ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA
CARRERA DE SOFTWARE

(BASE DE DATOS 1**) (**1**)**

Informe de tipos de datos No. (2)

1. **DATOS GENERALES:**

   TEMA: Tipos de datos en SQL Server

   NOMBRE [redacted] **CODIGO(S)** [redacted]

   GRUPO No.: N/A

   FECHA DE ENTREGA: 15/04/2024

2. **OBJETIVO:**

Proporcionar una comprensión detallada de los tipos de datos disponibles en SQL Server 2022 Express, permitiendo a los usuarios seleccionar y aplicar los tipos de datos apropiados para optimizar el almacenamiento, la manipulación y la gestión de datos en sus aplicaciones de base de datos, con el fin de lograr un rendimiento óptimo, mantener la integridad de los datos y facilitar la interoperabilidad con otros sistemas y aplicaciones.

3. **INTRODUCCION:**

SQL Server 2022 Express es una edición gratuita y ligera del popular sistema de gestión de bases de datos de Microsoft. Diseñada para pequeñas aplicaciones y entornos de desarrollo, ofrece un conjunto completo de funcionalidades de bases de datos y es ideal para desarrolladores, estudiantes y pequeñas empresas. Los tipos de datos son una parte fundamental de cualquier sistema de base de datos, ya que definen cómo se almacenan, manejan y procesan los datos.

En SQL Server 2022 Express, los tipos de datos determinan el tipo y formato de los valores que se pueden almacenar en las columnas de las tablas, los tipos de datos disponibles son variados e incluyen enteros, decimales, caracteres, fechas y horas, datos binarios, tipos espaciales y más. Conocer y comprender los diferentes tipos de datos es esencial para crear bases de datos eficientes y confiables.

Esta introducción ofrece una visión general de la importancia de los tipos de datos en SQL Server 2022 Express y su papel clave en la construcción de bases de datos robustas y escalables.

4. ACTIVIDADES DESARROLLADAS

En SQL Server 2022 Express, el manejo eficaz de los datos comienza con la elección del tipo de dato adecuado para cada columna de la base de datos. A continuación, se detalla cada uno de los principales tipos de datos disponibles, junto con sus características y aplicaciones más comunes:

- **Numéricos exactos**

Los tipos de datos numéricos exactos en SQL Server 2022 Express se utilizan para almacenar valores enteros o decimales con alta precisión. Estos tipos de datos son adecuados para aplicaciones que requieren un manejo preciso de datos numéricos. A continuación, se detallan los tipos de datos numéricos exactos disponibles, junto con ejemplos de cada uno:

bigint: Este tipo de dato permite almacenar valores enteros grandes en el rango de -9,223,372,036,854,775,808 a 9,223,372,036,854,775,807. Es útil para manejar grandes cantidades de datos, como el número total de transacciones o identificadores únicos de gran tamaño.
Por ejemplo, se puede almacenar el número total de transacciones en una cadena de tiendas, como 2,500,000.
numeric: Almacena valores decimales con una precisión y escala especificadas. Es ideal para almacenar datos financieros, como precios de productos o valores monetarios, donde se requiere una precisión exacta.
Por ejemplo, se puede almacenar el precio de un artículo como 19.99 o una tasa de interés como 5.75%.
bit: Es un tipo de dato de un solo bit que almacena valores booleanos (verdadero o falso). Se utiliza para representar estados binarios, como si un producto está disponible o no.
Por ejemplo, se puede almacenar el estado de disponibilidad de un producto: verdadero para disponible y falso para no disponible.
smallint: Almacena valores enteros en el rango de -32,768 a 32,767. Es adecuado para manejar cantidades pequeñas, como la cantidad de productos en stock.
Por ejemplo, se puede almacenar la cantidad de productos en stock, como 150.
decimal: Similar a **numeric**, este tipo de dato permite almacenar valores decimales con una precisión y escala especificadas. Puede usarse para datos financieros, como salarios o tasas de interés, donde se requiere exactitud.
Por ejemplo, se puede almacenar el salario de un empleado como 3000.75.
smallmoney: Almacena valores monetarios con precisión de cuatro decimales en el rango de -214,748.3648 a 214,748.3647. Es útil para manejar transacciones financieras pequeñas.
Por ejemplo, se puede almacenar el costo de un artículo pequeño como 15.99.
int: Permite almacenar valores enteros en el rango de -2,147,483,648 a 2,147,483,647. Es adecuado para manejar cantidades enteras, como el número de clientes atendidos en un día.
Por ejemplo, se puede almacenar el número de clientes atendidos en un día como 120.

tinyint: Almacena valores enteros en el rango de 0 a 255. Es útil para manejar datos pequeños, como el número de elementos en un inventario.
Por ejemplo, se puede almacenar el número de elementos en un inventario, como 20.
money: Almacena valores monetarios con precisión de cuatro decimales en el rango de -922,337,203,685,477.5808 a 922,337,203,685,477.5807. Este tipo de dato es ideal para manejar grandes transacciones financieras y cantidades de dinero.
Por ejemplo, se puede almacenar una gran transacción financiera como 1,000,000.0000.

