CURSO BASE DE DATOS 2
(#08)
TRIGGERS

BIENVENIDAS / BIENVENIDOS

Grupo N3G REM

A/P Jorge Mario Benitez Ruiz,
DSI
Jorge.Benitez@fi365.ort.edu.uy

- **Inicio puntual** 19:30 hs.
- **Es deseable** CAMARA ENCENDIDA y
- **Se recomienda** MICROFONO en Mute **al Inicio**

Disparadores (Triggers)

Para que usamos los Triggers:

- Notificación cuando ocurren ciertas condiciones
- Reforzar las restricciones de integridad
  - Los disparadores son mas potentes que las restricciones
- Mantenimiento de datos derivados
  - Actualización automática de datos derivados evitando anomalías debidas a la redundancia.
  - Ejemplo: Un disparador actualiza el saldo total de una cuenta bancaria cada vez que se inserta, elimina o modifica un movimiento de dicha cuenta

Disparadores (Triggers)

- Cumplimiento de Evento-Momento-Accion

Disparadores (Triggers)

- EVENTO

FUENTE – que ocasiona la ocurrencia de un evento ?

•Una instrucción DML (antes o despúes) : INSERT, DELETE, UPDATE

•Una instrucción para gestión de transacciones: COMMIT, ROLLBACK

•Una excepción: bloqueo, violación de autorización, etc.

•El reloj: el 10 de Junio a las 13:45hs

•La aplicación: externo a la BD

Disparadores (Triggers)

- Momento

Momento en que ocurre el EVENTO

- Antes (**BEFORE**)
- Después (**AFTER**)
- En lugar de (**INSTEAD OF**) *equivalente al BEFORE*

Disparadores (Triggers)

- **Acción**

ACCIÓN – Que se puede incluir en la reacción ?

•Operación en la BD: ordenes de SQL, INSERT, DELETE …

•Comandos de SQL extendido: PL/SQL, T-SQL

•Llamadas externas: envío de mensajes

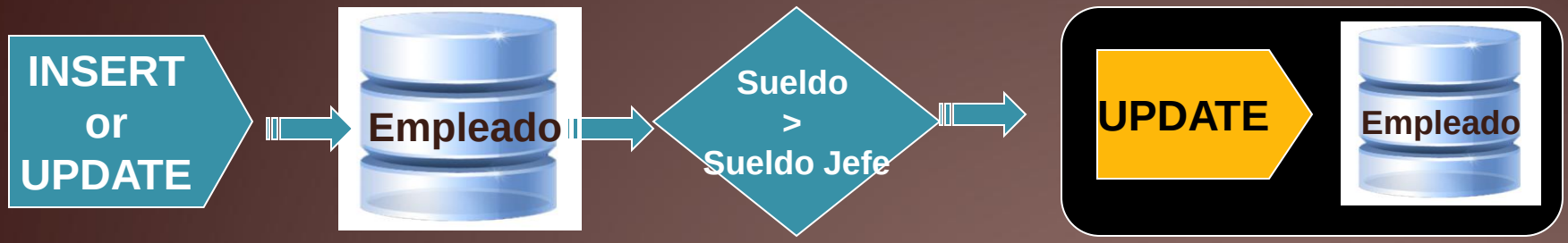
•Abortar la transacción

Disparadores (Triggers)

- Está asociado a una única tabla base
- Es el concepto clave para implementar **BD activas**
- Tiene 3 partes:
  - Un **evento**: indica la acción sobre la tabla base que causará se active el disparador (INSERT, UPDATE, DELETE)
  - Un **tiempo de acción**: indica cuando se activará el disparo
    - BEFORE = Antes del evento
    - AFTER = Después del evento
  - Una **acción**: se llevan a cabo si ocurre el evento, puede ser de dos tipos:
    - Sentencia SQL
    - Un bloque atómico de sentencias SQL

Disparadores (Triggers)

Restricción: “Sueldo empleado mayor que del jefe, se equipara el sueldo del Jefe”



Evento	**after**	**Insert or update** on Empleado
Condición	**If :new.sueldo > (select sueldo from empleado where cargo=‘jefe’)**	
Acción	**Update Empleado Set sueldo=:new.sueldo Where cargo=‘jefe’**	

Disparadores (Triggers) T-SQL

SINTAXIS del TRIGGER:

```
CREATE TRIGGER <Nombre_Trigger>
ON <Nombre_Tabla>
AFTER INSERT,DELETE,UPDATE
AS
BEGIN
      COMANDOS de T-SQL
END
```

Disparadores (Triggers) T-SQL

Es necesario conocer las tablas: **inserted** y **deleted**.

Los triggers DML utilizan dos tablas especiales denominadas: **inserted** y **deleted**.
SQL Server crea y administra automáticamente ambas tablas.

La estructura de **las tablas inserted y deleted** es la misma que **tiene la tabla** que ha desencadenado la ejecución del trigger.

Disparadores (Triggers) T-SQL

La tabla (**inserted**) solo está disponible en las operaciones **INSERT** y **UPDATE** y en ella están los valores resultantes después de la inserción o actualización.

