Tarefa 1. Crear disparadores para actualizar atributos derivados

A tarefa consiste en crear e probar o funcionamento dos disparadores necesarios na base de datos *tendabd* para poder manter actualizado o valor das columnas *clt\_vendas* e *clt\_ultima\_venda* na táboa *clientes*. A columna *clt\_vendas* garda información do número de ventas que se lle fixeron ao cliente, e a columna *clt\_ultima\_venda* garda información da data na que se lle fixo a última venda.

Solución

* Código de creación

Hai que crear tres disparadores para a táboa vendas, para as operacións AFTER INSERT, AFTER UPDATE, e AFTER DELETE.

Antes hai que executar a sentenza SHOW TRIGGERS para saber se xa hai algún disparador para esas operacións:

**show** **triggers** **from** tendabd**;**

Despois de confirmar que non hai disparadores asociados a esas operacións, escríbese o guión de sentenzas para crear os tres disparadores:

-- disparador que actualiza as columnas clt\_vendas e clt\_ultima\_venda despois de

-- inserir unha fila na táboa de vendas

**drop** **trigger** **if** **exists** tendabd**.**vendasAI**;**

**delimiter** **//**

**create** **trigger** tendabd**.**vendasAI **after** **insert** **on** tendabd**.**vendas

**for** **each** **row**

**begin**

**update** clientes

**set** clt\_vendas **=** **ifnull(**clt\_vendas**,**0**)+**1**,**

clt\_ultima\_venda **=** **date(new.**ven\_data**)**

**where** clt\_id **=** **new.**ven\_cliente**;**

**end**

**//**

**delimiter;**

-- disparador que actualiza as columnas clt\_vendas e clt\_ultima\_venda despois de

-- modificar unha fila na táboa de vendas

**drop** **trigger** **if** **exists** tendabd**.**vendasAU**;**

**delimiter** **//**

**create** **trigger** tendabd**.**vendasAU **after** **update** **on** tendabd**.**vendas

**for** **each** **row**

**begin**

-- actualización da columna clt\_vendas

**if** **old.**ven\_cliente **!=** **new.**ven\_cliente **then**

**update** clientes

**set** clt\_vendas **=** clt\_vendas**-**1

**where** clt\_id **=** **old.**ven\_cliente**;**

**update** clientes

**set** clt\_vendas **=** clt\_vendas**+**1

**where** clt\_id **=** **new.**ven\_cliente**;**

**end** **if;**

-- actualización da columna clt\_ultima\_venda

**if** **date(new.**ven\_data**)** **>** **(select** clt\_ultima\_venda

**from** clientes

**where** clt\_id **=** **new.**ven\_cliente**)**

**then**

**update** clientes

**set** clt\_ultima\_venda **=** **date(new.**ven\_data**)**

**where** clt\_id **=** **new.**ven\_cliente**;**

**end** **if;**

**end**

**//**

**delimiter;**

-- disparador que actualiza as columnas clt\_vendas e clt\_ultima\_venda despois de

-- borrar unha fila na táboa de vendas

**drop** **trigger** **if** **exists** tendabd**.**vendasAD**;**

**delimiter** **//**

**create** **trigger** tendabd**.**vendasAD **after** **delete** **on** tendabd**.**vendas

**for** **each** **row**

**begin**

-- actualización da columna clt\_vendas

**update** clientes

**set** clt\_vendas **=** clt\_vendas**-**1

**where** clt\_id **=** **old.**ven\_cliente**;**

-- actualización da columna clt\_ultima\_venda

**if** **date(old.**ven\_data**)** **=** **(select** clt\_ultima\_venda

**from** clientes

**where** clt\_id **=** **old.**ven\_cliente**)**

**and** **(select** **count(\*)**

**from** vendas

**where** ven\_cliente**=old.**ven\_cliente

**and** **date(**ven\_data**)** **=** **date(old.**ven\_data**))** **=** 0

**then**

**update** clientes

**set** clt\_ultima\_venda **=** **(select** **max(date(**ven\_data**))** **from** vendas

**where** ven\_cliente**=old.**ven\_cliente

**and** **date(**ven\_data**)** **!=** **date(old.**ven\_data**))**

**where** clt\_id **=** **old.**ven\_cliente**;**

**end** **if;**

**end**

**//**

**delimiter** **;**

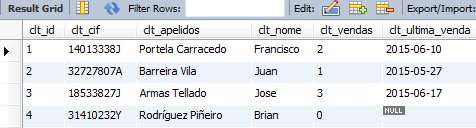
* Proba de funcionamento

Pódese probar o funcionamento dos disparadores inserindo, modificando e borrando unha fila na táboa de vendas:

-- comprobación para a operación de inserción na táboa de vendas:

**select** clt\_id**,** clt\_cif**,** clt\_apelidos**,** clt\_nome**,** clt\_vendas**,** clt\_ultima\_venda

**from** tendabd**.**clientes**;**



**insert** **into** vendas **(**ven\_tenda**,** ven\_empregado**,** ven\_cliente**,** ven\_data**)**

**values** **(**1**,**1**,**1**,**now**());**

**select** clt\_id**,** clt\_cif**,** clt\_apelidos**,** clt\_nome**,** clt\_vendas**,** clt\_ultima\_venda

**from** tendabd**.**clientes**;**

u7a3_tareafa03_probas2.emf

-- comprobación para a operación de modificación na táboa de vendas:

