Universidad
Inca Garcilaso de la Vega
Nuevos Tiempos. Nuevas Ideas
Facultad de Ingeniería de Sistemas, Cómputo y Telecomunicaciones

Disparadores

Asignatura: Gestión de Base de Datos
Semana 13
Docente: Mg. Christian Almóguer Martínez
Mail: almoguer@uigv.edu.pe

AGENDA

- Disparadores
- Ventajas
- Formas de aplicar Disparadores.

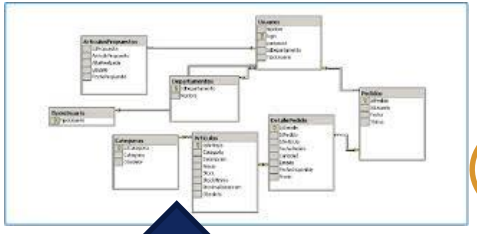
Disparadores

- Son módulos que se ejecutan de manera automática al insertar, eliminar o actualizar datos de una tabla ocasionados por estos eventos ocurridos en la base de datos.
- Estos se almacenan como objetos.
- La diferencia entre un procedimiento es que este último se ejecuta de manera explicita por el usuario, una aplicación o un triggers. ***En cambio, los disparadores de manera implícita.***

Ventajas

- Ayuda a mantener la integridad, permitiendo conservar la consistencia entre los datos relacionados.

- Son automáticos, porque funcionan cualquiera sea el origen de la modificación de los datos.

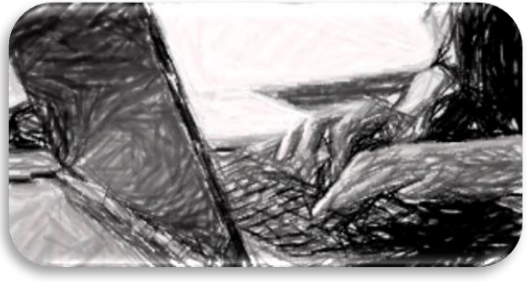


Existe un trigger definido para esta tabla

Alumnos : Tabla

Id_Alumno	Nombre	Apellidos	email	Dirección	Localidad	CP
3	Pepo	Pérez Pepín	pepo@pepolano	c/ del Pepo sn	Pepoland	50014
4	Godofredo	Lopez Perez	godo@godolano	Pº Independenc	Zaragoza	50015
5	Lalo	Lalín Lalero	lalo@lalolano	c/ del Lalo 3	Zaragoza	50015
6	Lola	Lalín Lalero	lola@lalolano	c/ del Lalo 3	Zaragoza	50015
7	Elena	Nito de la Prade	ele@troya.es	c/ Media 2	Zaragoza	50001
8	Benito	Camela Pronto	beni@hill.es	c/ Londres 34	Zaragoza	50010
(Autonumérico)						

Tabla



Comando DML

- Insert,
- Update
- Delete

Estructura de un Trigger

- **Activación**: es la sentencia que permite "disparar" el código a ejecutar. Conocido como evento.
- **Restricción**: es la condición necesaria para realizar el código. Esta restricción puede ser de tipo condicional o de tipo nulo.
- **Acción** : es la secuencia de instrucciones a ejecutar una vez que se han cumplido las condiciones inícials.

Formato de un Trigger

Create [or replace] **Trigger** Nombre_trigger
[before | after | instead of]
[<evento> or <evento>] [Of Campo] ON <tabla> ← activación
[FOR EACH ROW]
[WHEN (condición| old |new)] ← restricción
Declare
  variables
Begin
  sentencia(s); ← acción
End;

Características del Formato

- Es posible controlar **más de un evento** en un solo trigger, uniéndolos con la expresión **OR**.

- Los momentos de activación de un disparador son: **Before, After e Instead Of** .

- **Instead Of** , hace que se ejecute la acción del trigger en vez de ejecutar la sentencia DML que lo activa. Este no es posible usarlo con tablas sólo con vistas.

