UTN
Facultad Regional Córdoba

TUP

Tecnicatura Universitaria en Programación

LABORATORIO DE COMPUTACIÓN I

Unidad Temática N°1:

Introducción al SQL. Sentencias de Definición de Datos

Material Teórico

1° Año – 1° Cuatrimestre

Índice

¿QUÉ ES SQL? 2
Usos de SQL 3
Beneficios 3
¿QUÉ ES SQL SERVER? 4
Arquitectura cliente-servidor 4
Aplicaciones cliente 5
Modelos de almacenamiento de datos 5
Objetos de SQL Server 6
TRANSACT-SQL 7
Tipos de Sentencias 7
SENTENCIAS DE DEFINICIÓN DE DATOS – DDL 7
Creación de una tabla 8
Tipos de datos 9
Modificación de una tabla 13
Eliminación de una tabla 14
BIBLIOGRAFÍA 15

¿QUÉ ES SQL?

Imagen 1: Extraída de Banco de Imágenes Pixabay

SQL, **Lenguaje de Consulta Estructurado** (Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas. Unifica características del álgebra y el cálculo relacional permitiendo lanzar consultas con el fin de recuperar información de interés de una base de datos, de una forma sencilla y en un leguaje similar al inglés utilizado para la comunicación entre personas.

Es un lenguaje estándar utilizado para comunicarse con bases de datos **relacionales**, definido y publicado por el ISO (*International Organization for Standardization – Organización Internacional de Estandarización*) y el ANSI (*American National Standards Institute – Instituto de Estandar Nacional Americano*).

El hecho de que sea estándar no quiere decir que sea idéntico para cada base de datos.

El Lenguaje Estructurado de Consulta (SQL) es un lenguaje de base de datos normalizado, utilizado por los diferentes motores de bases de datos para realizar determinadas operaciones sobre los datos, o sobre la estructura de los mismos. Pero como sucede con cualquier sistema de normalización hay excepciones para casi todo. De hecho, cada motor de bases de datos tiene sus peculiaridades y lo hace diferente de otro motor. Por lo tanto, el lenguaje SQL normalizado (ANSI) no nos servirá para resolver todos los problemas, aunque si se puede asegurar que cualquier sentencia escrita en ANSI será interpretable por cualquier motor de datos.

Imagen 2: Extraída de Banco de Imágenes Pixabay

Usos de SQL

Además de ser una herramienta de consulta, SQL se utiliza para controlar las funciones que un SGBD proporciona, incluyendo:

- **Definición de Datos:** Permite que el usuario defina la estructura y la organización de los datos almacenados, así como las relaciones existentes entre ellos.
- **Recuperación de Datos:** Permite a un usuario o a un programa recuperar y utilizar los datos almacenados en una base de datos
- **Manipulación de Datos:** Permite a un usuario o a un programa actualizar la base de datos añadiendo datos nuevos, borrando los viejos y modificando los almacenados previamente.
- **Control de Acceso:** Se utiliza para restringir la capacidad de un usuario para recuperar, añadir y modificar datos, protegiendo los datos almacenados contra accesos no autorizados
- **Compartición de Información:** Para coordinar la compartición de datos entre usuarios concurrentes, asegurándose que no haya interferencias entre ellos.
- **Integridad de datos:** Define restricciones de integridad en la base de datos, protegiéndola de alteraciones indebidas a actualizaciones inconsistentes o fallos de sistema

Beneficios

- **Independencia de los proveedores:** Esfuerzo mínimo de conversión entre un DBMS y otro.
- **Portabilidad entre sistemas informáticos:** Las aplicaciones SQL que comienzan en sistemas monousuarios, pueden ser transferidos a sistemas mayores según van creciendo.
- **Fundamento Relacional:** SQL es un lenguaje de base de datos relacionales, y ha llegado a ser popular junto con el modelo de base de datos.
- **Estructura de alto nivel parecida al inglés:** las sentencias son frases sencillas en inglés haciendo que SQL sea fácil de entender y aprender.
- **Consultas interactivas ad hoc** (en línea, al instante): SQL es un lenguaje de consultas interactivas que proporciona a los usuarios acceso temporario a los datos almacenados.
- **Múltiples vistas de datos:** se puede dar diferentes vistas de la estructura y contenidos de una base de datos a diferentes usuarios.

