Fundamentos de Bases de Datos.
Práctica 7.

Profesor: M.I. Gerardo Avilés Rosas
gar@ciencias.unam.mx
Laboratorio: Luis Eduardo Castro Omaña
lalo_ castro@ciencias.unam.mx

14 de octubre de 2019

Se dan a conocer especificaciones de entrega para la practica 7.

1. DDL

Un esquema de base de datos se especifica mediante un conjunto de definiciones expresadas mediante un lenguaje especial llamado lenguaje de definición de datos (DDL)[1].

Un diccionario de datos contiene metadatos, es decir, datos acerca de los datos. El esquema de una tabla es un ejemplo de metadatos. Un sistema de base de datos consulta el diccionario de datos antes de leer o modificar los datos reales.

Especificamos el almacenamiento y los métodos de acceso usados por el sistema de bases de datos por un conjunto de instrucciones en un tipo especial de DDL denominado lenguaje de almacenamiento y definición de datos. Estas instrucciones definen los detalles de implementación de los esquemas de base de datos, que se ocultan usualmente a los usuarios.

2. sqlcmd

La utilidad sqlcmd es una herramienta de línea de comandos para la ejecución ad hoc e interactiva de instrucciones y scripts de Transact-SQL y para la automatización de tareas de creación de scripts de Transact-SQL. Es posible utilizar la herramienta de las siguientes formas:

[1] Siglas en inglés de Data Definition Language

- Los usuarios escriben instrucciones Transact-SQL interactivamente de una forma similar al modo en que trabajan con el símbolo del sistema. Los resultados se muestran en el símbolo del sistema.
- Los usuarios envían un trabajo sqlcmd especificando la ejecución de una instrucción Transact-SQL individual o dirigiendo la utilidad hacia un archivo de texto que contiene las instrucciones Transact-SQL que se van a ejecutar. El resultado se dirige normalmente hacia un archivo de texto, aunque también se puede mostrar en el símbolo del sistema.

Algunos de los comandos mas utilizados en sqlcmd son:

- La opción del servidor (-S) que identifica la instancia de Microsoft SQL Server a la que se conecta sqlcmd.

  ```
  sqlcmd -S <nombre-servidor>\<nombre-instancia>
  ```

- Las opciones de autenticación (-E, -U y -P) que especifican las credenciales que usa sqlcmd para conectarse a la instancia de SQL Server. Conectarse a la instancia con nombre mediante la Autenticación de Windows para ejecutar de forma interactiva instrucciones Transact-SQL:

  ```
  sqlcmd -U <nombreUsuario> -S <nombre-servidor>\<nombre-instancia>
  ```

- Las opciones de entrada (-Q, -q e -i) que identifican la ubicación de la entrada a sqlcmd. Conectarse a una instancia con nombre mediante la Autenticación de Windows y especificar los archivos de entrada:

  ```
  sqlcmd -S <nombre-servidor>\<nombre-instancia> -i <script.sql>
  ```

  Conectarse a la instancia predeterminada del equipo local mediante la Autenticación de Windows, ejecutar una consulta y mantener la ejecución de sqlcmd después de la finalización de la consulta:

  ```
  sqlcmd -q "<Instruccion Transact-SQL>"
  ```

- La opción de salida (-o) que especifica el archivo en el que se guardará la salida de sqlcmd.

  ```
  sqlcmd -S <nombre-servidor>\<nombre-instancia> -i <script.sql> -o <MyOutput.rpt>
  ```

- Para ver una lista de opciones admitidas por la utilidad sqlcmd, ejecuta:

  ```
  sqlcmd -?
  ```

Para mas información sobre sqlcmd puedes revisar la documentación en: `https://msdn.microsoft.com/es-es/library/ms162773.aspx`

3. Tablas

Una tabla consiste de renglones y columnas. Las columnas definen los datos que queremos almacenar y las filas contienen los valores para esas columnas.

Existen los siguientes tipos de tablas en SQL Server:

- Heap Table: Una tabla permanene, utilizada para almacenar renglones de datos.
- Clustered Table: Una tabla permanente con un indice clusterizado.
- Local Temporary Table: Comienzan con #. Tablas que existen en un bloque específico de código. Son tablas temporales creadas dentro de tempdb y se eliminan cuando el código se ha ejecutado.
- Global Temporary Table: Comienzan con ##. Tablas disponibles para todos los usuarios hasta que son destruidas. Se deben destruir manualmente, a menos que se reinicie el sistema(al reiniciar el SQL Server, se reconstruye tempdb, por lo que cualquier tabla temporal será destruida).