Es decir, los datos insertados. **Inserted** estará vacía en una operación **DELETE**.

En la segunda (**deleted**), disponible en las operaciones **UPDATE** y **DELETE**, están los valores anteriores a la ejecución de la actualización o borrado.

Es decir, los datos que serán borrados. Deleted estará vacía en una operación **INSERT**.

Disparadores (Triggers) T-SQL

¿Existe una tabla **UPDATED**?

No. Hacer una actualización es lo mismo que borrar (**deleted**) e insertar los nuevos (**inserted**).

La sentencia **UPDATE** es la única en la que inserted y deleted **tienen datos simultáneamente**.

No se pueden modificar directamente los datos de las tablas: inserted y deleted.

Disparadores (Triggers) T-SQL

El siguiente **ejemplo**, graba un histórico de saldos cada vez que se modifica un saldo de la tabla cuentas.

```
CREATE TRIGGER TR_CUENTAS
ON CUENTAS
AFTER UPDATE
AS
BEGIN
SET NOCOUNT ON;
INSERT INTO HIS_SALDOS(IDCUENTA, SALDO, FXSALDO)
                    SELECT IDCUENTA, SALDO, getdate()
                    FROM INSERTED

END
```

Una consideración a tener en cuenta es que el trigger se ejecutará aunque la instrucción DML (UPDATE, INSERT o DELETE) no haya afectado a ninguna fila. En este caso inserted y deleted devolverán un conjunto de **datos vacío**.

Disparadores (Triggers) T-SQL

Cargamos una tabla auditoria luego de insertar un registro sobre la tabla Shippers

```
CREATE TRIGGER trg_audit_insert
ON Shippers
AFTER INSERT
AS
   DECLARE @NewValue AS char(40)
BEGIN

 SELECT @NewValue = CompanyName FROM inserted
 INSERT INTO Audit VALUES
      ('SHIPPERS','INSERT','',@NewValue,getdate())

END
```

Disparadores (Triggers) T-SQL

Cargamos una tabla auditoria luego de modificar un registro sobre la tabla Shippers

```
CREATE TRIGGER trg_audit_delete
ON Shippers
AFTER DELETE
AS
   DECLARE @NewValue AS char(40)
   DECLARE @OldValue AS char(40)
BEGIN

   SELECT @NewValue = CompanyName FROM inserted
   SELECT @OldValue = CompanyName FROM deleted
   INSERT INTO Audit VALUES
        ('SHIPPERS','UPDATE',@OldValue,@NewValue,getdate())

END
```

Disparadores (Triggers) T-SQL

Disparadores INSTED OF (sustituto de BEFORE de PL/SQL)

"instead of": sobreescribe la acción desencadenadora del trigger.

Se puede definir solamente un disparador de este tipo para cada acción (insert, delete o update) sobre una tabla o vista.

Sintaxis:

```
create trigger NOMBREDISPARADOR
on NOMBRETABLA o VISTA
instead of
ACCION  insert, update o delete
as
SENTENCIAS
```

Disparadores (Triggers) T-SQL

Por **ejemplo**:

Forzar a que siempre se ingrese el nombre de la tabla Shippers en letras mayúsculas

```
CREATE TRIGGER trgInsert
   ON  Shippers
   INSTEAD OF insert
AS
DECLARE
 @CompanyName nvarchar(40),
 @Phone nvarchar(24)
BEGIN
    SELECT @CompanyName=CompanyName,@Phone=Phone
    FROM inserted
    INSERT INTO Shippers VALUES(UPPER(@CompanyName),@Phone)
END
```

The following example sends an e-mail message to a specified person (MaryM) when the customer table changes
The following example sends an e-mail message to a specified person (MaryM) when the customer table changes

Disparadores (Triggers) T-SQL

Por ejemplo:
Enviar un e-mail a una persona (MaryM) cuando la Tabla: Customer cambie:

```
CREATE TRIGGER reminder2
ON   Customers
AFTER INSERT, UPDATE, DELETE
AS
   EXEC msdb.dbo.sp_send_dbmail
        @profile_name = 'Northwind Administrator',
        @recipients = 'jbensom@northwind.com',
        @body = 'No olvidar imprimir informe Clientes para EquipoVentasn.',
        @subject = 'Recordatorio';
GO
```

The following example sends an e-mail message to a specified person (MaryM) when the customer table changes

Disparadores (Triggers) T-SQL

DDL TRIGGERS

```
CREATE TRIGGER safety
ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS
   PRINT Hay que deshabilitar el Trigger "safety" para drop o alter
tablas!'
   ROLLBACK;
GO
```

DESHABILITAR o HABILITAR TRIGGERS ya EXISTENTES

```
DISABLE TRIGGER safety ON DATABASE;

GO
```

The following example sends an e-mail message to a specified person (MaryM) when the customer table changes