- La cláusula **OLD** es usada para registros viejos y **NEW** para registros nuevos. Cuando se usen los nombres en la parte de acción, deberán ir precedidos de dos puntos (:), ejemplo **:New.Campo_tabla.**

Tipos de Trigger

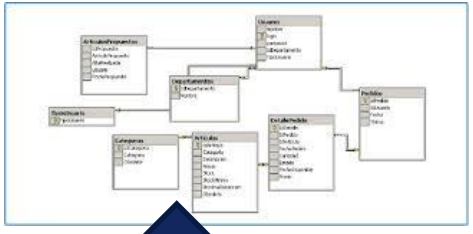
Se clasifican según la cantidad de ejecuciones a realizar:

- **Triggers a nivel de fila**: son aquellas que se ejecutaran una vez por cada fila procesada por la orden DML activadora.

- **Triggers de secuencia**: son aquellos que sin importar la cantidad de veces que se cumpla con la condición, su ejecución es única.

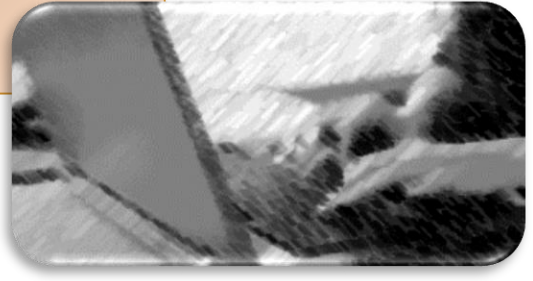
Momento de Activación
BEFORE

- Este parámetro permite ejecutar un trigger ANTES que la acción DML se ejecute sobre la base de datos.



1 Trigger

2 Comando DML
- Insert,
- Update
- Delete



Alumnos : Tabla

Id_Alumno	Nombre	Apellidos	email	Dirección	Localidad	CP
3	Pepo	Pérez Pepín	pepo@pepolanc	c/ del Pepo sn	Pepoland	50014
4	Godofredo	Lopez Perez	godo@godolanc	Pº Independenc	Zaragoza	50015
5	Lalo	Lalín Lalero	lalo@lalolanc	c/ del Lalo 3	Zaragoza	50015
6	Lola	Lalín Lalero	lola@lalolanc	c/ del Lalo 3	Zaragoza	50015
7	Elena	Nito de la Prade	ele@troya.es	c/ Media 2	Zaragoza	50001
8	Benito	Camela Pronto	beni@hill.es	c/ Londres 34	Zaragoza	50010
	(Autonumérico)					

Tabla

BEFORE

Ejemplo: Con Before

- Crear una tabla con el nombre **tem_salario.**
- Luego , un disparador que permita agregar un registro a la tabla creada, antes de ejecutarse la acción de actualización en la tabla empleado.

Ejemplo 1: Crear Tabla

```
create table  tem_salario
(
Fecha date not null,
Promedio number,
Situacion  varchar(30)
)
```

Ejemplo 1: Crear Trigger /Before

```
CREATE or replace TRIGGER t_ejemplo1
BEFORE UPDATE ON employees
Declare
 Prom number(9,2);
Begin
   Select avg(salary) into prom from employees;
  Insert Into tem_salario
  Values (sysdate, prom, ' Insertado por trigger ');
End;
```

Ejemplo 1: Verificar trigger

1. Select * from tem_salario;
2. Select * from employees;
3. Update employees
   set salary=salary – 500
   where employee_id= 105;
3. Select * from employees;
4. Select * from tem_salario;

Ejemplo 2: con BEFORE

- Crear una secuencia con el nombre **seq_numero,** con valor inicial 1000 e incremento en 1.
- Crear una tabla con el nombre **tem_sal_anterior.**
- Luego, crear un disparador con el nombre **t_ejemplo2** que permita agregar un registro a la tabla creada, siempre y cuando el sueldo antes de la actualización para cada registro sea menor a 5000.