3.1. Crear Tablas

Crea una nueva tabla en SQL Server

```
CREATE TABLE
    [ database_name . [ schema_name ] . | schema_name . ] table_name
    ( { <column_definition> } [ ,...n ] )
[ ; ]
```

Database_name: Es el nombre de la base de datos en la que se crea la tabla. database_name debe especificar el nombre de una base de datos existente. Si no se especifica, database_name el valor predeterminado es la base de datos actual. El inicio de sesión para la conexión actual debe estar asociado con un identificador de usuario existente en la base de datos especificada por database_name, y el Id. de usuario debe tener permisos CREATE TABLE.

schema_name: Es el nombre del esquema al que pertenece la nueva tabla. Un esquema es un contenedor que permite agrupar objetos, lo cual permite asignar permisos de seguridad a un conjunto entero de objetos, en lugar de asignarlos individualmente. Si no se específica un esquema, los objetos creados se agregan a el esquema dbo.

table_name: Es el nombre de la nueva tabla. Los nombres de tabla deben seguir las reglas para identificadores. table_name puede tener un máximo de 128 caracteres, excepto para los nombres de tablas temporales locales (nombres precedidos de un solo signo de número (#)) que no puede superar los 116 caracteres.

3.2. Editar Tablas

Modifica una definición de tabla al alterar, agregar o quitar columnas y restricciones, reasignar y regenerar particiones, o deshabilitar o habilitar restricciones y desencadenadores.

```
ALTER TABLE [ database_name . [ schema_name ] . | schema_name . ] table_name
{
    ALTER COLUMN column_name
    {
        [ type_schema_name. ] type_name
            [ (
                {
                   precision [ , scale ]
                 | max
                 | xml_schema_collection
                }
            ) ]
        [ NULL | NOT NULL ]
      }
```

3.3. Tipos de datos

En SQL Server, cada columna, variable local, expresión y parámetro tiene un tipo de datos relacionado. Un tipo de datos es un atributo que especifica el tipo de datos que el objeto puede contener: datos de enteros, datos de caracteres, datos de moneda, datos de fecha y hora, cadenas binarias, etc.

Tipos de datos de SQL Server se organizan en las siguientes categorías:

- Numéricos exactos:
  - Enteros: Tipos de datos numéricos exactos que utilizan datos enteros.

Tipo de datos	Intervalo	Storage
bigint	De $-2^63(-9{,}223{,}372{,}036{,}854{,}775{,}808)$ a $2^63-1(9{,}223{,}372{,}036{,}854{,}775{,}807)$	8 bytes
int	De $-2^31(-2{,}147{,}483{,}648)$ a $2^31-1(2{,}147{,}483{,}647)$	4 bytes
smallint	De $-2^15(-32{,}768)$ a $2^15-1(32{,}767)$	2 bytes
tinyint	De 0 a 255	1 byte

  - Decimal y númericos: Tipos de datos numéricos que tienen precisión y escala fijas. Decimal y numeric son sinónimos y se pueden usar indistintamente.

decimal[(p[,s])] y numeric[(p[,s])] Números de precisión y escala fijas. Cuando se utiliza la precisión máxima, los valores válidos se sitúan entre -10^38+1 y 10^38-1. numérico es funcionalmente equivalente a decimal.

p (precisión): El número total máximo de dígitos decimales que almacenará, tanto a la izquierda como a la derecha del separador decimal. La precisión debe ser un valor comprendido entre 1 y la precisión máxima de 38. La precisión predeterminada es 18.

s (escala): El número de dígitos decimales que se almacenará a la derecha del separador decimal. Este número se resta de p para determinar el número máximo de dígitos a la izquierda del separador decimal. El número máximo de dígitos decimales que se puede almacenar a la derecha del separador decimal. Escala debe ser un valor comprendido entre 0 y p. Solo es posible especificar la escala si se ha especificado la precisión. La escala predeterminada es 0.

- Caracteres: Son tipos de datos de cadena de longitud fija o de longitud variable.

  char [(n)] Datos de cadena no Unicode de longitud fija. n define la longitud de cadena y debe ser un valor entre 1 y 8.000. El tamaño de almacenamiento es n bytes.

  varchar [(n | max)] Datos de cadena no Unicode de longitud variable. n define la longitud de cadena y puede ser un valor entre 1 y 8.000. max indica que el tamaño máximo de almacenamiento es $2^{31}-1$ bytes (2 GB). El tamaño de almacenamiento es la longitud real de los datos especificados + 2 bytes.

