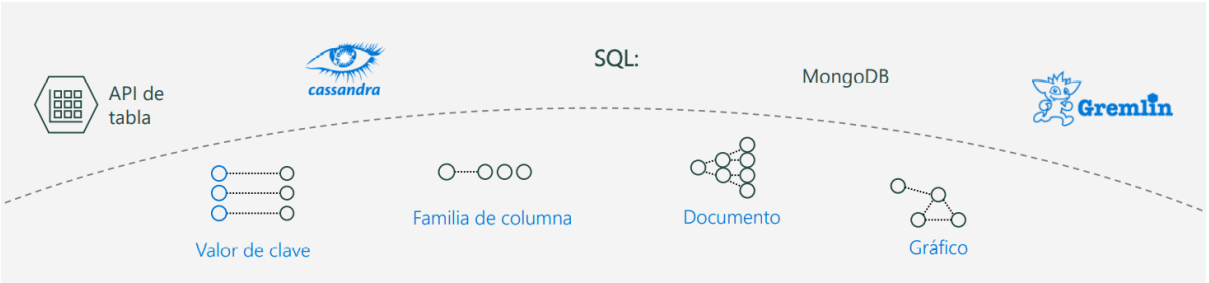


Imagen tomada de [10]

*** Azure Cosmos DB REST API**

La API REST interactúa con Azure Cosmos DB mediante el protocolo HTTP. Al igual que con cualquier API REST, los verbos HTTP se utilizan para informar qué acción realizar. En general, son los siguientes:

•POST: Crea recursos de elementos
•GET: Leer un recurso de elemento o una lista de recursos
•PUT: Reemplazar un recurso de elemento existente
•DELETE: Eliminar un recurso de elemento existente
•HEAD: Se utiliza de manera similar a GET, excepto que solo devolverá las cabeceras de respuesta

El URI de destino para la API se basa en el punto final del URI creado para la cuenta de base de datos. [8]

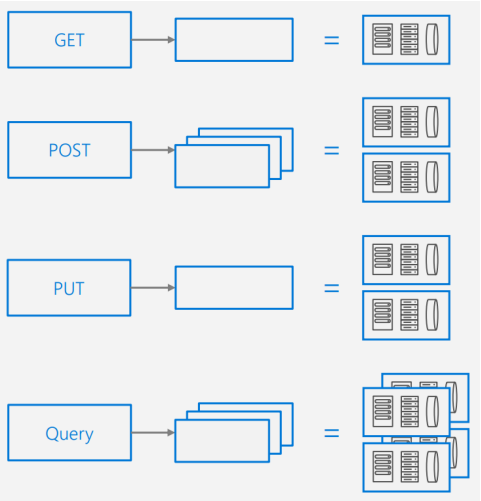


Imagen tomada de [13]

*** API DocumentDB**

Está construida sobre la API REST y está implementada en varios idiomas y plataformas incluyendo: .NET, Java, NodeJS, JavaScript y Python a través de sus respectivos SDK.

Usando la API DocumentDB puedes consultar documentos usando un SQL, sintaxis similar a la utilizada en Entity Framework, solo se extiende a la consulta Documentos JSON. También puede administrar los recursos de la cuenta y realizar acciones como crear bases de datos, colecciones, procedimientos almacenados, etc.

*** API de MongoDB**

Con la API de MongoDB, podemos aprovechar su conocimiento de MongoDB. En la mayoría de los casos, una aplicación MongoDB existente funcionará sin ningún cambio de código. Todo lo que necesita es migrar sus bases de datos a un Azure Cuenta Cosmos DB que implemente la API MongoDB y que se cambie la cadena de conexión de la aplicación; será transparente para la aplicación.

En un sentido, la aplicación pensará que está hablando con MongoDB cuando de hecho está hablando con Azure Cosmos DB. El portal de Azure también incluye funcionalidad para que pueda abrir un mongo shell donde puede consultar sus documentos como lo haría con MongoDB.

