BASE DE DATOS AVANZADO I

Unidad 2: Lenguaje de manipulación de datos DML

Tema 4: Lenguaje para la manipulación de datos DML

Tema 4: Lenguaje para la manipulación de datos DML

Capacidades

1. Identifica los comandos de manipulación de datos y sus diferentes opciones.
2. Implementa sentencias optimizadas para consultar, ingresar y eliminar registros así también actualiza datos.

1. Lenguaje para la manipulación de datos

¿Qué es DML?

- El lenguaje de manipulación de datos (DML) es un conjunto de comandos utilizado para recuperar y trabajar con datos en base de datos. La mayoría también funciona en Data Warehouse.
- Estos comandos son usadas para agregar, modificar, consultar o eliminar datos de una base de datos.

Estas operaciones insertan, modifican, eliminan y consultan los datos almacenados en las tablas pero no su estructura, ni su definición.[4]

1. Lenguaje para la manipulación de datos

¿Cuáles son los comandos DML?

- Los comandos DML son:

2. Insertar registros

INSERT INTO

- Permite la inserción de nuevos datos en una tabla, se realiza añadiendo filas enteras a la tabla.
- La inserción se puede realizar de una fila o de varias filas a la vez. Veremos las dos opciones por separado y empezaremos por la inserción de una fila.

2. Insertar registros

INSERT INTO

Inserción Individual de una FILA

- Para realizar la inserción individual de filas SQL posee la instrucción INSERT INTO.
- Su sintaxis es la siguiente:

```
INSERT INTO Nombre_tabla
[(nombre_columna1, nombre_columna1,nombre_columna n..)]
VALUES (expr1, expr2, expr n...)
```

2. Insertar registros

INSERT INTO

Inserción Individual de una FILA

- Insertar nuevo registro en los campos idcategoria y nombrecategoria de la tabla Categorías ubicada en el esquema Compras de la base de datos Negocios.

```
Use Negocios
go

Insert Compras.categorias
(IdCategoria, NombreCategoria)
Values
('99','Cereales')
go
```

2. Insertar registros

INSERT INTO

Inserción Múltiples FILAS

- Permite insertar varios registros en una tabla.
- Utilice una combinación de la sentencia INSERT junto a una sentencia SELECT. Se insertan todos los registros devueltos por la consulta. Su sintaxis es la siguiente:

```
INSERT INTO Nombre_tabla
[(nombre_columna1, nombre_columna2,..)]
SELECT [(<campo1>[,<campo2>,...])]
FROM <nombre_tabla_origen>;
```

2. Insertar registros

INSERT INTO

Inserción Múltiples FILAS

- Insertar todos los registros de la tabla Categories de la Base de Datos NorthWind hacía la tabla categoría de la Base de Datos Negocios.

```
Use Negocios
go

Insert Compras.categorias
(IdCategoria, NombreCategoria)
Select CategoryID, CategoryName from Northwind.dbo.Categories
go
```

2. Insertar registros

BULK INSERT

- Importa un archivo de datos en una tabla o vista de base de datos con un formato especificado por el usuario en SQL Server.
- Esta instrucción transfiere datos eficazmente entre SQL Server y orígenes de datos heterogéneos. (Flat File). Su sintaxis es:

```
BULK INSERT  Nombre_tabla
FROM [origen de datos]
WITH (      FIRSTROW='# de fila del primer registro',
            FIELDTERMINATOR='separador de columna',
            ROWTERMINATOR='separador de Fila');
```

2. Insertar registros

BULK INSERT

- Importar los datos de los clientes almacenados en el archivo de texto ubicado en la unidad D: Clientes.txt hacia la tabla Clientes del esquema Ventas de la base de datos Negocios.

```
Use Negocios
go

BULK INSERT Ventas.Clientes
FROM 'D:\Clientes.txt'
WITH (FIELDTERMINATOR=',');
go
```

