Database Update

What happens when a foreign key accepts nulls?

GeneXus"

GeneXus	
---------	--



There are two ways to update the database: interactively through the transaction object, using its screen, or through code.



1. Business Component: Insert(), Update(), Delete()

	Attraction					
	AttractionId	AttractionId	AttractionName	Countryld	Cityld	Categoryld
	 AttractionName CountryId 	1	Louvre Museum	2	1	1
	CountryName	2	The Great Wall	3	1	2
	- 🔁 CityId - 🖌 CityName	3	Eiffel Tower	2	1	2
	CategoryId	4	Forbidden City	3	1	2
		+				
Insert, Update, Delete						
	Business Compon	ent True				

2. Procedure: New, For Each, Delete

To update through code there were two options:

Using the business component of the transaction, through its methods, or exclusively within a procedure, through the New, For Each commands with direct assignment of the attributes to be modified, and the Delete command.

_	_	
	_	
	=	

	ι	Jniqueness Controls	Refer	rential In Control	
Business Component		~		~	
Proc (new, for each, dele	te)	~			
	Attractionid	AttractionName	Countryld	Cityld	Categoryld
	1	Louvre Museum	2	1	1
	2	The Great Wall	3	1	2
	2	The Great Wall Eiffel Tower	3 2	1	2

As we know, both options control the uniqueness of records, that is, no records with a repeated primary or candidate key are ever allowed to be left in the database. In the example, it will never be possible to have two attractions with the same identifier. And if, for example, AttractionName were a candidate key, no two attractions with the same name would be allowed either.

There's a big difference between updating via a Business Component and doing it via direct update commands in a procedure: only Business Components will execute the logic coming from the rules and events of the transaction, and only in this way will referential integrity be controlled through a program. Why do we say "through a program"?



ne	Туре	Description	Formula	Nullable
Attraction	Attraction	Attraction		I
AttractionId	Id	Attraction Id		No
AttractionName	Name	Attraction Name		No
 AttractionAddress 	Address, GeneXus	Attraction Address		No
- 🔻 CountryId	Id	Country Id		No
🗠 🖌 CountryName	Name	Country Name		
🔻 🤁 CityId	Id	City Id		No
🖌 🖌 CityName	Name	City Name		
	Id	Colores th		Yes
🗠 🏲 CategoryId	10	Category Id		Tes
CategoryId	Name	Category Name		163
Category X Category X Structure S Web Form Rul	Name es Events Variables Patterns	Category Name		
Category X Category X Structure S Web Form Rul	Name es Events Variables Patterns Type	Category Name	Formula	Nullable
Category X Category X Structure S Web Form Rul ame Category	Name es Events Variables Patterns Type Category	Category Name Category Name Description Category	Formula	Nullable
CategoryName	Name es Events Variables Patterns Type	Category Name	Formula	

If we have the Attraction transaction that records the tourist attractions that always belong to a country and city and to some category (like monument, museum, etc.), then clearly in the associated table we will have foreign keys. In particular, the CategoryId attribute will be a foreign key to the Category table.

The GeneXus developer will not have to program in the transaction the referential integrity control for each foreign key because it will already be automatically included in the generated program. It's part of the transaction's logic.

_	еп	-	<i>.</i>	-
	e.		<u>ки</u>	-

	Travel Agency
ttraction	
Id	2
Name	The Great Wall
Country Id	
Country Name	China
City Id	
City Name	Beijing
Category Id	12 Z No matching 'Category'.
Category Name	To making caregory.
	Ş
	CONFIRM CANCEL

So, when the user wants to assign an existing attraction to a nonexistent category, the transaction informs us that this is not possible. It's making a referential integrity control, before the database itself does it. O

	Trave	l Agency	
Attraction Id Category Id Update Attraction BC	0	Update Attraction PROC	
	ţ.		

Let's suppose that we implemented a web panel that allows the user to enter an attraction ID and a category ID, to change the category of the attraction to this other one.

There are two options to do it through code: one is through the Attraction Business Component, and the other will be through a procedure.



Let's see how we would implement the first one.

With the Business Component property activated, we define a variable of this type, and in the event associated with the button we load it from the &AttractionId variable that the user enters through the screen.

If it exists, that is, the Load does not fail, then we change the category for the one entered by the user, and try to update it. This will work just like the transaction, since the code associated with that Update method will check for the existence of that category and only then send the update order to the database. And since that code will find that there is no such category, it will not try to update the database; it will return False and the "else" will be executed.

Let's try it at runtime.



On the other hand, if we want to do the same update through a procedure, as we see here, we send the attraction and category identifier to this procedure, and in it we search for the attraction and the category is directly updated.