La elección del tipo de dato numérico exacto adecuado depende de los requerimientos específicos de la aplicación, como la precisión necesaria y el rango de valores esperados. Seleccionar el tipo de dato correcto puede ayudar a garantizar la eficiencia y precisión en el manejo de los datos.

- **Numéricos aproximados**

Los tipos de datos numéricos aproximados en SQL Server 2022 Express se utilizan para almacenar valores de punto flotante con una precisión variable. Estos tipos de datos son útiles cuando se necesita manejar valores numéricos que pueden contener decimales, pero no requieren una precisión exacta. A continuación, se describen los tipos de datos numéricos aproximados disponibles, junto con ejemplos de cada uno:

float: El tipo de dato **float** almacena valores de punto flotante con una precisión especificada. La precisión se define por el tamaño del **float** en bits. Los **float** de 32 bits tienen una precisión simple, mientras que los **float** de 64 bits tienen una precisión doble.
Por ejemplo, se puede almacenar un cálculo científico como la distancia entre dos puntos en el espacio tridimensional, como 134.56.
Otro ejemplo puede ser almacenar el resultado de un cálculo de área aproximada de una forma geométrica, como 542.87.
real: El tipo de dato real almacena valores de punto flotante con una precisión estándar (32 bits). Ofrece una precisión menor en comparación con **float** de 64 bits, pero ocupa menos espacio en memoria.
Por ejemplo, se puede almacenar la velocidad del viento en una estación meteorológica, como 12.3.
Otro ejemplo puede ser almacenar el nivel de presión atmosférica medido en una ubicación específica, como 1013.25.

De forma general, se recomienda utilizar **float** para cálculos científicos o de ingeniería que requieren una mayor precisión en el almacenamiento de valores numéricos con decimales. Por otro lado, **real** es adecuado para aplicaciones que no requieren una precisión tan alta, pero desean optimizar el uso de la memoria. Al seleccionar el tipo de dato adecuado, se debe considerar la precisión necesaria y la eficiencia en el almacenamiento de datos para lograr el mejor rendimiento posible en la base de datos.

- **Fecha y hora**

Los tipos de datos de fecha y hora en SQL Server 2022 Express se utilizan para almacenar valores temporales, como fechas, horas o combinaciones de ambas. Estos tipos de datos son útiles para aplicaciones que requieren un manejo preciso de información temporal.

A continuación, se describen los tipos de datos de fecha y hora disponibles, junto con ejemplos de cada uno:

date: Almacena solo la fecha (año, mes, día) sin información de la hora. Este tipo de dato es útil para registrar fechas importantes, como fechas de nacimiento o fechas de contratación de empleados.
Por ejemplo, se puede almacenar la fecha de nacimiento de un cliente como '1990-05-21'.
datetimeoffset: Almacena valores de fecha y hora con precisión de nanosegundos, además de una zona horaria. Este tipo de dato es útil para aplicaciones globales que necesitan manejar diferentes zonas horarias.
Por ejemplo, se puede almacenar una transacción que ocurrió en una ubicación específica con una zona horaria particular, como '2024-04-13 10:15:30.1234567 +02:00'.
datetime2: Almacena valores de fecha y hora con una precisión de hasta nanosegundos y un rango de valores más amplio en comparación con **datetime**. Es útil para aplicaciones que requieren una alta precisión en el manejo de datos temporales.
Por ejemplo, se puede almacenar la hora exacta de un evento programado con precisión de nanosegundos, como '2024-04-13 10:15:30.1234567'.
smalldatetime: Almacena valores de fecha y hora con precisión de minuto. Este tipo de dato es útil para datos históricos o registros de eventos pasados.
Por ejemplo, se puede almacenar la fecha y hora de inicio de un evento pasado como '2024-04-13 10:15'.
datetime: Almacena valores de fecha y hora con precisión de tres milisegundos. Es adecuado para registrar la fecha y hora exactas de eventos, transacciones o cualquier actividad que requiera una precisión razonable.
Por ejemplo, se puede almacenar la fecha y hora de una transacción como '2024-04-13 10:15:30.123'.
time: Almacena solo la hora (hora, minuto, segundo, y precisión hasta nanosegundos). Es útil para registrar horarios específicos, como la hora de inicio o finalización de turnos de trabajo.
Por ejemplo, se puede almacenar la hora de inicio de un turno de trabajo como '10:15:30'.

