Clase Lenguaje SQL

Concepto

Conocido como SQL por sus siglas en inglés (Structured Query Language), es un lenguaje de consultas estructuradas.

IBM desarrolló la versión original de SQL, originalmente denominado Sequel, como parte del proyecto System R a principios de 1970. El lenguaje Sequel ha evolucionado desde entonces y su nombre ha pasado a ser SQL (Structured Query Language, lenguaje estructurado de consultas). Hoy en día, numerosos productos son compatibles con el lenguaje SQL y se ha establecido como el lenguaje estándar para las bases de datos relacionales.

Se usa para describir conjuntos de datos que pueden ayudar a responder preguntas. La sintaxis SQL se basa en el idioma inglés y usa muchos de los mismos elementos que la sintaxis del lenguaje Visual Basic para desarrollo de aplicaciones.

Este lenguaje no solo sirve para obtener información de los datos almacenados, sino también para conocer la metadata de la base de datos. La metadata permite conocer el origen y la estructura de cada uno de los objetos que componen la base de datos, desde una columna en una tabla, hasta el contenido de un store procedure.

El lenguaje se implementa por medio de un sistema de gestión de base de datos. De estos existen muchas tecnologias en la actualidad, como por ejemplo SQL, Teradata, Oracle, etc.

Sintaxis

La sintaxis en el lenguaje de SQL se define como el conjunto de reglas que deben seguirse al escribir el código de consultas estructuradas para considerarse como correctas y así completar la ejecución exitosamente.

Estas reglas permiten que se use de forma adecuada cada sentencia en las instrucciones sql, de esta forma si una cláusula está mal escrita, o mal ubicada, el sistema de gestión de base de datos arroja un error de sintaxis, el cual alerta de corregir lo necesario para obtener el resultado esperado.

Gino Luciano Rojo

Sentencias

También denominadas comandos o cláusulas, son las palabras reservadas que componen el lenguaje para ejecutar acciones sobre la base de datos. Gracias a estas los usuarios pueden operar en las bases de datos, y se caracterizan porque al usarlas en la redacción, la fuente adquiere colores distintos al estándar.

Clausula Select y From

Una cláusula SELECT se usa para especificar los nombres de los campos que contienen los datos que quiere usar en una consulta. También puede usar operaciones matemáticas, u operaciones con funciones especiales, en lugar de o además de los campos. Incluso puede usar otra instrucción SELECT como campo, esto se conoce como una subconsulta.

Supongamos que quiere saber el número de teléfono de sus clientes. Suponiendo que el campo que almacena los números de teléfono de los clientes se denomina txtCustPhone, la cláusula SELECT es la siguiente:

SELECT txtCustPhone

La cláusula FROM permite especificar las tablas que contienen los campos que se usarán en la cláusula SELECT.

SELECT txtCustPhone FROM Customers

Si la instrucción SQL tiene dos o más campos con el mismo nombre, debe agregar el nombre del origen de datos de cada campo al nombre del campo en la cláusula SELECT. Use el mismo nombre del origen de datos que se usa en la cláusula FROM o asigne un alias al mismo.

Si quiere incluir todos los campos de un origen de datos, puede enumerarlos todos de forma individual en la cláusula SELECT, o bien puede usar el carácter comodín de asterisco (*). Cuando use el asterisco, el SGBD determina al ejecutar la consulta qué campos contiene el origen de datos e incluye todos esos campos en la consulta. Esto ayuda a asegurarse de que la consulta se mantiene actualizada si se agregan nuevos campos al origen de datos.

Puede usar el asterisco con uno o varios orígenes de datos en una instrucción SQL. Si usa el asterisco y hay varios orígenes de datos, debe incluir el nombre del origen de datos con el asterisco, para que Access pueda determinar desde qué origen de datos se incluyen los campos.

Por ejemplo, supongamos que quiere seleccionar todos los campos de la tabla Pedidos, pero solo la dirección de correo de la tabla Contactos. La cláusula SELECT podría ser similar a esta:

SELECT Pedidos.*, Contactos.Email_Address From…

Seleccionar valores distintos

Si sabe que la instrucción va a seleccionar datos redundantes y en su lugar preferiría ver solo valores diferentes, puede usar la palabra clave DISTINCT en la cláusula SELECT. Por ejemplo, supongamos que los clientes representan distintos intereses, algunos de los cuales usan el mismo número de teléfono. Si quiere asegurarse de que cada número de teléfono solo se vea una vez, la cláusula SELECT sería la siguiente:

SELECT DISTINCT txtCustomerPhone

Clausula AS

Permite usar nombres de sustitutos para campos o expresiones: la palabra clave AS.

Se puede cambiar la etiqueta que se muestra para un campo en la vista de resultados mediante la palabra clave AS y un alias de campo en la cláusula SELECT. Un alias de campo es un nombre que se asigna a un campo de una consulta para facilitar la lectura de los resultados. Por ejemplo, si quiere seleccionar los datos de un campo denominado txtCustPhone y el campo contiene números de teléfono de clientes, puede mejorar la legibilidad de los resultados mediante un alias de campo en la instrucción SELECT.

Es importante tener en cuenta que este alias no afecta el nombre original del campo en la tabla de origen.

Por ejemplo:
SELECT txtCustPhone AS Telefono

Clausula WHERE

En una instrucción SQL, la cláusula WHERE especifica criterios que tienen que cumplir los valores de campo para que los registros que contienen los valores se incluyan en los resultados de la consulta.

Se pueden usar criterios de consulta en la cláusula WHERE de una instrucción SELECT.

Una cláusula WHERE tiene la siguiente sintaxis básica:

WHERE campo = criterio

Por ejemplo, para obtener el número de teléfono de un cliente, pero solo se tiene el apellido del cliente es Valladares. En lugar de buscar en todos los números de teléfono de la base de datos, se puede usar una cláusula WHERE para limitar los resultados y que resulte más sencillo encontrar el número de teléfono. Si se da por hecho que los apellidos se almacenan en un campo denominado Apellidos, la cláusula WHERE sería similar a lo siguiente:

Gino Luciano Rojo

Select NumTelefono From Clientes
WHERE Apellido = ‘Valladares’

Clausula HAVING

En una instrucción SQL, la cláusula HAVING especifica criterios que tienen que cumplir los valores de un campo nuevo generado en la consulta, este campo debe estar compuesto por alguna operación especial de agrupación como por ejemplo SUM, MAX, AVG, etc.

De esta forma los registros que apliquen al criterio o condición podrán ser filtrados y obtenidos como resultado.

Se pueden usar criterios de consulta en la cláusula HAVING de una instrucción SELECT, tales como los usados en la consulta WHERE.

Una cláusula HAVING tiene la siguiente sintaxis básica:

Por ejemplo, la sintaxis de la consulta para obtener los nombres de los clientes que tuvieron compras superiores a 3000, en el total de la tabla:

Select Nombre, Sum(Compra) as TOTAL From Compras
HAVING Sum(Compras) > 3000

Clausula ORDER BY

Esta cláusula permite, ordenar el conjunto de resultados de una consulta por la lista de columnas especificada y, opcionalmente.
El orden en que se devuelven las filas en un conjunto de resultados no se puede garantizar, a menos que se especifique una cláusula ORDER BY.

Una cláusula ORDER BY tiene la siguiente sintaxis básica:

Por ejemplo, la sintaxis de la consulta para obtener todos los nombres y apellidos de los clientes, ordenado en orden alfabética de A hasta Z:

Select Nombre, Apellido From Clientes
ORDER BY Nombre ASC

Gino Luciano Rojo