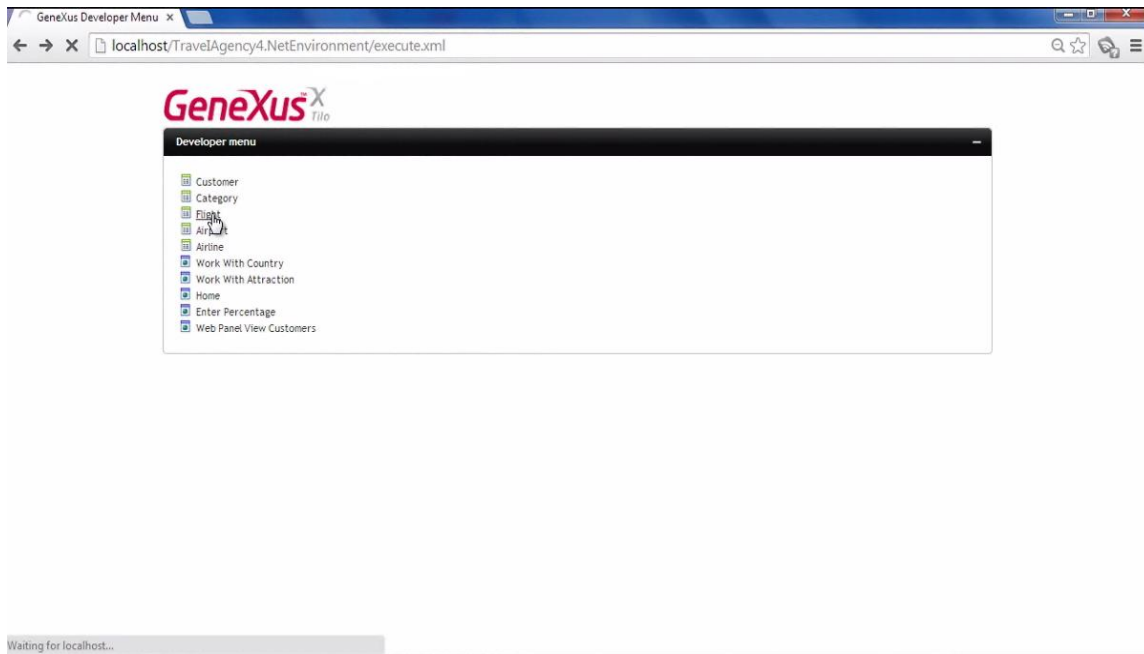
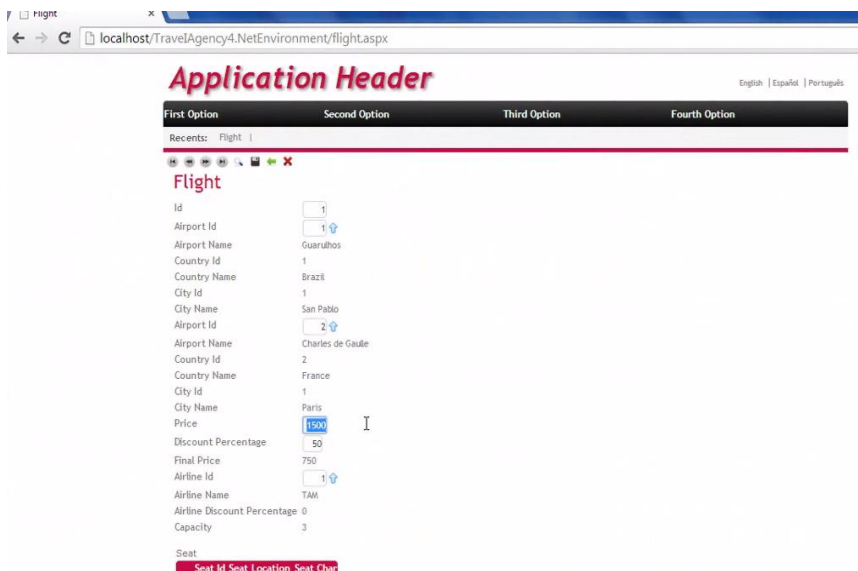


## Actualización de la Base de Datos - For each , delete , new



Hasta el momento, para actualizar los datos de la base de datos, hemos empleado las transacciones en sus 2 formas de uso:

- Ejecutando su pantalla e ingresando datos en forma interactiva



- Y ejecutadas como Business Component, a través de una variable, sin usar la pantalla

**"Flight" transaction :**

**"EnterPercentage" web panel:**

```

Event Enter
For each Flight
  &BCFlight.Load(FlightId)
  &BCFlight.FlightPrice = &BCFlight.FlightPrice * (1+ &Percentage/100)
  &BCFlight.Save()
  If &BCFlight.Success()
    Commit
  Else
    Rollback
  Endif
Endfor
Endevent

```

GeneXus training

Conoceremos ahora una alternativa más para realizar inserciones, modificaciones y eliminaciones en la base de datos.