*** Gráfico API**

Una base de datos gráfica, a diferencia de una base de datos relacional, representa los datos como existe en el mundo real, como la gente, los coches, las computadoras, y así sucesivamente, que son conectados naturalmente, y no trata de cambiarlos de ninguna manera para definir como entidades. Los gráficos se componen de vértices y aristas. Ambos vértices y bordes pueden tener cualquier número de propiedades. Vértices representan específicos objetos como una persona, lugar o evento. Un borde es una relación entre vértices.

Para tener un poco más claro lo que se mencionó previamente, proporcionaremos un ejemplo con la intención de que quede de forma clara, un vértice puede ser una persona, las propiedades de este vértice son nombre, edad y género. Añadiendo otro ejemplo tenemos como vértice un teléfono, las propiedades de este vértice son

marca y OS. Una ventaja para estos vértices podría ser "una persona utiliza un teléfono."
Los gráficos son muy útiles para entender una amplia gama de conjuntos de datos en diferentes campos como la ciencia y los negocios. Las bases de datos gráficas permiten trabajar, valga la redundancia, con gráficos de forma natural y eficiente, añadiendo que por lo general son NoSQL por su capacidad para adaptarse rápidamente a esquemas nuevos o actualizados. Por eso implementarlos en Azure Cosmos DB es un ajuste natural.
Los gráficos le permiten trabajar con datos de una manera poderosa aprovechando gráficos transversales encontrados en muchos casos de uso y patrones porque superan las bases de datos SQL y NoSQL tradicionales en varios pedidos de magnitud. Azure Cosmos DB implementa bases de datos gráficas utilizando el TinkerPop estándar. Puede utilizar el lenguaje traversal Apache TinkerPop, remlin, o cualquier otro sistema gráfico compatible con TinkerPop como Apache Spark GraphX.

*** Tabla API**

Esta API implementa la misma funcionalidad de Azure Table Storage, pero con los beneficios de escalabilidad y rendimiento de Cosmos DB. Otro beneficio es que todas las propiedades están indexadas, a diferencia de Azure Table Storage, que solo indexa el PartitionKey y RowKey.
Contiene un pequeño programa que se conecta a una cuenta de Azure Cosmos BD implementando la API Table. Se encuentra la cadena de conexión en la cuenta de Azure Cosmos DB. Primero, debe decirle al SDK a qué cuenta de almacenamiento que se está conectando; lo hace con un CloudStorageAccount que toma la cadena de conexión como parámetro. A continuación, crea una Objeto CloudTableClient, que se utiliza para realizar las operaciones contra la base de datos.
Usando el objeto cliente de la tabla de la nube, se obtiene una referencia a la tabla de productos utilizando el método GetTableReference(), y se almacena en una variable CloudTable. Si la tabla no existe, se crea usando el método CreateIfNotExists(). [7]

Propuesta para negocios

Rimma Nehme, con experiencia en sistemas distribuidos, OS, sistemas de IA, hardware e internos de Azure [1], comenta: "Si usted tiene escribir pesado cargas de trabajo, que abarcan múltiples geos, y necesita esta ingestión casi en tiempo real de sus datos, esto se vuelve extremadamente atractivo para IoT, web, escenarios de juegos móviles, Cosmos DB es para usted."