De forma general, los tipos de datos de fecha y hora deben seleccionarse según la precisión y rango de valores necesarios para la aplicación. El tipo adecuado asegurará un manejo preciso de la información temporal y una optimización en el uso de recursos de la base de datos.

- **Cadenas de caracteres**

Los tipos de datos de cadenas de caracteres en SQL Server 2022 Express se utilizan para almacenar texto o información de tipo alfanumérico. Estos tipos de datos son útiles para manejar datos como nombres, descripciones o direcciones. A continuación, se describen los tipos de datos de cadenas de caracteres disponibles, junto con ejemplos de cada uno:

char: Almacena cadenas de caracteres de longitud fija. El tamaño de la cadena se especifica al definir el tipo de dato y se rellena con espacios en blanco si el valor almacenado es más corto que la longitud especificada. Este tipo de dato es útil cuando se espera que las cadenas tengan una longitud constante.
Por ejemplo, se puede almacenar un código de producto de longitud fija, como 'ABC123', definiendo **char(6)** para asegurarse de que siempre ocupe 6 caracteres.

varchar: Almacena cadenas de caracteres de longitud variable. A diferencia de **char**, **varchar** solo ocupa el espacio necesario para los caracteres almacenados, lo que puede ayudar a optimizar el uso de almacenamiento. Este tipo de dato es adecuado para almacenar nombres, descripciones o cualquier otra información que pueda variar en longitud.
Por ejemplo, se puede almacenar el nombre de un cliente, como 'Juan Pérez', en un **varchar(50)** para permitir hasta 50 caracteres.
text: Almacena cadenas de texto de longitud variable de hasta 2 GB. Este tipo de dato es útil para manejar grandes cantidades de texto, como descripciones largas o contenido de documentos. Sin embargo, se desaconseja su uso en aplicaciones nuevas, ya que **varchar(max)** ofrece una funcionalidad similar y es más eficiente.
Por ejemplo, se puede almacenar la descripción detallada de un producto o artículo, como una descripción larga del contenido de un libro.

En general, se recomienda utilizar **varchar** para la mayoría de los casos de uso, ya que ofrece flexibilidad en términos de longitud variable y eficiencia en el almacenamiento. El tipo **char** puede ser útil cuando se espera una longitud constante para las cadenas de caracteres. Por último, **text** es adecuado para manejar grandes cantidades de texto, aunque **varchar(max)** es generalmente una mejor opción para aplicaciones nuevas.

- **Cadenas de caracteres Unicode**

Los tipos de datos de cadenas de caracteres Unicode en SQL Server 2022 Express se utilizan para almacenar texto en formatos que admiten caracteres Unicode. Unicode es un estándar que permite representar caracteres de diferentes idiomas y símbolos especiales. Estos tipos de datos son útiles para manejar datos de texto que pueden incluir caracteres de diferentes alfabetos o símbolos internacionales. A continuación, se describen los tipos de datos de cadenas de caracteres Unicode disponibles, junto con ejemplos de cada uno:

nchar: Almacena cadenas de caracteres Unicode de longitud fija. El tamaño de la cadena se especifica al definir el tipo de dato y se rellena con espacios en blanco si el valor almacenado es más corto que la longitud especificada. Es útil para almacenar texto de longitud constante, especialmente cuando se prevé la necesidad de admitir caracteres Unicode.
Por ejemplo, se puede almacenar un código de producto de longitud fija en caracteres Unicode, como '产品123', definiendo **nchar(6)** para asegurar que ocupe siempre 6 caracteres.
nvarchar: Almacena cadenas de caracteres Unicode de longitud variable. Al igual que varchar, este tipo de dato solo ocupa el espacio necesario para los caracteres almacenados, lo que ayuda a optimizar el uso de almacenamiento. Es adecuado para almacenar nombres, descripciones o cualquier información que pueda variar en longitud y que pueda contener caracteres Unicode.
Por ejemplo, se puede almacenar un nombre de cliente en un idioma diferente, como 'François Dupont', en un **nvarchar(50)** para permitir hasta 50 caracteres.
ntext: Almacena cadenas de texto Unicode de longitud variable de hasta 2 GB. Este tipo de dato es útil para manejar grandes cantidades de texto Unicode, como descripciones largas o contenido de documentos. Sin embargo, se desaconseja su uso en aplicaciones nuevas, ya que **nvarchar(max)** ofrece una funcionalidad similar y es más eficiente.
Por ejemplo, se puede almacenar un documento de texto extenso en varios idiomas, como una traducción de un libro, utilizando **ntext** para manejar la longitud necesaria.