3. Eliminar registros

- **DELETE**

- Para borrar datos de una tabla, debemos utilizar la sentencia **DELETE.**
- Su sintaxis es la siguiente:
  - **DELETE FROM** Nombre_Tabla [WHERE { condición }]
- Donde:
- *Nombre_Tabla nombre de la tabla donde se desea borrar los datos. La cláusula WHERE sigue el mismo formato que la vista en la*
- *sentencia SELECT y determina qué registros se borrarán.*

3. Eliminar registros

DELETE

Eliminar los registros de la tabla Clientes del esquema Ventas, cuyo código inicie con Co.

```
Use Negocios
go

Delete From Ventas.clientes
Where IdCliente Like 'C0%'
go
```

3. Eliminar registros

- **TRUNCATE TABLE**

- Para realizar un borrado completo de tabla debemos considerar la posibilidad de utilizar la sentencia TRUNCATE, mucho más rápida que DELETE.

- Su sintaxis es la siguiente:

    - **TRUNCATE** TABLE Nombre_Tabla

- ***TRUNCATE*** *no es transaccional. No se puede deshacer.* ***TRUNCATE*** *no admite clausula* ***WHERE****. Borra toda la tabla. No todos los gestores de bases de datos admiten la sentencia* ***TRUNCATE****.*

3. Eliminar registros

TRUNCATE TABLE

Para efecto de comprobar este comando, creamos una tabla llamada Consumer, con un campo Identity. Luego Insertamos nombres a esta tabla desde la tabla Clientes.

```
Use Negocios
go

Create Table dbo.Consumer
(
id          int identity,
fullname    varchar(50)
)
go

Insert dbo.Consumer
Select NomCliente from Ventas.clientes
go
```

3. Eliminar registros

TRUNCATE TABLE

Ejemplo: Eliminar todos los registros de la tabla Consumer.

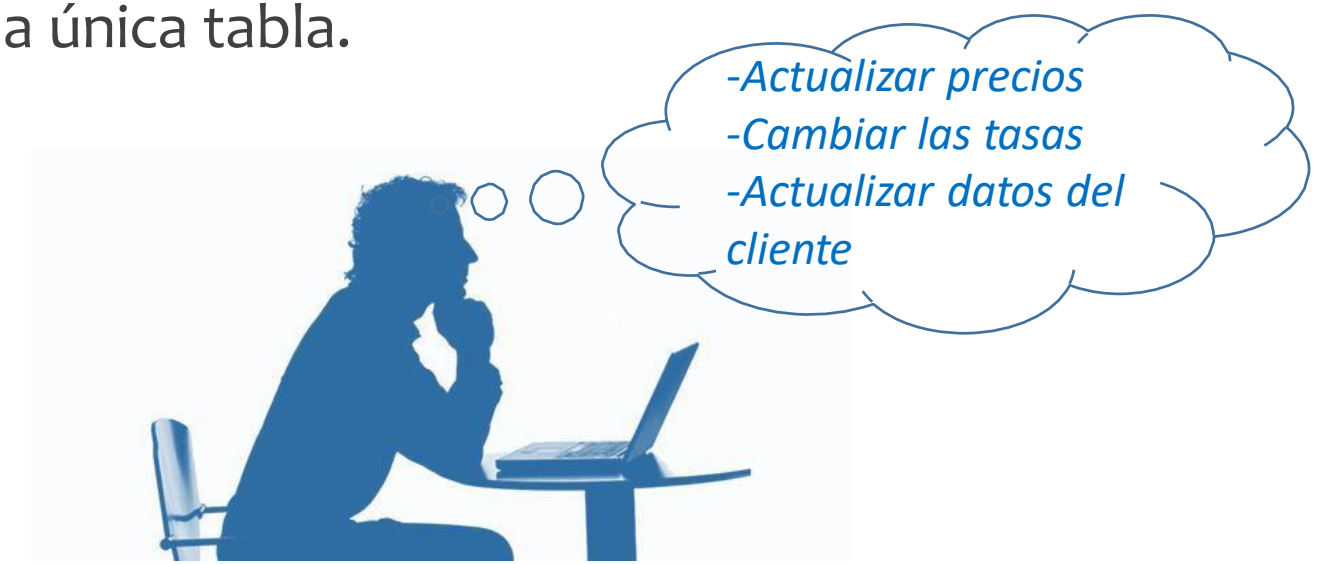
```
Use Negocios
go

Truncate Table dbo.Consumer
go
```