- **Definición dinámica de datos:** con SQL la estructura de una base de datos puede ser modificada y ampliada dinámicamente, incluso mientras los usuarios están accediendo a los contenidos de la base de datos.
- **Arquitectura cliente/servidor:** los programas de las computadoras utilizan SQL para comunicarse, a través de una red de área local, con los servidores de base de datos que almacenan los datos compartidos.

¿QUÉ ES SQL SERVER?

SQL Server es un Sistema de Gestión de Bases de Datos Relacionales (**SGBDR** o RDBMS Relational Data Base Management System), diseñado para trabajar con grandes cantidades de información. Ofrece el soporte de información para aplicaciones Cliente/Servidor, las cuales mediante una interfaz y una red LAN los clientes acceden a los datos.

Arquitectura cliente-servidor

Imagen 3: Extraída de Banco de Imágenes Pixabay

SQL Server utiliza la arquitectura cliente-servidor para separar la carga de trabajo en tareas que se ejecutan en equipos servidores y tareas que se ejecutan en equipos cliente:

- El cliente es el responsable de la lógica empresarial y de presentar los datos al usuario. Normalmente, el cliente se ejecuta en uno o varios equipos, pero también puede ejecutarse en el equipo servidor junto con SQL Server.
- SQL Server administra las bases de datos y asigna los recursos disponibles del servidor (como la memoria, el ancho de banda de la red y las operaciones de disco) entre las distintas solicitudes.

La arquitectura cliente-servidor permite diseñar y distribuir aplicaciones para mejorar una gran variedad de entornos. Las interfaces de programación de los clientes proporcionan los medios para que las aplicaciones se ejecuten en equipos clientes independientes y se comuniquen con el servidor a través de la red.

Como es un entorno Cliente/Servidor, los datos se guardan en forma centralizada en un computador denominado SERVIDOR, siendo éste el responsable de:

- Mantener las relaciones entre los datos de una base de datos
- Asegurar que los datos estén correctamente almacenados y que no se infrinjan las reglas que definen las relaciones entre los datos

- Recuperar todos los datos hasta un punto de coherencia garantizada, en caso de que se produzca un error del sistema

Del lado del cliente se accede al servidor a través de una aplicación, la cual está basada en el estándar SQL, utilizando un conjunto de sentencias y comandos que permiten especificar la información que se desea recuperar o modificar.

Aplicaciones cliente

Los usuarios no tienen acceso a SQL Server directamente, sino que utilizan distintas aplicaciones cliente escritas para tener acceso a los datos. Estas aplicaciones tienen acceso a SQL Server mediante:

SQL Server consta de componentes cliente y servidor que almacenan y recuperan datos. SQL Server utiliza una arquitectura de comunicación por capas para aislar las aplicaciones de la red y los protocolos subyacentes.

Esta arquitectura permite distribuir la misma aplicación en diferentes entornos de red.

Modelos de almacenamiento de datos

SQL Server administra bases de datos OLTP y OLAP

- **Bases de datos OLTP** (On-Line Transactional Processing: procesamiento transaccional en linea) Son bases de datos orientadas al procesamiento de transacciones que involucra acciones de inserción, eliminación y actualización de datos. SQL Server permite que un gran número de usuarios realice transacciones y modifique simultáneamente datos en tiempo real en bases de datos OLTP. Algunos ejemplos de bases de datos OLTP serían sistemas de transacciones bancarias y de billetes de avión.

Imagen 4: Extraída de Banco de Imágenes Pixabay

- **Bases de datos OLAP** (On-Line Analytical Processing: procesamiento analítico en línea) En general estas bases de datos involucran acciones de consulta La tecnología OLAP organiza y resume grandes cantidades de datos, de manera que un analista pueda evaluar los datos rápidamente en tiempo real.

Imagen 5: Extraída de Banco de Imágenes Pixabay

Los datos son organizados de otra manera para permitir la realización de informes, análisis corporativos, estadísticas que ayudan a la toma de decisiones.