- Fechas: Tipo de datos que almacenan datos de fechas.
  time: Define un tiempo de un día. La hora es sin conocimiento de zona horaria y se basa en un reloj de 24 horas.
  date: Define una fecha con el formato YYYY-MM-DD

4. Integridad de datos

4.1. Restricciones

Las restricciones le permiten definir la manera en que Motor de base de datos exigirá automáticamente la integridad de una base de datos.Las restricciones definen reglas relativas a los valores permitidos en las columnas y constituyen el mecanismo estándar para exigir la integridad. El uso de restricciones es pre-

ferible al uso de Desencadenadores DML[2], reglas y valores predeterminados.El optimizador de consultas también utiliza definiciones de restricciones para generar planes de ejecución de consultas de alto rendimiento.

SQL Server admite las siguientes clases de restricciones:

- NOT NULL especifica que la columna no acepta valores NULL. Un valor NULL no es lo mismo que cero (0), en blanco o que una cadena de caracteres de longitud cero, como . NULL significa que no hay ninguna entrada. La presencia de un valor NULL suele implicar que el valor es desconocido o no está definido. Se recomienda evitar la aceptación de valores NULL, dado que pueden implicar una mayor complejidad en las consultas y actualizaciones. Por otra parte, hay otras opciones para las columnas, como las restricciones PRIMARY KEY, que no pueden utilizarse con columnas que aceptan valores NULL.

- Las restricciones CHECK exigen la integridad del dominio mediante la limitación de los valores que se pueden asignar a una columna. Una restricción CHECK especifica una condición de búsqueda booleana (se evalúa como TRUE, FALSE o desconocido) que se aplica a todos los valores que se indican en la columna. Se rechazan todos los valores que se evalúan como FALSE. En una misma columna se pueden especificar varias restricciones CHECK.

- Las restricciones UNIQUE exigen la unicidad de los valores de un conjunto de columnas. En una restricción UNIQUE, dos filas de la tabla no pueden tener el mismo valor en las columnas. Las claves principales también exigen exclusividad, pero no aceptan NULL como uno de los valores exclusivos.

- Las restricciones PRIMARY KEY identifican la columna o el conjunto de columnas cuyos valores identifican de forma exclusiva cada una de las filas de una tabla. Dos filas de la tabla no pueden tener el mismo valor de clave principal. No se pueden asignar valores NULL a ninguna de las columnas de una clave principal. Se recomienda utilizar una columna pequeña de tipo entero como clave principal. Todas las tablas tienen que tener una clave principal. Una columna o combinación de columnas certificada como valor de clave principal se denomina clave candidata.

- Las restricciones FOREIGN KEY identifican y exigen las relaciones entre las tablas. Una clave externa de una tabla apunta a una clave candidata de otra tabla. No se puede insertar una fila que tenga un valor de clave externa, excepto NULL, si no hay una clave candidata con dicho valor. La cláusula ON DELETE controla las acciones que se llevarán a cabo si intenta eliminar una fila a la que apuntan las claves externas existentes. La cláusula ON DELETE tiene las siguientes opciones:

    - NO ACTION especifica que la eliminación produce un error.

[2]Se tratará con mas detalle en la siguiente práctica.

- CASCADE especifica que también se eliminan todas las filas con claves externas que apuntan a la fila eliminada.
- SET NULL especifica que todas las filas con claves externas que apuntan a la fila eliminada se establecen en NULL.
- SET DEFAULT especifica que todas las filas con claves externas que apuntan a la fila eliminada se establecen en sus valores predeterminados.

La cláusula ON UPDATE define las acciones que se llevarán a cabo si intenta actualizar un valor de clave candidata a la que apuntan las claves externas existentes. Esta cláusula también admite las opciones NO ACTION, CASCADE, SET NULL y SET DEFAULT.