***TRUNCATE** no sólo elimina, sino también resetea el valor identity.*

4. Actualizar datos

UPDATE

- Para la actualización de datos SQL dispone de la sentencia UPDATE.
- La sentencia UPDATE permite la actualización de uno o varios registros de una única tabla.



4. Actualizar datos

UPDATE

- Su sintaxis es la siguiente:

```
UPDATE Nombre_tabla
SET    nombre_columna1 = expr1,
       nombre_columna2 = expr2,..
[WHERE { condición }]
```

Nombre_Tabla *tabla donde se actualiza los datos.*
Nombre_columna *es el nombre de columna o campo cuyo valor se desea cambiar.*
Expr *el nuevo valor que se desea asignar al campo que le precede.*

4. Actualizar datos

UPDATE

Ejemplo: Actualizar los datos del Cliente CACTU.

```
Use Negocios
go

Update Ventas.clientes
Set      NomCliente  = 'Bembos Burger',
         DirCliente  = 'Las Begonias 444',
         idpais      = '001',
         fonoCliente = '(51)12832474'
Where IdCliente = 'CACTU'
go
```

4. Actualizar datos

UPDATE

Ejemplo: Actualizar el precio de los productos incrementando 10%, solo si han sido suministrados por proveedores de Colombia.

```
Update Compras.productos
Set  PrecioUnidad = PrecioUnidad * 1.10
Where IdProveedor IN (Select IdProveedor
                      from Compras.proveedores
                      Where idpais IN (Select Idpais
                                       From Ventas.paises
                                       where NombrePais='Colombia'))
go
```

5. Sincronizar registros

MERGE

- Permite realizar múltiples acciones sobre una tabla tomando uno o varios criterios de comparación, es decir, realiza operaciones de inserción, actualización o eliminación en una tabla de destino según los resultados de una combinación con una tabla de origen.
- Por ejemplo, puede sincronizar dos tablas insertando, actualizando o eliminando las filas de una tabla según las diferencias que se encuentren en la otra.



5. Sincronizar registros

MERGE

- Nos sirve básicamente para dos cosas:

Sincronizar los datos de 2 tablas. Suponga que tenemos 2 bases distintas (Producción y Desarrollo por ejemplo) y queremos sincronizar los datos de una tabla para que queden exactamente iguales.

Cuando tenemos nuevos datos que queremos almacenar en una tabla y no sabemos si la Llave Primaria ya existe o no, por lo tanto, no sabemos si hacer un UPDATE o un INSERT en la tabla.

5. Sincronizar registros

MERGE

- Su sintaxis es la siguiente:

```
MERGE [INTO] <target table>
USING <source table>
ON <join/merge predicate>
WHEN MATCHED <statement to run
WHEN NOT MATCHED <statement to run>;
```

5. Sincronizar registros

MERGE

- Ejemplo:
- Implemente un escenario para actualizar o insertar un registro a la tabla países: Si existe el código del país, actualice su nombre; sino inserte el registro a la tabla

5. Sincronizar registros

MERGE

```
Use Negocios
go

Begin
Declare @v_IdPais Char(3), @v_NomPais Varchar(100)
Set @v_IdPais   = '009'
Set @v_NomPais  = 'Brasil'
    Merge Ventas.Paises As Target
    Using (Select @v_IdPais, @v_Nompais)  As Source (IdPais, NombrePais)
    On (Source.IdPais = Target.IdPais)
    When Matched Then
        Update Set Target.NombrePais = Source.NombrePais
    When Not Matched Then
        Insert Values (Source.idPais, Source.NombrePais);
End
go
```

Ejercicio 1

- En la empresa Recobra, se realiza inventario mensualmente, debiendo registrarse el id del Producto, fecha y la cantidad o si ya existe el Id del producto, actualizar la fecha y cantidad. La tabla en donde se actualiza o registra es Inventario_Producto.