Disparadores (Triggers) T-SQL

DESHABILITAR o HABILITAR TRIGGERS ya EXISTENTES

```
ENABLE TRIGGER { [ schema_name . ] trigger_name [ ,...n ] | ALL }
ON { object_name | DATABASE | ALL SERVER } [ ; ]

DISABLE TRIGGER { [ schema_name . ] trigger_name [ ,...n ] | ALL }
ON { object_name | DATABASE | ALL SERVER } [ ; ]
```

```
CREATE TRIGGER safety
ON DATABASE
FOR DROP_TABLE, ALTER_TABLE
AS
   PRINT 'You must disable Trigger "safety" to drop or alter tables!'
   ROLLBACK;
GO

DISABLE TRIGGER safety ON DATABASE;
GO
ENABLE TRIGGER safety ON DATABASE;
GO
```

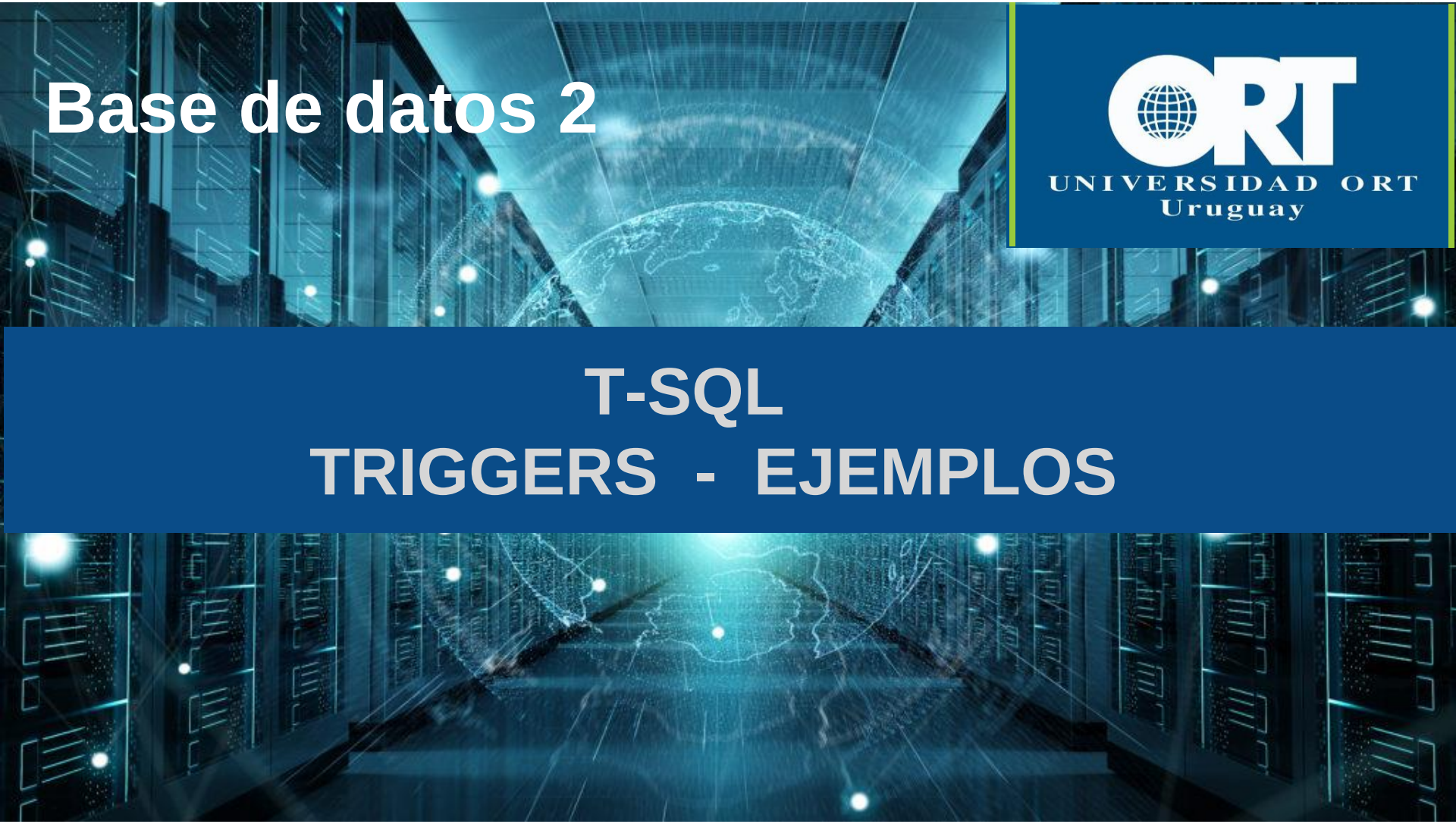
Disparadores (Triggers) Ejemplo

Crearemos una Database **DEMOTRIGGERS** con las Tablas: **Productos**, Historial, **Ventas Compras**.

Para la Tabla **Productos** llevaremos un Historial que registre los cambios producidos por los comandos SQL: **INSERT**, **DELETE** y **UPDATE**.

Para ello Crearemos en la Tabla **Productos** los **triggers** para **INSERT , DELETE y UPDATE**

Ver: SQLQuery_Ejercicio de Clase TRIGGERS demo .sql



Base de datos 2

UNIVERSIDAD ORT
Uruguay

T-SQL
TRIGGERS - EJEMPLOS

UNIVERSIDAD ORT
Uruguay