Ejemplo 2: Crear Secuencia y Tabla

```
Create sequence  seq_numero
Start with 1000
Increment by 1
```

```
Create table  tem_sal_anterior
( Operacion number not null primary key,
  Sal_ant number,
  Sal_act number,
  situacion  varchar(30)
)
```

Ejemplo 2: Before y Condición

```
CREATE or replace TRIGGER t_ejemplo2
BEFORE Update Of salary ON employees
For each row
      when (old.salary< 5000)
Begin
Insert  into  tem_sal_anterior
Values (seq_numero.nextval,
                    :old.salary,
                    :new.salary,
                    'Insertado por trigger');
End;
```

Ejemplo 2: Verificar Trigger

1. Select * from employees order by 7;

104	Bruce	Ernst	BERNST	590.423.4568	21/05/07	IT_PROG	6000	(null)	103	60
105	David	Austin	DAUSTIN	590.423.4569	25/06/05	IT_PROG	4800	(null)	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	02/05/06	IT_PROG	4800	(null)	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	02/07/07	IT_PROG	4200	(null)	103	60
103	Alexander	Hunold	AHUNOLD	590.423.4567	01/03/06	IT_PROG	9000	(null)	102	60

Ejemplo 2: Verificar Trigger

```
Update employees
    set  salary=salary+550
    where  job_id='IT_PROG‘

Select * from employees order by 7;
```

104	Bruce	Ernst	BERNST	590.423.4568	21/05/07	IT_PROG	6550	(null)	103	60
105	David	Austin	DAUSTIN	590.423.4569	25/06/05	IT_PROG	5350	(null)	103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	02/05/06	IT_PROG	5350	(null)	103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	02/07/07	IT_PROG	4750	(null)	103	60
103	Alexander	Hunold	AHUNOLD	590.423.4567	01/03/06	IT_PROG	9550	(null)	102	60

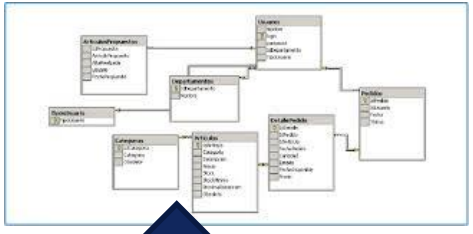
Ejemplo 2: Verificar Trigger

```
Select * from tem_sal_anterior;
```

1	1000	4800	5350	Insertado por trigger
2	1001	4800	5350	Insertado por trigger
3	1002	4200	4750	Insertado por trigger

Momento de Activación
AFTER

- Este parámetro permite ejecutar un trigger DESPUES que la acción DML se ejecute sobre la base de datos.



1

2

Comando DML

- Insert,
- Update
- Delete

Alumnos : Tabla

Id_Alumno	Nombre	Apellidos	email	Dirección	Localidad	CP
3	Pepo	Pérez Pepín	pepo@pepoland	c/ del Pepo sn	Pepoland	50014
4	Godofredo	Lopez Perez	godo@godoland	Pº Independenc	Zaragoza	50015
5	Lalo	Lalín Lalero	lalo@lalolan	c/ del Lalo 3	Zaragoza	50015
6	Lola	Lalín Lalero	lola@lalolan	c/ del Lalo 3	Zaragoza	50015
7	Elena	Nito de la Prade	ele@troya.es	c/ Media 2	Zaragoza	50001
8	Benito	Camela Pronto	beni@hill.es	c/ Londres 34	Zaragoza	50010
(Autonumérico)						

Tabla

AFTER

Ejemplo: con AFTER

- Crear una tabla con el nombre **tem_sal_mayor.**
- Crear un disparador con el nombre **t_ejemplo3** que permita agregar un registro a la tabla tem_sal_mayor, después de ejecutarse la acción de actualización en la tabla emp.