En general, **nvarchar** es el tipo de dato recomendado para la mayoría de los casos de uso que involucran texto Unicode, ya que ofrece flexibilidad en términos de longitud variable y eficiencia en el almacenamiento. El tipo **nchar** puede ser útil cuando se espera una longitud constante para las cadenas de caracteres Unicode. Por último, **ntext** es adecuado para manejar grandes cantidades de texto Unicode, aunque **nvarchar(max)** es generalmente una mejor opción para aplicaciones nuevas.

- **Cadenas binarias**

Los tipos de datos de cadenas binarias en SQL Server 2022 Express se utilizan para almacenar datos binarios, como imágenes, archivos o cualquier otro tipo de información en formato binario. Estos tipos de datos son útiles para manejar datos que no se pueden representar como texto, como multimedia o archivos de gran tamaño. A continuación, se describen los tipos de datos de cadenas binarias disponibles, junto con ejemplos de cada uno:

binary: Almacena cadenas de datos binarios de longitud fija. El tamaño de los datos se especifica al definir el tipo de dato y se rellena con ceros si los datos almacenados son más cortos que la longitud especificada. Es útil para almacenar datos binarios de longitud constante.
Por ejemplo, se puede almacenar un identificador único de longitud fija en formato binario, como un identificador de producto, definiendo **binary(16)** para asegurar que ocupe siempre 16 bytes.
varbinary: Almacena cadenas de datos binarios de longitud variable. Al igual que **varchar** para cadenas de caracteres, **varbinary** solo ocupa el espacio necesario para los datos almacenados, lo que ayuda a optimizar el uso de almacenamiento. Es adecuado para almacenar imágenes, archivos o cualquier otro tipo de datos binarios que puedan variar en tamaño.
Por ejemplo, se puede almacenar la imagen de un producto como un archivo binario, definiendo **varbinary(max)** para permitir un tamaño variable y manejar imágenes de diferentes resoluciones.
image: Almacena grandes cantidades de datos binarios de hasta 2 GB. Este tipo de dato es adecuado para manejar imágenes, videos o archivos de gran tamaño. Sin embargo, se desaconseja su uso en aplicaciones nuevas, ya que **varbinary(max)** ofrece una funcionalidad similar y es más eficiente.
Por ejemplo, se puede almacenar un archivo de video promocional de un producto utilizando **image** para manejar el tamaño necesario.

En general, **varbinary** es el tipo de dato recomendado para la mayoría de los casos de uso que involucran datos binarios, ya que ofrece flexibilidad en términos de longitud variable y eficiencia en el almacenamiento. El tipo **binary** puede ser útil cuando se espera una longitud constante para los datos binarios. Por último, **image** es adecuado para manejar grandes cantidades de datos binarios, aunque **varbinary(max)** es generalmente una mejor opción para aplicaciones nuevas.

- **Otros tipos de datos**

SQL Server 2022 Express ofrece una variedad de otros tipos de datos que son útiles para manejar datos especializados en aplicaciones de bases de datos. Estos tipos de datos

abarcan desde identificadores únicos hasta datos de geometría espacial. A continuación, se describen los tipos de datos disponibles, junto con ejemplos de cada uno:

