Sánchez Wilhelm Jeremias
Bases de Datos Grupo 1

Tarea 8.

Investigar tipos de datos en PostgreSQL: numéricos, caracteres, fechas, y dos que les llamen la atención.

Datos numéricos:

Name	Storage Size	Description	Range
`smallint`	2 bytes	small-range integer	-32768 to +32767
`integer`	4 bytes	typical choice for integer	-2147483648 to +2147483647
`bigint`	8 bytes	large-range integer	-9223372036854775808 to +9223372036854775807
`decimal`	variable	user-specified precision, exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
`numeric`	variable	user-specified precision, exact	up to 131072 digits before the decimal point; up to 16383 digits after the decimal point
`real`	4 bytes	variable-precision, inexact	6 decimal digits precision
`double precision`	8 bytes	variable-precision, inexact	15 decimal digits precision
`smallserial`	2 bytes	small autoincrementing integer	1 to 32767
`serial`	4 bytes	autoincrementing integer	1 to 2147483647
`bigserial`	8 bytes	large autoincrementing integer	1 to 9223372036854775807

Los tipos numéricos consisten en enteros de dos, cuatro y ocho bytes, números de punto flotante de cuatro y ocho bytes, y decimales de precisión seleccionable.

Los tipos de datos **`smallserial`**, **`serial`** y **`bigserial`** no son tipos reales, sino simplemente una conveniencia notacional para crear columnas de identificadores únicos (similar a la propiedad **`AUTO_INCREMENT`** admitida por algunas otras bases de datos).

Datos de caracteres:

Name	Description
`character varying(n), varchar(n)`	variable-length with limit
`character(n), char(n), bpchar(n)`	fixed-length, blank padded
`text`	variable unlimited length

`SQL` define dos tipos de caracteres principales: **`character varying(n)`** y **`character(n)`**, donde **`n`** es un número entero positivo. Ambos tipos pueden almacenar cadenas de hasta **`n`** caracteres (no bytes) de longitud. Intentar almacenar una cadena más larga en una columna de estos tipos resultará en un error, a menos que los caracteres excedentes sean todos espacios, en cuyo caso

la cadena se truncará a la longitud máxima. Si la cadena a almacenar es más corta que la longitud declarada, los valores del tipo **character** serán rellenados con espacios; los valores del tipo **character varying** simplemente almacenarán la cadena más corta.

Además, **PostgreSQL** proporciona el tipo de dato **text**, que almacena cadenas de cualquier longitud. Aunque el tipo **text** no está en el estándar **SQL**, varios otros sistemas de gestión de bases de datos **SQL** también lo tienen. **text** es el tipo de dato de cadena nativo de **PostgreSQL**, ya que la mayoría de las funciones integradas que operan en cadenas están declaradas para tomar o devolver **text** en lugar de **character varying**. Para muchos propósitos, **character varying** actúa como si fuera un dominio sobre **text**.

El nombre de tipo **varchar** es un alias de **character varying**, mientras que **char** y **bpchar** son alias de **character**. Los alias **varchar** y **char** están definidos en el estándar **SQL**, pero **bpchar** es una extensión de **PostgreSQL**.

Datos de fecha y hora:

Name	Storage Size	Description	Low Value	High Value	Resolution
`timestamp [ (p) ] [ without time zone ]`	8 bytes	both date and time (no time zone)	4713 BC	294276 AD	1 microsecond
`timestamp [ (p) ] with time zone`	8 bytes	both date and time, with time zone	4713 BC	294276 AD	1 microsecond
`date`	4 bytes	date (no time of day)	4713 BC	5874897 AD	1 day
`time [ (p) ] [ without time zone ]`	8 bytes	time of day (no date)	00:00:00	24:00:00	1 microsecond
`time [ (p) ] with time zone`	12 bytes	time of day (no date), with time zone	00:00:00+1559	24:00:00-1559	1 microsecond
`interval [ fields ] [ (p) ]`	16 bytes	time interval	-178000000 years	178000000 years	1 microsecond

PostgreSQL admite el conjunto completo de tipos de fecha y hora **SQL**. Las fechas se cuentan según el calendario gregoriano, incluso en años anteriores a la introducción de ese calendario. **time**, **timestamp** e **interval** aceptan un valor de precisión opcional **p** que especifica el número de dígitos fraccionarios retenidos en el campo de segundos. Por defecto, no hay un límite explícito en la precisión. El rango permitido para **p** va de 0 a 6. El tipo de dato **time with time zone** está definido por el estándar **SQL**, pero su definición exhibe propiedades que plantean dudas sobre su utilidad. En la mayoría de los casos, una combinación de **date**, **time**, **timestamp without time zone** y **timestamp with time zone** debería proporcionar un conjunto completo de funcionalidades de fecha/hora requeridas por cualquier aplicación.

Datos monetarios:

Name	Storage Size	Description	Range
money	8 bytes	currency amount	-92233720368547758.08 to +92233720368547758.07

El tipo de dato **money** almacena una cantidad de moneda con una precisión fraccional fija. La precisión fraccional se determina mediante la configuración **lc_monetary** de la base de datos. El rango mostrado en la tabla asume que hay dos dígitos fraccionales. Se acepta entrada en una variedad de formatos, incluyendo literales enteros y de punto flotante, así como el formato típico de moneda, como '**$1,000.00**'. La salida generalmente se presenta en la última forma mencionada, pero depende de la configuración regional.

Dado que la salida de este tipo de dato es sensible a la configuración regional, podría no funcionar cargar datos de dinero en una base de datos que tenga una configuración diferente de **lc_monetary**.

Los valores de los tipos de datos **numeric**, **int** y **bigint** pueden convertirse a **money**. La conversión desde los tipos de datos **real** y **double precision** se puede hacer mediante una conversión a **numeric** primero. Sin embargo, esto no se recomienda. Los números de punto flotante no deben utilizarse para manejar dinero debido al potencial de errores de redondeo. Un valor de dinero se puede convertir a **numeric** sin pérdida de precisión. La conversión a otros tipos podría potencialmente perder precisión.

Datos geométricos:

Name	Storage Size	Description	Representation
point	16 bytes	Point on a plane	(x,y)
line	32 bytes	Infinite line	{A,B,C}
lseg	32 bytes	Finite line segment	((x1,y1),(x2,y2))
box	32 bytes	Rectangular box	((x1,y1),(x2,y2))
path	16+16n bytes	Closed path (similar to polygon)	((x1,y1),...)
path	16+16n bytes	Open path	[(x1,y1),...]
polygon	40+16n bytes	Polygon (similar to closed path)	((x1,y1),...)
circle	24 bytes	Circle	<(x,y),r> (center point and radius)

Los tipos de datos geométricos representan objetos espaciales bidimensionales. Se dispone de un conjunto completo de funciones y operadores para realizar diversas operaciones geométricas, como escalamiento, traslación, rotación y determinación de intersecciones.