Objetos de SQL Server

Una base de datos es una colección de datos, tablas y otros objetos. Los objetos de base de datos le ayudan a estructurar los datos y a definir los mecanismos que mantienen la integridad de los datos. En la tabla siguiente se describen los objetos de base de datos de SQL Server.

Objeto de base de datos	Descripción
Tabla	Define un conjunto de filas que tienen columnas asociadas. Tipo de datos: Define los valores permitidos para los datos de una columna o variable.
SQL Server proporciona tipos de datos del sistema.	Los usuarios crean tipos de datos definidos por el usuario.
Restricción	Define reglas relativas a los valores permitidos en las columnas y es el mecanismo estándar para exigir la integridad de los datos.
Valor predeterminado	Define un valor que se almacena en una columna si no se especifica explícitamente ningún otro valor.
Regla	Contiene información que define los valores válidos que se pueden almacenar en una columna o en un tipo de datos.
Índice	Es una estructura de almacenamiento que proporciona acceso rápido para la recuperación de los datos y puede exigir la integridad de los datos.
Vista	Proporciona una forma de ver los datos de una o varias tablas o vistas de una base de datos.
Función	Definida por el usuario puede devolver un valor escalar o una tabla. Las funciones se utilizan para encapsular la lógica que se realiza con frecuencia.
Procedimiento almacenado	Es una colección con nombre de instrucciones Transact-SQL compiladas previamente que se ejecutan juntas.

Tabla 1: Elaboración propia.

TRANSACT-SQL

SQL Server tiene incorporado un lenguaje denominado Transact-SQL que contiene los comandos y sentencias necesarios para administrar una instancia de SQL Server, crear y administrar sus objetos, y consultar, insertar, modificar, y eliminar datos de las tablas.

Imagen 6: Extraída de Banco de Imágenes Pixabay

Tipos de Sentencias

Las distintas sentencias que se pueden ejecutar son:

- **DE DEFINICIÓN DE DATOS** (DDL Data Definition Language): se utilizan para crear, modificar y eliminar objetos de una base de datos. Algunas son: CREATE TABLE, ALTER TABLE, DROP TABLE, CREATE INDEX etc.

Imagen 7: Extraída de Banco de Imágenes Pixabay

- **DE MANIPULACIÓN DE DATOS** (DML Data Manipulation Language): se utilizan para recuperar, agregar, modificar o eliminar datos de una base de datos. Por ejemplo: SELECT, INSERT, UPDATE, DELETE.
- **DE CONTROL DE ACCESO** (DCL Data Control Language): conceden o suprimen privilegios a usuarios. Por ejemplo: GRANT y REVOKE.

Imagen 8: Extraída de Banco de Imágenes Pixabay

- **DE CONTROL DE TRANSACCIONES** (TCL Transaction Contro Languaje)**:** finalizan o abortan la transacción actual: COMMIT, ROLLBACK.
- **DE PROGRAMACIÓN:** DECLARE, OPEN, EXECUTE, DESCRIBE, etc.

SENTENCIAS DE DEFINICIÓN DE DATOS – DDL

Como ya se vio anteriormente, este tipo de sentencias son utilizadas para crear y manejar objetos y propiedades de una base de datos, tales como tablas, vistas, procedimientos, etc. Las sentencias son:

Para crear un objeto:

```
CREATE nombre_objeto
```

Para modificar un objeto:

```
ALTER nombre_objeto
```

Para eliminar un objeto:

```
DROP nombre_objeto
```

Creación de una tabla

Sintaxis de una sentencia básica de creación de tabla:

```
CREATE TABLE nombre_tabla
(nom_col1 tipo_dato [IDENTITY(inicio,incremento)][NULL|NOT NULL],
 nom_col2  tipo_dato,
 nom_col_n tipo_dato
 [CONSTRAINT nombre_constraint PRIMARY KEY(nom_col1)],
 [CONSTRAINT nombre_constraint FOREIGN KEY (nom_col2)
             REFERENCES tabla2 (nombre_columna_tabla2)]
)
```

En donde:

- **CREATE TABLE** *nombre_tabla*: se debe especificar el nombre de la tabla a crear.
- ***Nom_col1 tipo_dato***: se describen los nombres de columnas (nombre de campo) con su tipo de dato, cuando sean más de dos campos se separa con comas.
- **IDENTITY(***inicio***,** *incremento***)**: es una propiedad que se utiliza para definir a una columna como columna identidad. Columna cuyo contenido se genera automáticamente iniciando con el valor que se le indica en *inicio* que es el primer valor a insertarse en la tabla e *incremento* es un valor que indica en cuánto se incrementa el siguiente valor a insertarse.