When we run it with a non-existent category... ouch! A database exception will occur and the user will have to deal with this unfriendly screen. Here, the program is not performing the necessary referential integrity controls. But the database is making them. This the reason for this error.

We could capture the error to make something more friendly so that the program doesn't cancel abruptly. To this end, the Error_handler rule is used. As a parameter we must enter the name of a subroutine that we must define in the object, and there we will program its behavior in case of a database error, asking first for that error. You can see details about this in our wiki.

tt Preferences	₽ ×	I	Properties	₽	×
TravelAgency		>	🛃 🙀 Filter		×
TravelAgency As the invironment			DataStore: SQL Server		
Generators			Type Data	Store	
C# Default (C# Web)			Description SQL	Server	
CeneXus Cloud			> Access technology settings		
				1	
			Database schema		
			Primary key definition Prim	ary key	
> Patterns				Declare referential integrity Yes	
> 🏪 Workflow			Default tables storage area		
			Default indices storage area		
			Default temporary storage area		
			Generate COMMENT ON statemer No		
			> Database information		
Generation Karage KB Explorer KI Preferences		1	Properties 🎁 Toolbox 🕞 Tests Explorer	€ Tests Results	

We can also remove the referential integrity statements from the database so that it doesn't make the controls. Although it is not recommended, we have this property at the Data Store level in case we need it.



 If select CategoryId from Category where CategoryId =102

AttractionId	AttractionName	Countryld	Cityld	Categoryld
1	Louvre Museum	2	1	1
2	The Great Wall	3	1	102 🗙
3	Eiffel Tower	2	1	2
4	Forbidden City	3	1	2

tructure 🐔 Web Form 1	Rules Events Variables	Patterns		
me	Туре	Description	Formula	Nullable
Attraction	Attraction	Attraction		
- 📍 AttractionId	Id	Attraction Id		No
- 🖓 AttractionName	Name	Attraction Name		No
 AttractionAddress 	Address, GeneXus	Attraction Address		No
- 者 CountryId	Id	Country Id		No
CountryName	Name	Country Name		
- 🔁 CityId	Id	City Id		No
CityName	Name	City Name		
CategoryId	Id	Category Id		Yes

AttractionId	AttractionName	Countryld	Cityld	Categoryld
1	Louvre Museum	2	1	T
2	The Great Wall	3	1	null
3	Eiffel Tower	2	1	2
4	Forbidden City	3	1	2

By default, then, if we don't change that property, the referential integrity may or may not be controlled programmatically, but it will always be controlled by the database.

However, we had seen a case where a non-existent foreign key value was allowed. It's just that strictly speaking, it wasn't a value. That's when we allowed the foreign key to be null. How did we tell the database that we would allow this for a foreign key?

In the transaction structure we had the Nullable column, which was set to No by default, but could be changed to Yes.

6	00	2	(us
	E 11	en	us.

	Travel Agency	
Attraction		
Id	2	
Name	The Great Wall	
Country Id	٤ 🗹 🔊	
Country Name	China	
City Id		
City Name	Beijing	
Category Id		
Category Name		
		CONFIRM

This meant that when the transaction was executed, and the category identifier was left empty, the program didn't make any referential integrity controls, and neither did the database.



	Travel Agency	
traction Id tegory Id Update Attraction BC	d o ↓	Attraction PROC
	A detailine	ni in the code
	<pre>weak submit a spectra spectra provide the trace of means, includes a provide spectra of located of the means the spectra of the spectra</pre>	seflict occurred in database "idflamanda03136/fa0f42ee994a10ed", table Elon(tim) -100000 (Index, hollowen async(lose) +776 m, BulkorgysingleHemalister bulkorgymandler, Td9ParserState00ject statc00j IndexString, Bolen i Isterral, Bolen ferbaccribespaceretrargetion, returnStruam, String mithol, TaxicampletionSurver 1 completion, 1012 t Boolean senforDips, Inti2 Limeout, Booleank usedfacke, Boolean anymetrit accurred in database "idflamanddd03116/fa0f42ee994a10ed", table "dbo.ca

However, if we try to do the same thing, but through the Business Component... ouch! It's giving us basically the same database exception as before, for reference integrity failure. **What's** going on?

Why was it possible to use the transaction but not its Business Component?



The empty value and the null value are two very different things.

The empty value is a value. Depending on the data type, it will be one of the possible values of that type. Thus, if the type is numeric, the empty value is zero. Databases have empty values specified according to each data type. But, strictly speaking, the null value is not a value. In fact, **it's** a NON-VALUE. Saying that Categoryld is Null is the same as saying that its value is not specified and is not known. **It's** not the same as saying it is empty.