Particiones

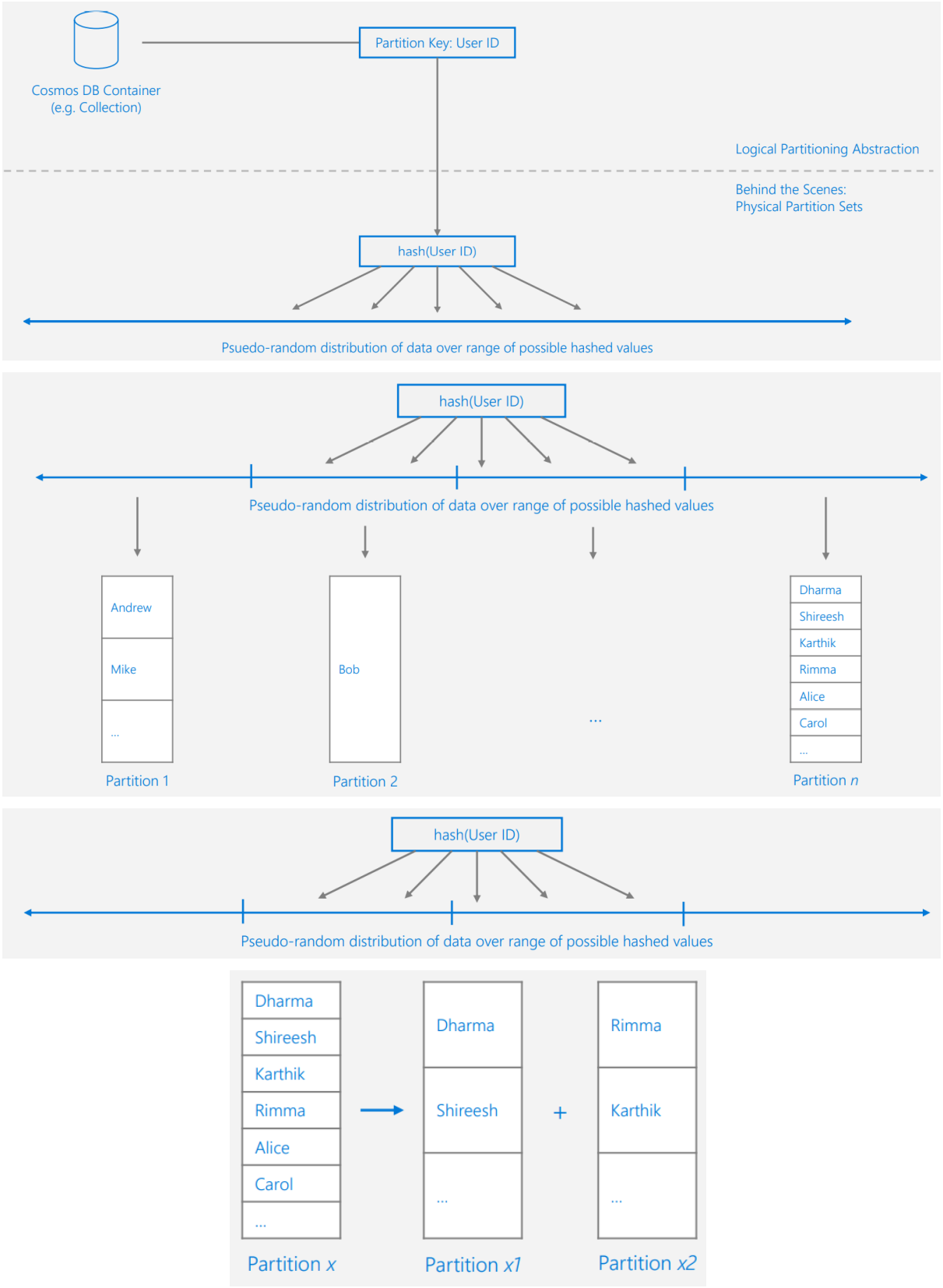


Imagen tomada de [11]

Resumen/Conclusión

Entendimos los conceptos centrales de Azure Cosmos DB. Comenzamos por entender qué es la distribución global y cómo ayuda a aprovechar la alta disponibilidad y rendimiento mediante la creación de réplicas de las bases de datos a través de múltiples regiones de Azure. Luego examinamos los diferentes modelos de consistencia, sus beneficios y cuándo usar cada uno.

Analizamos que la mayoría de las soluciones comerciales disponibles solo ofrecen dos modelos de consistencia, fuerte y eventual, pero Azure Cosmos DB ofrece tres modelos más que equilibran los requisitos entre disponibilidad y rendimiento.

Siguiendo con el concepto de partición y por qué es importante. Se estudió el concepto de contenedores y cómo ayudan a la interacción entre la aplicación y las particiones físicas. Asimismo, se vieron las consideraciones para particionado y los criterios para elegir la clave de partición correcta para la base de datos utilizada.