  Provee un valor único e incremental, y se suele utilizar en conjunto con la **PRIMARY KEY**. Solo puede ser asignada a tipos de datos numéricos, y puede existir solo una columna con esta propiedad por tabla.
- **CONSTRAINT**: esta propiedad es opcional y se utiliza para hacer cumplir la integridad de los datos. Algunas de las Constraint que se pueden crear son:

- **PRIMARY KEY**: para establecer la clave principal o primaria de la tabla
- **FOREIGN KEY**: para establecer las claves secundarias o foráneas de la tabla con respecto a otras.

- Lo que está entre **[ ]** es opcional incorporarlo o no en la sentencia.
- Lo que se separa por la barra vertical "**|**" es que debe elegirse uno o el otro.
- Las palabras en mayúsculas son palabras reservadas del lenguaje es decir que deben escribirse tal como se presenta en la sintaxis y no deben utilizarse para nombres de otros elementos como objetos, campos, etc.
- El Management Studio de SQL Server, por defecto, no es "case sensitivity" es decir no distingue mayúsculas y minúsculas por lo que se puede escribir la sentencia toda en minúsculas.

Tipos de datos

Como se sabe, una base de datos está compuesta de tablas (entidades) donde almacenamos registros catalogados en función de distintos campos (atributos). Un aspecto previo a considerar es la naturaleza de los valores que introducimos en esos campos. Dado que una base de datos trabaja con todo tipo de información, es importante especificarle qué tipo de valor se le está introduciendo de manera que, por un lado, facilite la búsqueda posterior y por otro, optimice los recursos de memoria, lo que se conoce como integridad de dominio.

Es importante especificar qué tipo de información será permitida en cada una de las columnas que conforman la estructura de la fila. Por ejemplo, si se desea almacenar precios de productos en una columna se debería especificar que el tipo de datos sea numérico, más específicamente moneda, si se desea almacenar nombres se debe escoger un tipo de dato que permita almacenar información de tipo carácter.

Cada tipo de dato en SQL Server tendrá un nombre propio, algunos de los cuales se muestran en la tabla siguiente:

Tipo genérico	Tipo de dato almacenado	Nombre en SQL Server	Cantidad de bytes que ocupa en el almacenamiento
Binario	Almacenan cadenas de bits expresados en números hexadecimales.	**binary**	Longitud fija (hasta 8 Kb)
		varbinary	Puede variar en el nro. de dígitos hexadecimales (hasta 8 Kb.)
		image	Longitud variable y puede exceder los 8 Kb.
Alfanumérico o Caracter	Combinación de letras, símbolos y números.	**char**	Longitud fija (hasta 8 Kb)
		varchar	Puede variar el nro. de caracteres (hasta 8Kb.)
		text	Pueden ser caracteres ASCII y exceder los 8 Kb.
Fecha y Hora	Fechas y horas (día, mes y año y hora, minuto y segundos según el formato del sistema	**datetime**	Fechas entre el 01/01/1753 hasta 31/12/9999 (8 by)
		smalldatetime	Fechas entre el 01/01/1900 hasta 06/06/2079 (4 by)
Numéricos Decimal (punto fijo)	Números con decimales de coma fija.	**decimal**	Pueden tener hasta 38 dígitos
		numeric	Equivalente al decimal.
Numéricos Decimal (punto flotante)	Números con decimales aproximados	**float**	Entre 1,79E + 308 hasta 1,79 + 308
		real	Entre 3,40E + 38 hasta 3,40 + 38.
Numéricos Enteros	Datos numéricos enteros positivos o negativos. Tinyint no tiene negativos	**bigint**	Entre 2^63 hasta 2^63 - 1 (8 by)
		int	Entre -2,147,483,648 hasta 2,147,483,648 (4by)
		smallint	Entre -32,768 hasta 32,767 (2by)
		tinyint	Entre 0 y 255 (1 by)
Numérico Moneda	Numérico decimal para cantidades monetarias positivas o negativas	**money**	Entre 922,337,203,685,477,5808 positivos y negativos (8 by)
		smallmoney	Entre 214,748,3648 (4 by)
	Especiales	**bit**	Datos que consisten de 1 o 0. Similar al True o False