6	еп	ν.	1 0
	e 11	<u> </u>	ь.

	em	oty ≠	null				
	For Attractioned 2-2 Categorid 0		AttractionId	AttractionName	Countryld	Cityld	Categoryld
	For AttractionId = 2 \rightarrow CategoryId = 0		1	Louvre Museum	2	1	1
			2	The Great Wall	3	1	2
			3	Eiffel Tower	2	1	2
			4	Forbidden City	3	1	2
_			4 AttractionId	Forbidden City AttractionName	3 Countryld	1 Cityld	2 Categoryld
-	For AttractionId = 2 → CategoryId = null						
	For AttractionId = 2 → CategoryId = null			AttractionName	Countryld	CityId	Categoryld
, I	For AttractionId = 2 → CategoryId = null		Attractionid	AttractionName Louvre Museum	Countryld 2	Cityid 1	CategoryId

Therefore, empty and null values are two very different things in databases. If we are implementing a program and send the order to change the category of attraction 2 to 0 in the database, what we will get is none other than the screen with the database exception for referential integrity failure. On the other hand, if we change it to null, there will be no problem, just like when we did it through the transaction.

So, the question is: Why is the transaction asking the database to change the Null and instead the business component is sending the empty value, zero?



In the properties of the foreign key attribute, Categoryld, we see one called "Nulls in Forms" that is set to "Empty as Null." This means that if the attribute accepts nulls, as in this case, every time the value is left empty in the form, it will be considered to be Null when sent to the database and not empty.

KB Explorer	* ×	🔯 Start Page X 🏹 Test* X 🐌 UpdateAttraction X 🥅 Attraction X	~	O Properties	* ×
n: Name or Pattern		Web Form Rules Events Conditions Variables		2 Filter	>
TravelAgencyBC				1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	_
🔗 Root Module			Attraction Id &attractio	onld	
> 🛅 GeneXus		<pre>1 Event 'Update Attraction' 2 &attraction.Load(&attractionId)</pre>			
AddToCityTour		3 If &attraction.Fail()			
AddToCityTours		<pre>4 msg("The attraction doen't exist")</pre>			
Real AskMinutesDiff		5 else			
Attraction		6 If &categoryId.IsEmpty() 8 &categoryId.SetNull()			€ °
Associated Tables		else	Category Id &category	ld	
> 💦 WorkWithAttraction		9 &attraction.CategoryId = &categoryId	d ×		þ
AttractionInCityTours		10 endif			
> I Category		11 if <u>&attraction.Update()</u> 12 Commit		0	
> 🥅 CityTour		13 msg("Data was successfully updated")			
A CityTourAttraction		14 else	Update Attraction BC	Update Attraction Proc	
Country		15 msg("Data update has failed")			
DeleteFromCityTours		16 endif 17 endif			
Gx0010		18 Endevent			
Gx0020		19			
Gx0031		20 ⊟ Event 'UpdateAttractionProc'			
Gx0040		<pre>21 &message = UpdateAttraction(&attractionId, &categoryId) 22 msg(&message)</pre>			
Gx0050		22 Endourset			
Gx0061		Travel Agency			
ModifyMinsToCityTours		inuveragency			
ModifyMinsToCityToursDP					
Test		Data was successfully updated			
UpdateAttraction					
Domains	A	ttraction Id 2			
DE References					
Customization	C	ategory Id 0			
Generators	L L	Update Attraction BC Update	te Attraction PROC		
Documentation		opone na delor de	io na della mini neo o		
		0			
		Left	×		

But this doesn't apply when the update is made through the Business Component, which doesn't have a form.

Therefore, if the &Categoryld variable is empty, this element will be left with an empty and not null value, and this is why the referential integrity fails when trying to update. We have to specify the null value if the variable is empty.

There are two ways to do so: Either we do it locally, only here, or we do the same mapping that gets the attribute property, but this time also valid in the Business Component, wherever it is used.

For the first thing, you only have to specify that if the Categoryld variable is empty, the Business Component element will be set to null. The SetNull method applies to attributes that accept null values and their corresponding ones in the Business Component variables, as in this case.

If we try it now, we see that we can do it.

The drawback of this alternative is that if we want to leave this foreign key null through a Business Component variable anywhere else, we will have to do the same.

As we were saying, the other, more general option is to incorporate this SetNull as a rule in the transaction. Here we remove it then...



And go to the transaction to specify that the attribute is set to null when its value is left empty. This is already done by default when using the transaction screen, thanks to this property; so, strictly speaking, we can ask that this rule is only executed for the Business Component.