El siguiente concepto fue el rendimiento, una unidad de solicitud es un número normalizado para todos los diferentes modelos de datos basados en necesidades computacionales que ejecuten una operación. Esto era necesario para proporcionar una medida estándar para calcular el rendimiento y la facturación.

Comprendimos las diferentes configuraciones para asegurar el inicio de las bases de datos desde el almacenamiento con cifrado en reposo, la red con firewall y acceso a los datos con claves maestras y tokens de recursos.

Terminamos con el entendimiento de las diferentes API que pueden ser utilizadas para interactuar con Azure Cosmos DB. Se revisó cada una de ellas, a partir de la API REST y las implementaciones a su alrededor tales como la API DocumentDB. Las aplicaciones MongoDB existentes pueden trabajar sin problemas con Azure Cosmos DB con prácticamente ningún cambio en la codificación de aplicaciones. Por último se estudiaron las implementaciones de la forma gráfica y su uso.

Las organizaciones tendrán que invertir significativamente para garantizar que los principios del Reglamento general de protección de datos (RGPD) se implementen y mantengan de manera efectiva en sus entornos. Microsoft mismo está pasando por el proceso sustancial de validar que todos sus sistemas de datos cumplen con la disposición de cumplimiento del RGPD, y está abordando esta tarea metódica y meticulosamente en toda la empresa.

La plataforma Azure Cosmos DB proporciona muchas capacidades integradas para ayudar a cumplir con varios requisitos del RGPD y ayuda a facilitar significativamente este proceso. Desde controles granulares que se pueden definir, hasta la integración con servicios de gestión de autenticación centralizados y métodos líderes en la industria para proteger y mantener la disponibilidad de datos, Azure Cosmos DB - ofrece un amplio conjunto de potentes capacidades para abordar los principios de privacidad de datos en el nivel de la plataforma de datos.

Referencias

1. Guay Paz, José Rolando (2018). Microsoft Azure Cosmos DB Revealed: A Multi-Model Database Designed for the Cloud. Recuperado de: http://projanco.com/Library/Microsoft%20Azure%20Cosmos%20DB%20Revealed-%20A%20Multi-Modal%20Database%20Designed%20for%20the%20Cloud.pdf
2. Microsoft. *Azure Cosmos DB*. (s. f.). Developer tools, technical documentation and coding examples. Recuperado de: https://docs.microsoft.com/en-us/azure/cosmos-db/
3. Microsoft. *Introduction to Azure Cosmos DB*. (s. f.). Developer tools, technical documentation and coding examples. Recuperado de: https://docs.microsoft.com/en-us/azure/cosmos-db/introduction
4. Microsoft. *Understanding the differences between Azure Cosmos DB NoSQL and relational databases*. (s. f.). Developer tools, technical documentation and coding examples. Recuperado de: https://docs.microsoft.com/en-us/azure/cosmos-db/relational-nosql
5. Microsoft. *Azure Cosmos DB resource model*. (s. f.). Developer tools, technical documentation and coding examples. Recuperado de: https://docs.microsoft.com/en-us/azure/cosmos-db/account-databases-containers-items
6. Microsoft. *Query data with Azure Cosmos DB's API for MongoDB*. (s. f.). Developer tools, technical documentation and coding examples. Recuperado de: https://docs.microsoft.com/en-us/azure/cosmos-db/mongodb/tutorial-query-mongodb
7. Nehme, Rimma (2018). Linkedin profile of Product Mangager & Architect of Cosmos DB. Recuperado de: https://www.linkedin.com/in/rimma-nehme?original_referer=
8. Microsoft (2021). Azure Cosmos DB and GDPR
9. Microsoft. *Precios de Azure Cosmos DB | Microsoft Azure*. (s. f.). Cloud Computing Services. Recuperado de: https://azure.microsoft.com/es-es/pricing/details/cosmos-db
10. Azure Cosmos DB Best Practices.pdf
11. Desarrollo Nativo de Nube con AKS y Cosmos DB.pdf
12. Cosmos DB in the new emerging world - Use cases and applications.pdf
13. Schema-Agnostic Indexing with Azure DocumentDB.pdf