Tipo genérico	Tipo de dato almacenado	Nombre en SQL Server	Cantidad de bytes que ocupa en el almacenamiento
		cursor	Es empleado en variables o procedimientos almacenados.
		timestamp	Indica la actividad que ocurre sobre una fila.
		uniqueidentifier	Es un identificador único de la fila dentro de la base de datos
		sql_variant	Almacena varios tipos de datos excepto el tect, ntext, timestamp, image.
		table	Almacena un resultado de una consulta.
Unicode	Se puede almacenar datos Unicode. Utilizan 2 byte por cada carácter	**nchar**	Longitud fija hasta 4000 caracteres Unicode.
		nvarchar	Longitud variable hasta 4000 caracteres unicode
		Ntext	Exceden los 4000 caracteres unicodes.

Tabla 2: Elaboración propia.

Ejemplo de creación de tablas

Se quiere crear las tablas cuya estructura se muestra a continuación:

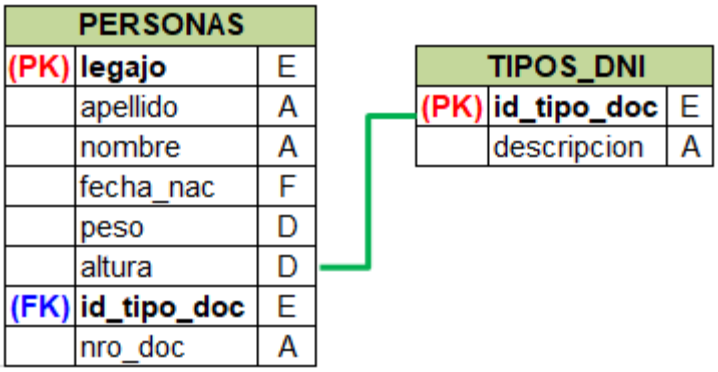

Gráfico 1: Elaboración propia.

Se van a crear las tablas con sus claves primarias y foráneas en el mismo momento por ello por integridad referencial se creará primero la tabla tipos_dni ya que su clave primaria está referenciada por una clave foránea de personas:

```
CREATE TABLE tipos_dni
(id_tipo_doc int IDENTITY (1,1),
 descripcion varchar(15)
```

```
 CONSTRAINT pk_tipo_doc PRIMARY KEY (id_tipo_doc)
)
CREATE TABLE personas
(legajo int,
 apellido varchar(50) not null,
 nombre varchar(50),
 fecha_nac datetime,
 peso decimal(4,2),
 altura decimal(3,2),
 id_tipo_doc int,
 nro_doc varchar(8)
 CONSTRAINT pk_persona PRIMARY KEY (legajo),
 CONSTRAINT fk_persona_tipo_doc  FOREIGN KEY (id_tipo_doc)
            REFERENCES tipos_dni (id_tipo_doc)
)
```

La propiedad identity (1,1) del campo id_tipo_doc de la tabla tipos_dni indica que el primer tipo de documento ingresado en esa tabla llevará el valor 1, y el resto se le incrementará uno al siguiente por lo que el segundo llevará en número 2, el tercero, 3 y así sucesivamente.

TIPOS_DNI	
id_tipo_doc	descripcion
1	DNI
2	Pasaporte
3	LC
4	...

Gráfico 2: Elaboración propia.

El campo descripción se supone que como máximo tendrá 15 caracteres. A este tipo de datos es necesario indicarle el tamaño máximo sino se asume 1. Los campos tipo decimal llevan dos números en el tamaño, el primero es el tamaño total (cantidad de enteros y cantidad de decimales) y el segundo es la cantidad de decimales. En cambio, el tipo int no requiere tamaño ya que es fijo al igual que el tipo datetime y el resto de los tipos de datos enteros.