We can always indicate that a rule is only executed when it is in the Web transaction, or when it is in the Business Component, by placing this mark.

Actually, if we want to write a set of rules that only apply to the Web transaction, or that only apply to the Business Component, we do it in this way.

Let's try it. We see that it is indeed another solution.

ategory Id	PROC • • • • • • • • • • • • • • • • • • •
ttraction Id 2 ategory Id 0 Update Attraction BC Update Attraction BC Update Attraction BC Update Attraction BC Update Attraction BC When the UPAT second se	- σ ×
Category Id Update Attraction BC Update	- σ ×
Update Attraction BC Update Attraction Update A	- σ ×
A Maximum	- σ ×
	- σ × ឱង&នារឿដេ
Server Error in //Idf180a9d4651110cf307ff42ea994a10e6' Application. The UPDATE statement conflicted with the FOREIGN KEY constraint "IATTRACTION!", The conflict occurred in database Td180a9d465110cf307ff42ea994a10e6', table "dbo.Category", column "CategoryId". The statement has been terminated. The statement has been terminated been terminated to the terminated been terminated. The statement has been terminated been terminated been terminated been terminated been terminated. The statement has been terminated been terminate	<u> ខ្នែង ឆ្ ឆ្ ៖</u>
The UPDATE statement conflicted with the FOREIGN KEY constraint "IATTRACTION1". The conflict occurred in database "Id19094651104/307442e9994.00e6", table "dbb.Category", column "CategoryId". The statement has been remining the state of the	
<pre>'dd18009465110c/3027f42ea994410e6', table 'dbb.Category', column 'Category'd'.' The statement has been remnined.' The statement has been remnined.' The statement has been remnined and the state of the state o</pre>	
Next Trace: [sqlException (9x80111944): the uPART statement coefficted with the FORLIGE KEY constraint "IATIBACTION". The coeffict occurred in database "Idf180a964451 the statement has keen terminated.] System-Dist-Apl[cont_sqlConcection.okerror(SqlException exception, boolean breakconnection, Action 1 wrapConcEoAction) +330638	tabligosynt.
[allocation (DAMIIIDAI) The UMAIT statement conflicted with the FORION EXY constraint "LATEACTORY", the conflict occurred in database "Lafiabasidain The statement has been treatment.] System.bits.aptic.aptic.aptic.com.oktrony(Splicception exception, boolean breakcomection, Action') wayeloscinokcion +330038	
The statement has been terminated.] System.Data.SqlClinet.SqlConnection.OnError(SqlException exception, Boolean breakConnection, Action'1 wrapCloseInAction) +3306188	
System.Data.sql[linet.Taburer.Threadreys]inoscharing[Taburersitate0jet strate0j, boilan callerusciencetimocia, boilan sayce[line] v76 System.Data.sql[linet.Taburer.Threadreys]inoscharing[Taburersitate0jet strate0j, boilan callerusciencetimocia, boilana sayce[line] v76 System.Data.sql[linet.Taburer.Threadreys]inoscharing[Taburersitate0jet strate0jet] System.Data.sql[linet.sql[linet.sql] System.Data.sql[linet.sql] System.Data.sql[linet.sql] System.Data.sql[linet.sql] System.Data.sql[linet.sql] System.Data.sql[linet.sql] System.Data.sql[linet.sql] System.Data.sql[linet.sql] System.Data.sql] System.Data.sql[linet.sql] System.Data.sql] System.Dat	TdsParserStateObject stateObj DescribeParameterEncryption, rout, Taskå task, Boolean asyn nSource1 completion, Int32 t
[GARGODIASSIGNED_IN: The URBAN SLATEMENT CONFILED with the FORLIGE KEY contraint "LATEMATION". The conflict occurred is database "lafiassoddslikeforr the statement has been tensionaled.] (Genema, Nata.ADD.GCCommund.exectionOperr() -070	f42ea994a18e6", table "dbo.Ca
[GAUGDDIAISTORTION Type:System.Bata.sk]Clint.sk]Exception.OBMs Error Code:S47.the UPBATE statement coefficted with the FOREIGN EXY constraint *IATTMCTDD He statement has been treminated.] Beenda, hata.JAD.ScComman Commander Commander (1) 950 Beenda, hata.JAD.Fac.DobgDeference -excepted (-149)	
GeneKus.Data.HTier.DataStoreProvider.execute(Int32 cursor, Object[] parms, Boolean batch) +848 GeneKus.Programs.uodateattraction.executePrivate() +443	11°. The conflict occurred in .

Finally, what about updating through the procedure? We have exactly the same problem.



The solution here can only be the local one. Let's try it...

And we see that it works.



training.genexus.com wiki.genexus.com