# Updating the Data Base

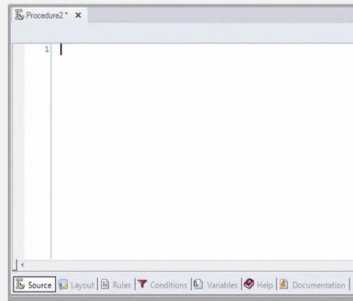
(another way)

GeneXus training

Debemos saber, que lo que veremos **solamente puede ser usado en objetos de tipo procedimiento**

The alternative we will learn, can only be used in...

## PROCEDURES

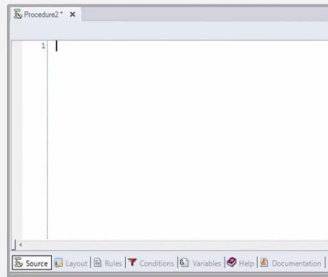


GeneXus training

a diferencia de la alternativa que vimos usando BC, que podía usarse desde cualquier objeto

The alternative we will learn, can only be used in...

## PROCEDURES



**Business Components can be used in any object!**

GeneXus training

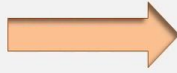
En los procedimientos contamos con un comando llamado New,

para insertar registros en una tabla.

Utilizando este comando, podemos asignarle valores a los atributos **de una tabla física**.

IN PROCEDURES...

New



Endnew

**DATABASE**

**Flight Table**

| Flight Id * | Flight Price | Flight Departure AirportId | Flight Arrival AirportId | Flight Discount Percentage | Flight Airlined |
|-------------|--------------|----------------------------|--------------------------|----------------------------|-----------------|
| 1           | 3000         | 1                          | 2                        | 20                         | 1               |
| 2           | 2250         | 4                          | 3                        | 15                         | 1               |
| 3           | 4500         | 3                          | 1                        | 30                         | 2               |

GeneXus training

Hablamos de una tabla y no de una transacción

Flight Transaction

Name

|                            |
|----------------------------|
| Flight                     |
| FlightId                   |
| FlightDepartureAirportId   |
| FlightDepartureAirportName |
| FlightDepartureCityId      |
| FlightDepartureCityName    |
| FlightArrivalAirportId     |
| FlightArrivalAirportName   |
| FlightArrivalCityId        |
| FlightArrivalCityName      |
| FlightPrice                |
| FlightDiscountPercentage   |
| FlightAirlined             |
| FlightName                 |

**DATABASE**

**Flight Table**

| Flight Id * | Flight Price | Flight Departure AirportId | Flight Arrival AirportId | Flight Discount Percentage | Flight Airlined |
|-------------|--------------|----------------------------|--------------------------|----------------------------|-----------------|
| 1           | 3000         | 1                          | 2                        | 20                         | 1               |
| 2           | 2250         | 4                          | 3                        | 15                         | 1               |
| 3           | 4500         | 3                          | 1                        | 30                         | 2               |

GeneXus training

porque no todos los atributos que están en la estructura de una transacción, están presentes en la tabla física.

Flight Transaction

Name

|                            |
|----------------------------|
| Flight                     |
| FlightId                   |
| FlightDepartureAirportId   |
| FlightDepartureAirportName |
| FlightDepartureCityId      |
| FlightDepartureCityName    |
| FlightArrivalAirportId     |
| FlightArrivalAirportName   |
| FlightArrivalCityId        |
| FlightArrivalCityName      |
| FlightPrice                |
| FlightDiscountPercentage   |
| FlightAirlined             |
| FlightName                 |



**DATABASE**

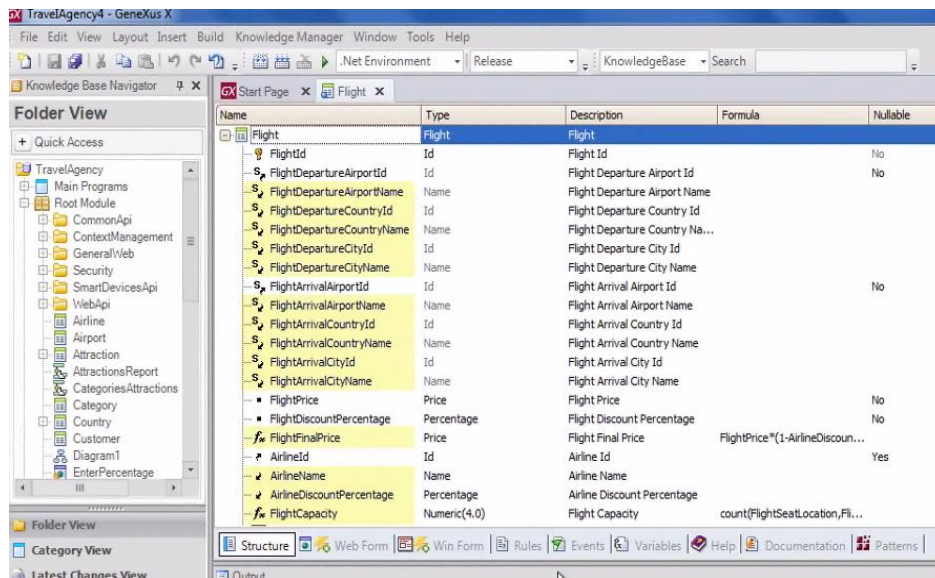
**Flight Table**

| Flight Id * | Flight Price | Flight Departure AirportId | Flight Arrival AirportId | Flight Discount Percentage | Flight Airlined |
|-------------|--------------|----------------------------|--------------------------|----------------------------|-----------------|
| 1           | 3000         | 1                          | 2                        | 20                         | 1               |
| 2           | 2250         | 4                          | 3                        | 15                         | 1               |
| 3           | 4500         | 3                          | 1                        | 30                         | 2               |