Ejercicio 1: Solución

- Solución:

```
Use BDRecobra
go

Declare @v_idProducto   int
Declare @v_cantidad     int
Set @v_idProducto=10
Set @v_cantidad=300
Merge into dbo.InventarioProducto as Target
Using (Select @v_idProducto,@v_cantidad) as Source (idProducto,cantidad)
On (Target.idProducto=Source.idProducto)
When Matched Then
    Update Set  Target.fecInventario=getdate(),
                Target.cantidad=Source.cantidad
When Not Matched Then
    Insert Values (Source.IdProducto, getdate(), Source.Cantidad);
go
```

Ejercicio 2

- El área de sistemas emite diariamente un archivo plano con información de las transacciones realizadas en las diferentes cuentas de los clientes, debiendo cargarlo en una tabla llamada Operaciones.
- El archivo plano está ubicado en el disco D:\ Transacciones.TXT y tiene el siguiente formato:

# Transacción	cuenta	tipo	fecha	monto	agencia
1230	12345-10	Depósito	10/12/17	1200	Higuereta
1245	23456-12	Retiro	21/12/17	500	Plaza Norte

Ejercicio 2: Solución

```
Use BDRecobra
go

Bulk Insert dbo.Operaciones
From 'D:\Transacciones.TXT"
With ( FirstRow          -  .2
       FieldTerminator ↪  ·1·,
       RowTerminator     -  "\n')

go
```

Conclusiones

- Los comandos DML nos permiten gestionar Datos.
- El comando BULK INSERT lo utilizamos para hacer una inserción masiva desde un archivo plano.
- El comando TRUNCATE TABLE, elimina totalmente todos los registros de una tabla, es mas rápido, pero no admite WHERE.
- El comando MERGE es de mucha utilidad cada que necesitemos realizar 02 o mas transacciones según sea el caso.

Bibliografía

- Microsoft (2017) Merge Transact SQL. Recuperado de: https://docs.microsoft.com/en-us/sql/t-sql/statements/merge-transact-sql?view=sql-server-2017

- Microsoft (2017) Bulk Insert (Transact-SQL). SQL. Recuperado de: https://docs.microsoft.com/en-us/sql/t-sql/statements/bulk-insert-transact-sql?view=sql-server-2017

- Microsoft (2017) Insert (Transact-SQL) : https://docs.microsoft.com/es-es/sql/odbc/microsoft/insert-sql-command?view=sql-server-2017

GRACIAS

SEDE LIMA CENTRO
Av. Uruguay 514
Cercado – Lima
Teléfono: 419-2900

SEDE INDEPENDENCIA
Av. Carlos Izaguirre 233
Independencia – Lima
Teléfono: 633-5555

SEDE BREÑA
Av. Brasil 714 – 792
(CC La Rambla – Piso 3)
Breña – Lima
Teléfono: 633-5555

SEDE TRUJILLO
Calle Borgoño 361
Trujillo
Teléfono: (044) 60-2000

SEDE SAN JUAN DE LURIGANCHO
Av. Próceres de la Independencia 3023-3043
San Juan de Lurigancho – Lima
Teléfono: 633-5555

SEDE BELLAVISTA
Av. Mariscal Oscar R. Benavides 3866 – 4070
(CC Mall Aventura Plaza)
Bellavista – Callao
Teléfono: 633-5555

SEDE AREQUIPA
Av. Porongoche 500
(CC Mall Aventura Plaza)
Paucarpata - Arequipa
Teléfono: (054) 60-3535