A las claves primarias no es necesario indicarle “not null” ya que una vez definida la constraint primary key por definición no aceptará nulos. Para los otros campos, como apellido será necesario indicarle que no se aceptan nulos, en caso que no se indique nada se asume que se permiten valores nulos (valores no definidos).

En la definición de la clave primaria:

```
 CONSTRAINT pk_persona PRIMARY KEY (legajo),
```

Se tiene luego de la palabra clave **CONSTRAINT** el nombre de la constraint que puede ser cualquiera que no exista previamente como objeto en la misma base de datos, pero, como los nombres de las tablas, debe hacer referencia a lo que es. Luego va el tipo de restricción **PRIMARY KEY** y por último el campo sobre el cual se crea la clave primaria entre paréntesis.

En la definición de la clave foránea:

```
CONSTRAINT fk_persona_tipo_doc FOREIGN KEY (id_tipo_doc)
           REFERENCES tipos_dni (id_tipo_doc)
```

Se tiene luego de la palabra clave **CONSTRAINT** el nombre de la constraint que puede ser cualquiera que no exista previamente como objeto en la misma base de datos. Luego va el tipo de restricción **FOREIGN KEY** , seguido por el campo sobre el cual se crea la clave foránea entre paréntesis, posterior a ello, (puede ser sobre la misma línea, aquí no entro en el ancho de la hoja) **REFERENCES** la tabla que contiene la clave primaria que referencia y por último el campo, entre paréntesis, que es PK de la tabla que es extremo uno en la relación.

Las comas se utilizan para separar elementos que son del mismo tipo por ello va separando los campos y luego de la constraint primary key.

Modificación de una tabla

Con la sentencia ALTER se puede: modificar la estructura o definición de un objeto, agregar o eliminar componentes del mismo.

En el caso particular de una tabla, se pueden modificar, agregar o eliminar restricciones (constraint), columnas, nombres de columnas, tipos de datos, etc.

Para agregar una columna:

```
ALTER TABLE nombre_tabla
ADD nombre_columna  tipo_dato
```

Ejemplo: se quiere agregar la contextura física en personas

```
ALTER TABLE personas
ADD contextura_fisica varchar(20)
```

Para eliminar una columna:

```
ALTER TABLE  nombre_tabla
DROP COLUMN  nombre_columna
```

Ejemplo: se quiere eliminar la columna o el campo contextura_fisica de la tabla personas

```
ALTER TABLE personas
DROP COLUMN contextura_fisica
```

Para modificar el tipo de dato:

```
ALTER TABLE  nombre_tabla
ALTER COLUMN  nombre_columna tipo_dato
```

Ejemplo: se quiere modificar el tamaño de la columna o el campo nro_doc de la tabla personas

```
ALTER TABLE personas
ALTER COLUMN nro_doc varchar(10)
```

Para agregar una clave primaria:

```
ALTER TABLE nombre_tabla
ADD CONSTRAINT nombre_clave PRIMARY KEY (columna_PK)
```

Ejemplo: se quiere agregar la clave primaria a la tabla títulos que no se determinó cuando se creó la tabla

```
Alter table titulos
add constraint pk_titulos primary key (id_titulo)
```

Eliminación de una tabla

Con la sentencia DROP se puede borrar definitivamente una tabla de la base de datos y todo su contenido. Sintaxis:

```
DROP TABLE nombre_table
```

Ejemplo: se quiere eliminar la tabla personas

```
drop table personas
```

BIBLIOGRAFÍA

Gorman K., Hirt A., Noderer D., Rowland-Jones J., Sirpal A., Ryan D. & Woody B (2019) Introducing Microsoft SQL Server 2019. Reliability, scalability, and security both on premises and in the cloud. Packt Publishing Ltd. Birmingham UK

Microsoft. SQL Server 2016. Disponible en: https://www.microsoft.com/es-es/sql-server/sql-server-2016

Opel, A. & Sheldon, R. (2010). Fundamentos de SQL. Madrid. Editorial Mc Graw Hill

Varga S., Cherry D., D'Antoni J. (2016). Introducing Microsoft SQL Server 2016 Mission-Critical Applications, Deeper Insights, Hyperscale Cloud. Washington. Microsoft Press

Imágenes extraídas del Banco de Datos Gratuito Pixabay. Disponible en:

https://pixabay.com/es/