cursor: Representa un cursor, que es un conjunto de datos obtenidos de una consulta que permite la navegación y manipulación de los datos una fila a la vez. Este tipo de dato se utiliza en procedimientos almacenados o funciones para trabajar con conjuntos de datos.
Por ejemplo, se puede utilizar un cursor para recorrer los resultados de una consulta y actualizar los registros uno por uno.
rowversion: Un tipo de dato que almacena un número de versión único y automático para cada fila en una tabla. Es útil para el control de concurrencia optimista, donde se asegura que los datos no hayan cambiado desde la última lectura.
Por ejemplo, se puede utilizar **rowversion** para verificar si un registro ha sido modificado desde la última vez que se leyó antes de realizar una actualización.
hierarchyid: Un tipo de dato que representa una posición en una jerarquía, como una estructura de árbol. Es útil para aplicaciones que necesitan manejar datos organizados jerárquicamente.
Por ejemplo, se puede utilizar **hierarchyid** para almacenar la estructura de una organización empresarial, incluyendo niveles jerárquicos y relaciones entre departamentos.
uniqueidentifier: Almacena identificadores únicos globales (**GUIDs**) que se generan automáticamente y garantizan unicidad en todo el mundo. Es útil para identificar registros de forma única.
Por ejemplo, se puede utilizar **uniqueidentifier** para asignar un identificador único a cada cliente en una base de datos.
sql_variant: Permite almacenar datos de diferentes tipos en una misma columna. Es útil cuando se necesita flexibilidad para almacenar valores de distintos tipos en un solo campo.
Por ejemplo, se puede utilizar **sql_variant** para almacenar valores de diferentes tipos de datos en una tabla que contiene información de configuraciones generales de una aplicación.
xml: Permite almacenar datos en formato XML, lo que facilita el manejo de documentos y datos estructurados. Este tipo de dato es útil para almacenar información jerárquica o de configuración.
Por ejemplo, se puede utilizar **xml** para almacenar la configuración de una aplicación en un formato estructurado.
Tipos de geometría espacial: Incluyen **geometry** y otros tipos que permiten almacenar datos espaciales bidimensionales, como puntos, líneas y polígonos. Son útiles para aplicaciones de sistemas de información geográfica (SIG).
Por ejemplo, se puede utilizar **geometry** para almacenar la ubicación y forma de un parque en una ciudad.
Tipos de geografía espacial: Incluyen **geography** y otros tipos que permiten almacenar datos espaciales geográficos en coordenadas geográficas (latitud y longitud). Son útiles para aplicaciones de geolocalización.
Por ejemplo, se puede utilizar **geography** para almacenar la ubicación de un lugar en un mapa en términos de latitud y longitud.
table: Representa una tabla temporaria que se utiliza dentro de una consulta o procedimiento almacenado. Este tipo de dato es útil para manejar conjuntos de datos temporales y realizar operaciones sobre ellos.
Por ejemplo, se puede utilizar table para almacenar resultados intermedios de una consulta compleja antes de combinarlos con otros datos.

En general, estos tipos de datos especializados se deben utilizar según las necesidades específicas de la aplicación para garantizar un manejo eficiente y preciso de la información.

5. CONCLUSIONES

En resumen, SQL Server 2022 Express ofrece una amplia variedad de tipos de datos que permiten manejar eficientemente diferentes tipos de información en una base de datos. Los tipos de datos numéricos exactos, aproximados, de fecha y hora, de cadenas de caracteres, de cadenas de caracteres Unicode y de cadenas binarias, así como otros tipos de datos especializados, proporcionan la flexibilidad necesaria para abordar una variedad de casos de uso.

Es importante seleccionar el tipo de dato adecuado para cada situación, considerando las necesidades específicas de la aplicación, como la precisión, el rango de valores, la longitud variable, y la compatibilidad con caracteres Unicode o datos binarios. Además, los tipos de datos especializados, como **sql_variant**, **uniqueidentifier**, y los tipos espaciales, ofrecen soluciones para casos de uso avanzados.

6. RECOMENDACIONES

- Selección cuidadosa de tipos de datos: Elija el tipo de dato que mejor se adapte a los requerimientos de la aplicación en términos de precisión, rango, y eficiencia en el almacenamiento.

- Optimización con **varchar** y **nvarchar**: Prefiera **varchar** y **nvarchar** para manejar cadenas de longitud variable y optimizar el uso del espacio.

- Uso eficiente de tipos espaciales: Utilice **geometry** y **geography** cuando necesite almacenar información geoespacial con precisión.

- Manejo de datos binarios con **varbinary**: Para almacenar datos binarios, utilice **varbinary** para flexibilidad en el tamaño y compatibilidad.

- Control de concurrencia: Emplee **rowversion** para garantizar la consistencia de los datos en entornos de acceso concurrente.

- Actualización continua: Manténgase informado sobre las actualizaciones de SQL Server para aprovechar nuevas funcionalidades y mejoras en el manejo de datos.

7. BIBLIOGRAFIA

Tipos de datos en SQL Server. (2022, agosto 25). SQLearning | Tutoriales de SQL y Transact-SQL; SQLearning. https://sqlearning.com/es/introduccion-sql-server/tipos-datos/

MikeRayMSFT. (s/f). Tipos de datos (Transact-SQL). Microsoft.com. Recuperado el 13 de abril de 2024, de https://learn.microsoft.com/es-es/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver16