BASE DE DATOS

LENGUAJE SQL
LENGUAJE DML

GRUPO 01T
PERIODO II
SEMANA 8

Consultas de acción (Insert, Update y Delete)

Objetivos

1. Agregar información a una o varias tablas almacenadas en una base de datos

2. Actualizar datos almacenados en una base de datos tomando ciertos criterios

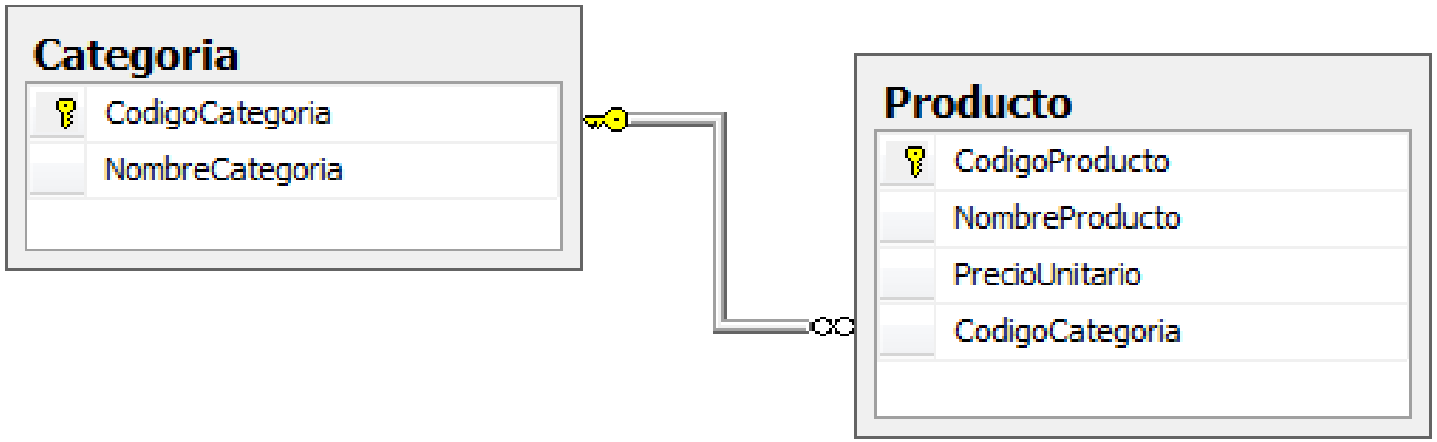
3. Eliminar datos o información innecesaria almacenada en una base de datos

Consultas de acción

- Las consultas de acción son aquellas consultas que no devuelven ningún registro, sino que se encargan de acciones como:
  - Agregar registros (INSERT)
  - Actualizar registros (UPDATE)
  - Eliminar registros (DELETE)

Consultas de acción

- Se tiene el siguiente diagrama de base de datos



Sentencia INSERT

- INSERT. Permite agregar, adicionar o insertar uno o más registros a una (y solo una) tabla en una base de datos relacional.

- Debe proporcionar en una sentencia INSERT igual número de campo y columnas.

- Si se van a proporcionar valores para todos los campos de una tabla pueden omitirse los nombres de dichos campos en la instrucción.

Sentencia INSERT

- Sintaxis
  - Forma1

INSERT INTO <NOMBRE_TABLA>
VALUES (<LISTA DE DATOS)

  - Forma 2

INSERT INTO <NOMBRE_TABLA> <LISTA_CAMPOS>
VALUES (<LISTA DE DATOS)

  - Forma 3

INSERT INTO <NOMBRE_TABLA>
VALUES (<LISTA DE DATOS1), (<LISTA DE DATOS2), (<LISTA DE DATOSN)

Sentencia INSERT

- Ejemplos:

```
--Insertando sin colocar los nombres de los campos esto indica que se
--debe agregar datos a todas las columnas NOT NULL y se debe tomar
--en cuenta el orden de los campos de la tabla
INSERT INTO Categoria VALUES
(1,'Bebidas')

--Colocando los nombres de los campos, no importa el orden de como esten
--los campos en la tabla
INSERT INTO Categoria (CodigoCategoria,NombreCategoria)VALUES
(2,'Carnes rojas')

INSERT INTO Categoria (NombreCategoria,CodigoCategoria)VALUES
('Harinas',3)


--Agregando varios registros al mismo tiempo en la tabla
INSERT INTO Categoria VALUES
(4,'Vegetales'),
(5,'Frutas'),
(6,'Mariscos')
```

Sentencia INSERT

- En el siguiente ejemplo se agrega datos a la tabla Producto en donde se respeta la relación entre las tablas, quiere decir que no se puede agregar un código de categoría si este no ha sido ingresado previamente en la tabla donde se encuentre la llave primaria (Tabla Categoria)

```
INSERT INTO Producto VALUES
(1, 'Soda Coca Cola', 1.25,1),
(2, 'Carne bistec',3.50,2),
(3, 'Camarones pequeños',1.15,6),
(4, 'Harina blanca',0.75,3),
(5, 'Te verde',1.0,1),
(6, 'Lomo de aguja',4.50,2),
(7, 'Soda de naranja',1.25,1),
(8, 'Chiles verdes',0.25,4),
(9, 'Tomates',0.2,4),
(10, 'Manzana verde',0.25,5)
```

Este dato indica el código de la categoría, siguiendo el ejemplo, aquí solo se pueden agregar datos del 1 al 6, los cuales son los datos almacenados en el campo CodigoCategoria de la tabla Categoria

Sentencia SELECT INTO

- SELECT INTO se utiliza para crear una tabla a partir de los valores de otra.

SELECT <LISTA_CAMPOS> **INTO** <NOMBRE_NUEVA_TABLA>

FROM <NOMBRE_TABLA>

Ejemplo:

- Se desea crear una tabla con los datos de la tabla producto que pertenezcan a la categoría bebidas

```
SELECT * INTO [Producto CategoriaBebidas]
FROM Producto
WHERE CodigoCategoria=1
```

Sentencia SELECT INTO

- Al hacer un SELECT a la tabla Producto se tienen los siguientes resultados:

CodigoProducto	NombreProducto	PrecioUnitario	CodigoCategoria
1	Soda Coca Cola	1.25	1
2	Carne bistec	3.50	2
3	Camarones pequeños	1.15	6
4	Harina blanca	0.75	3
5	Te verde	1.00	1
6	Lomo de aguja	4.50	2
7	Soda de naranja	1.25	1
8	Chiles verdes	0.25	4
9	Tomates	0.20	4
10	Manzana verde	0.25	5

El código de la categoría Bebidas tiene el valor de 1

- Después de ejecutar la sentencia SELECT INTO y al hacer un SELECT a la tabla Producto CategoriaBebidas, se obtienen los siguientes resultados:

CodigoProducto	NombreProducto	PrecioUnitario	CodigoCategoria
1	Soda Coca Cola	1.25	1
5	Te verde	1.00	1
7	Soda de naranja	1.25	1

Sentencia INSERT INTO – SELECT

- La consulta SELECT de la instrucción INSERT se puede utilizar para agregar valores a una tabla de la base de datos.

- Sintaxis:

INSERT INTO <NOMBRE_TABLADESTINO>

SELECT <LISTA_CAMPOS>

FROM <NOMBRE_TABLAORIGEN>

Sentencia INSERT INTO – SELECT

Ejemplo:

```
 --Crear la tabla Producto CategoriaVegetales, con las mismas propiedas
 --de la tabla Producto
CREATE TABLE [Producto CategoriaVegetales]
(CodigoProducto int NOT NULL,
NombreProducto varchar(50),
PrecioUnitario decimal(18,2),
CodigoCategoria int
CONSTRAINT pk_producto1 PRIMARY KEY (CodigoProducto)
CONSTRAINT fk_categoria1 FOREIGN KEY (CodigoCategoria)
REFERENCES Categoria(CodigoCategoria)
)
```

Al hacer un SELECT a la tabla esta no tiene datos

```
SELECT * FROM [Producto CategoriaVegetales]
```

Sentencia INSERT INTO – SELECT

```
--En el siguiente ejemplo, la instrucción INSERT agrega en la tabla Producto CategoriaVegetales,
--los datos de la tabla Producto donde el valor del campo CodigoCategoria sea igual a 4

INSERT INTO [Producto CategoriaVegetales]
 SELECT CodigoProducto,NombreProducto,PrecioUnitario,CodigoCategoria
 FROM Producto
 WHERE CodigoCategoria=4
```

- Al hacer un SELECT a la tabla esta debe tener los datos de los productos que pertenecen a la categoría con código igual a 4

```
SELECT * FROM [Producto CategoriaVegetales]
```

	CodigoProducto	NombreProducto	PrecioUnitario	CodigoCategoria
1	8	Chiles verdes	0.25	4
2	9	Tomates	0.20	4

- **Nota:** a diferencia con la sentencia SELECT INTO, aquí debe de crearse la tabla previamente

Sentencia UPDATE

- UPDATE. Permite la actualización o modificación de uno o varios registros de una única tabla.
- Se debe utilizar en conjunto con la cláusula SET con la cual se indicará(n) el(los) campo(s) a actualizar con el valor indicado.
- Una segunda cláusula WHERE, **opcional**, permite indicar qué registros deben ser actualizados.
- Si se omite la cláusula WHERE la ejecución de la consulta modificará todos los registros de la tabla.
- Sintaxis:

  UPDATE <NOMBRE_TABLA>

  SET <NOMBRE_CAMPO> =<NUEVO_VALOR>

Sentencia UPDATE

Ejemplos:

- Ejemplo 1. Actualizando datos a varios registros

```
--Actualiza el dato almacenado en el campo PrecioUnitario de cada registro
--de la tabla Producto CategoriaBebidas
UPDATE [Producto CategoriaBebidas] SET PrecioUnitario=1.50
```

- Al hacer un SELECT a la tabla se obtiene los siguientes resultados:

CodigoProducto	NombreProducto	PrecioUnitario	CodigoCategoria
1	Soda Coca Cola	1.50	1
5	Te verde	1.50	1
7	Soda de naranja	1.50	1

Sentencia UPDATE

- Ejemplo 2. Actualizando datos donde se cumpla una o varias condiciones

- Una condición:

```
--Actualiza el precio del producto donde el Codigo del producto
--sea igual 1
UPDATE [Producto CategoriaBebidas] SET PrecioUnitario=1.25
WHERE CodigoProducto=1
```

- Al hacer un SELECT a la tabla se obtiene los siguientes resultados:

CodigoProducto	NombreProducto	PrecioUnitario	CodigoCategoria
1	Soda Coca Cola	1.25	1
5	Te verde	1.50	1
7	Soda de naranja	1.50	1

Sentencia UPDATE

- Ejemplo 2. Actualizando datos donde se cumpla una o varias condiciones

- Varias condiciones:

```
--Actualiza el precio del producto donde el Codigo del producto
--sea igual 1 y el codigo de la categoria sea igual 1
UPDATE [Producto CategoriaBebidas] SET PrecioUnitario=1.75
WHERE CodigoProducto=1 AND CodigoCategoria=1
```

- Al hacer un SELECT a la tabla se obtiene los siguientes resultados:

	CodigoProducto	NombreProducto	PrecioUnitario	CodigoCategoria
1	1	Soda Coca Cola	1.75	1
2	5	Te verde	1.50	1
3	7	Soda de naranja	1.50	1

Sentencia DELETE

- DELETE. Permite eliminar o borrar todos los registros de una tabla.

  Sintaxis:

  DELETE FROM <NOMBRE_TABLA>

- La sentencia DELETE no borra la estructura física de la tabla únicamente elimina los datos.

- Ejemplos:

```
--Elimina todos los registros de la tabla Producto CategoriaBebidas
DELETE FROM [Producto CategoriaBebidas]

--Elimina los registros de la tabla Producto donde el codigo de la
--categoria sea igual a 4
DELETE FROM Producto
WHERE CodigoCategoria=4
```

ON DELETE CASDADE – ON UPDATE CASCADE

- Debido a que el sistema de gestión de base de datos hace cumplir las restricciones de referencia, se debe garantizar la integridad de los datos, si las filas de la tabla de la clave principal se van a eliminar o van a ser actualizadas, el gestor verifica si todavía existen filas dependientes en tablas de claves foráneas, esas referencias tienen que ser consideradas.

ON DELETE CASDADE

- Específica que si se intenta eliminar una fila con una clave primaria a la que hacen referencia claves foráneas de filas existentes en otras tablas, todas las filas que contienen dichas claves foráneas también se eliminan.

ON UPDATE CASCADE

- Específica que si se intenta actualizar un valor de clave primaria de una fila a cuyo valor de clave hacen referencia claves foráneas de filas existentes en otras tablas, también se actualizan todos los valores que conforman la clave foránea al nuevo valor especificado para la clave primaria.

ON DELETE CASDADE – ON UPDATE CASCADE

- Ejemplo.

Se quiere eliminar de la tabla la categoría donde el código sea igual a 6

```
DELETE FROM Categoria WHERE CodigoCategoria =6
```

Pero se obtiene el siguiente error:

```
Msg 547, Level 16, State 0, Line 1
The DELETE statement conflicted with the REFERENCE constraint "fk_categoria". The conflict occurred in database "Guia7", table "dbo.Producto", column 'CodigoCategoria'.
The statement has been terminated.
```

El cual indica que existe una referencia externa con ese dato que queremos eliminar, para verificar hacemos un SELECT a la tabla Producto

Para corregir el error se debe agregar a la restricción de clave foránea las instrucciones ON DELETE CASCADE y ON UPDATE CASCADE

ON DELETE CASDADE – ON UPDATE CASCADE

- Agregando las sentencias ON DELETE CASCADE y ON UPDATE CASCADE con la instrucción ALTER

```
-- Agregando la restricción nuevamente pero ahora se adiciona al final de la restricción
-- las sentencias  ON DELETE CASCADE y ON UPDATE CASCADE
ALTER TABLE Producto
ADD CONSTRAINT fk_categoria
FOREIGN KEY (CodigoCategoria) REFERENCES Categoria (CodigoCategoria)
ON DELETE CASCADE
ON UPDATE CASCADE
```

ON DELETE CASDADE – ON UPDATE CASCADE

- Ejemplo. Agregando las sentencias ON DELETE CASCADE y ON UPDATE CASCADE a nivel de tabla o cuando esta es creada

```
-- Creando una nueva tabla con el nombre Producto CategoriaFrutas

CREATE TABLE [Producto CategoriaFrutas]
(CodigoProducto int NOT NULL,
NombreProducto varchar(50),
PrecioUnitario decimal(18,2),
CodigoCategoria int
CONSTRAINT pk_producto2 PRIMARY KEY (CodigoProducto)
CONSTRAINT fk_categoria2 FOREIGN KEY (CodigoCategoria)
REFERENCES Categoria(CodigoCategoria)
ON DELETE CASCADE
ON UPDATE CASCADE
)
```