4.2. Incrementadores

4.2.1. IDENTITY

Cada tabla debe contener una columna como llave primaría. Es ampliamente recomendable utilizar una columna que se incremente por cada registro en la tabla, dicha columna podría ser la llave primaria o parte de ella de la tabla que la contenga. En Transact-SQL se crea una columna de identidad en una tabla; la cual, genera un campo que se incrementara en i cada vez que se realice un registro en la tabla que la contiene. Esta propiedad se usa con las instrucciones CREATE TABLE y ALTER TABLE de Transact-SQL, se declara como sigue:

```
IDENTITY [ (valor_inicio , incremento) ]
```

- valor_inicio: Es el valor que se utiliza para la primera fila cargada en la tabla.
- incremento: Se trata del valor incremental que se agrega al valor de identidad de la anterior fila cargada.

Debe especificar tanto el valor de inicialización como el incremento, o bien ninguno de los dos. Si no se especifica ninguno, el valor predeterminado es (1,1).

La propiedad de identidad de una columna no garantiza lo siguiente:

- Unicidad del valor - exclusividad debe exigirse a través de un PRIMARY KEY o UNIQUE restricción o UNIQUE índice.
- Valores consecutivos dentro de una transacción : no se garantiza que una transacción insertar varias filas obtenga valores consecutivos para las filas, ya que podría producirse otras inserciones simultáneas en la tabla. Si los valores deben ser consecutivos, a continuación, la transacción debe usar un bloqueo exclusivo en la tabla o usar el SERIALIZABLE nivel de aislamiento.

- Valores consecutivos después de reiniciar el servidor u otros errores –SQL Server podría guardar en caché los valores de identidad por motivos de rendimiento y algunos de los valores asignados podrían perderse durante un reinicio del servidor o error de base de datos. Esto puede tener como resultado espacios en el valor de identidad al insertarlo. Si no es aceptable que haya espacios, la aplicación debe usar mecanismos propios para generar valores de clave. Mediante un generador de secuencias con el NO-CACHE opción puede limitar los espacios a transacciones que nunca se hayan confirmado.

- Reutilización de valores : para una propiedad de identidad especificada con el valor de inicialización e incremento específico, la identidad que el motor no reutiliza los valores. Si una instrucción de inserción concreta produce un error o si la instrucción de inserción se revierte, los valores de identidad utilizados se pierden y no volverán a generarse. Esto puede tener como resultado espacios cuando se generan los valores de identidad siguientes.

4.2.2. Secuencias

Una secuencia es un objeto enlazado a un esquema definido por el usuario que genera una secuencia de valores numéricos según la especificación con la que se creó la secuencia. La secuencia de valores numéricos se genera en orden ascendente o descendente en un intervalo definido y se puede configurar para reiniciarse (en un ciclo) cuando se agota. Las secuencias, a diferencia de las columnas de identidad, no se asocian a tablas concretas. Las aplicaciones hacen referencia a un objeto de secuencia para recuperar su valor siguiente. La aplicación controla la relación entre las secuencias y tablas. Las aplicaciones de usuario pueden hacer referencia un objeto de secuencia y coordinar los valores a través de varias filas y tablas.

A diferencia de los valores de las columnas de identidad que se generan cuando se insertan filas, una aplicación puede obtener el número de secuencia siguiente sin insertar la fila mediante una llamada a la función NEXT VALUE FOR.

```
CREATE SEQUENCE [nombre_esquema . ] nombre_secuencia
    [ AS [ tipo_dato ] ]
    [ START WITH <i> ]
    [ INCREMENT BY <i> ]
    [ { MINVALUE [ <i> ] } | { NO MINVALUE } ]
    [ { MAXVALUE [ <i> ] } | { NO MAXVALUE } ]
    [ CYCLE | { NO CYCLE } ]
    [ { CACHE [ <i> ] } | { NO CACHE } ]
    [ ; ]
```

Los números de secuencia se generan fuera del ámbito de la transacción

actual. Se utilizan tanto si la transacción que usa el número de secuencia se confirma como si se revierte.

5. Actividad

Deberán crear las instrucciones DDL para crear el esquema de bases de datos utilizando como base el diagrama relacional que realizaron en la práctica pasada.

Deben especificar las llaves foráneas, compuestas y primarias de cada tabla; así como las restricciones con las que puede contar la tabla.

6. Entregables

Se debe crear un script DDL.sql el cual contendrá las instrucciones para la creación de la base de datos, sus tablas y sus restricciones.

Como todo se debe hacer por linea de comando, el archivo README.txt debe tener una explicación muy detallada de como debo correr su script, si al seguir las instrucciones no es posible crear la base de datos no se calificará la práctica.

La entrega deberá ser el día lunes 21 de Octubre de 2019.