-- a venda anterior cámbiase para o cliente 2

**update** vendas

**set** ven\_cliente **=** 2

**where** ven\_id **=** 151**;**

**select** clt\_id**,** clt\_cif**,** clt\_apelidos**,** clt\_nome**,** clt\_vendas**,** clt\_ultima\_venda

**from** tendabd**.**clientes**;**

u7a3_tareafa03_probas3.emf

-- comprobación para a operación de modificación na táboa de vendas:

-- bórrase a venda anterior (ven\_id=151)

**delete** **from** vendas

**where** ven\_id **=** 151**;**

**select** clt\_id**,** clt\_cif**,** clt\_apelidos**,** clt\_nome**,** clt\_vendas**,** clt\_ultima\_venda

**from** tendabd**.**clientes**;**

u7a3_tareafa03_probas4.emf

Tarefa 2. Crear disparadores para levar rexistros de operacións

A tarefa consiste en crear e probar os disparadores necesarios para levar o rexistro de todas as operacións que modifiquen (*insert, update* e *delete*) os datos almacenados nas táboas que hai no seu esquema *(centro, departamento, empregado*). Para iso débese crear unha táboa na base de datos *traballadores* para o rexistro de todas esas operacións. O código para crear a táboa de rexistro é:

/\*

Creación dunha táboa para levar un rexistro de todas as operacións

que se realicen sobre as táboas da base de datos de traballadores. Cada

operación de manipulación de datos (insert, update, delete) rexistrarase

nesta táboa de forma automática, creando os disparadores necesarios.

\*/

**create** **table** **if** **not** **exists** traballadores**.**rexistroOperacions

**(**

idOperacion **integer** unsigned **not** **null** auto\_increment**,**

usuario **char(**100**),** # usuario que fai oa modificación

dataHora **datetime,** # data e hora na que se fai a modificación

taboa **char(**50**),** # táboa na que se fai a modificación

operacion **char(**6**),** # operación de modificación: INSERT, UPDATE, DELETE

**primary** **key** **(**idOperacion**)**

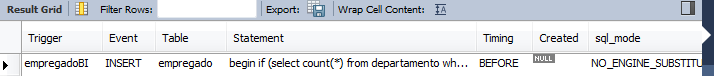
**)engine** **=** myisam**;**

Solución

Hai que crear tres disparadores por cada táboa para as operacións AFTER INSERT, AFTER UPDATE, e AFTER DELETE. En total nove disparadores.

Antes hai que executar a sentenza SHOW TRIGGERS para saber se xa hai algún disparador para esas operacións:

**show** **triggers** **from** traballadores**;**



Despois de confirmar que non hai disparadores asociados a esas operacións, escríbense os guións de sentenzas para crear os disparadores.

* Código de creación do disparador asociado á operación AFTER DELETE da táboa *departamento*

**delimiter** **//**

**create** **trigger** traballadores**.**departamentoAD **after** **delete** **on** departamento

**for** **each** **row**

**begin**

**insert** **into** traballadores**.**rexistroOperacions **(**usuario**,** dataHora**,** taboa**,** operacion**)**

**values** **(user(),now(),**'departamento'**,**'delete'**);**

**end**

**//**

delimiter **;**

* Código de creación do disparador asociado á operación AFTER INSERT da táboa *departamento*

**delimiter** **//**

**create** **trigger** traballadores**.**departamentoAI **after** **insert** **on** departamento

**for** **each** **row**

**begin**

**insert** **into** traballadores**.**rexistroOperacions **(**usuario**,** dataHora**,** taboa**,** operacion**)**

**values** **(user(),now(),**'departamento'**,**'insert'**);**

**end**

**//**

**delimiter** **;**

* Código de creación do disparador asociado á operación AFTER UPDATE da táboa *departamento*

**delimiter** **//**

**create** **trigger** traballadores**.**departamentoAU **after** **update** **on** departamento

**for** **each** **row**

**begin**

**insert** **into** traballadores**.**rexistroOperacions **(**usuario**,** dataHora**,** taboa**,** operacion**)**

**values** **(user(),now(),**'departamento'**,**'update'**);**

**end**

**//**

**delimiter** **;**

O resto dos disparadores para as táboas *empregado* e *centro* terían un código similar ao anterior da táboa *departamento*.

* Proba de funcionamento

As probas serían todas moi parecidas sen máis que cambiar o nome da táboa e a operación a realizar sobre ela. Mostrarase como comprobar o funcionamento do disparador *traballadores.departamentoAD*, executando unha sentenza DELETE sobre a táboa *departamento*, e consultando a táboa *rexistroOperacions*.

**delete** **from** traballadores**.**departamento

**where** depNumero **=** 100**;**

**select** **\*** **from** traballadores**.**rexistroOperacions**;**