Ejemplo 3: Crear Tabla

```
create table  tem_sal_mayor
(
correlativo number not null primary key,
mayor        number,
Usuario      varchar(15)
)
```

Ejemplo 3: Crear Trigger /After

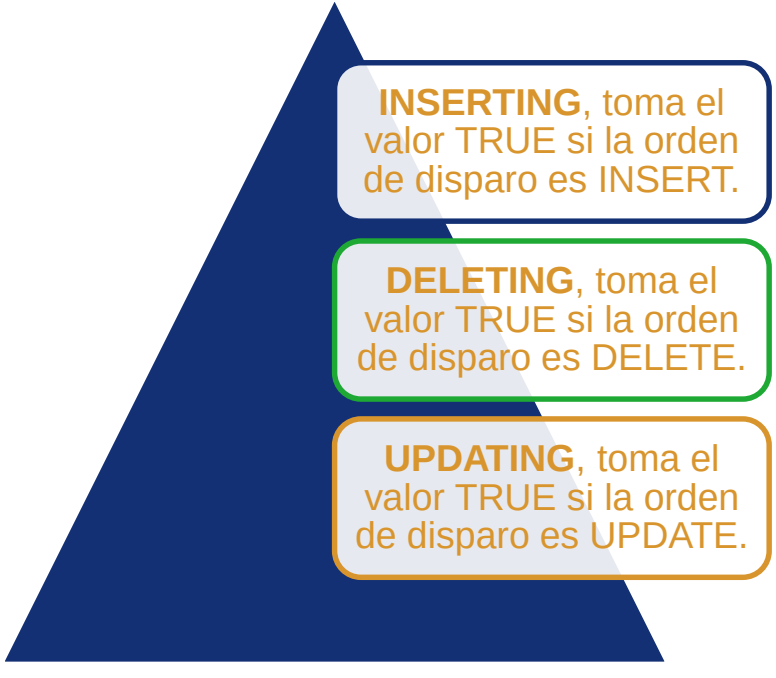
```
CREATE or replace TRIGGER t_ejemplo3
AFTER UPDATE ON employees
Declare
    num number;
    mayor number;
    usuario varchar2(15);
Begin
SELECT nvl(max(correlativo),0)+1 into num
  from tem_sal_mayor;
SELECT max(salary) into mayor
  from employees;
SELECT username into usuario from user_users;
INSERT INTO tem_sal_mayor
VALUES (num, mayor, usuario);
End;
```

Ejemplo 3: Verificar Trigger

1. Select * from employees;
2. update employees
   set salary=salary + 500
   where employee_id=104;
3. Select * from employees;
4. Select * from tem_sal_mayor;

Predicados en Triggers

Se puede utilizar tres predicados de tipo boolean para conocer la operación disparadora:



INSERTING, toma el valor TRUE si la orden de disparo es INSERT.

DELETING, toma el valor TRUE si la orden de disparo es DELETE.

UPDATING, toma el valor TRUE si la orden de disparo es UPDATE.

Ejemplo 4: Con Predicados

- Crear una secuencia con el nombre **seq_orden** con valor inicial 1000 e incremento en 2.
- Crear una tabla con el nombre **control.**
- Crear un disparador con el nombre **t_ejemplo4** que permita agregar un registro a la tabla creada según el modo de transaccion (insert, update ó delete).

Ejemplo 4: Crear Secuencia y Tabla

```
Create sequence seq_orden
Start with 1000
Increment by 2
```

```
Create table control
(
Orden number not null primary key,
fecha date,
operacion varchar(50)
)
```

Ejemplo 4: Trigger/Predicados

```
CREATE or replace TRIGGER t_ejemplo4
  after  UPDATE or INSERT or DELETE on employees
DECLARE
  mensaje varchar2(50);
BEGIN
  IF UPDATING THEN
        mensaje  := ‘Fila actualizada en tabla Empleado';
  ELSIF INSERTING THEN
        mensaje := ‘Fila Insertada en tabla Empleado ';
  ELSE
        mensaje:= ‘Fila Eliminada en tabla Empleado ';
  END IF;
-- Proceso de Inserción
INSERT into control
VALUES (seq_orden.nextval, SYSDATE, MENSAJE);
END;
```

Probando el Trigger

- Update employees
- Set salary=salary*1.20
- Where employee_id=200

- Select * from control

- Insert into employees
- Values(555,’Juan’,’Meza’,’jmeza’,’123,456,7890’, sysdate, ‘MK_MAN’, 12000, 0.12,100, 20)