GeneXus training

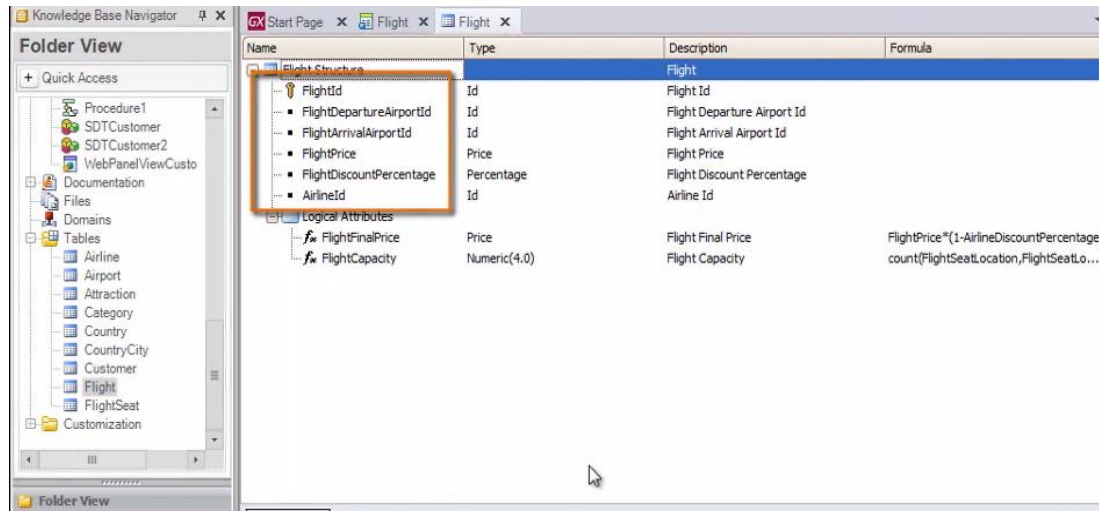
Video filmado con GeneXus X Evolution 3

Por ejemplo, si queremos insertar un vuelo, en la transacción Flight hay muchos atributos declarados en la estructura



que no están físicamente en la tabla FLIGHT sino que estan **en la tabla extendida de la tabla FLIGHT**. Los hemos incluido en la estructura para mostrarlos en el form o bien para usarlos en las reglas.

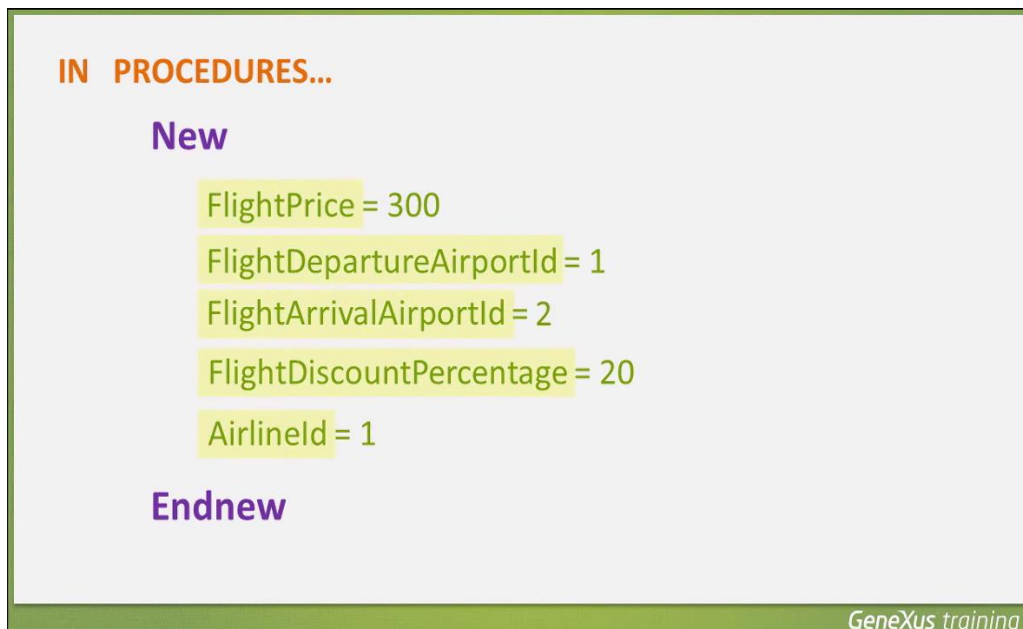
Si vamos al nodo Tables y ubicamos a la tabla FLIGHT



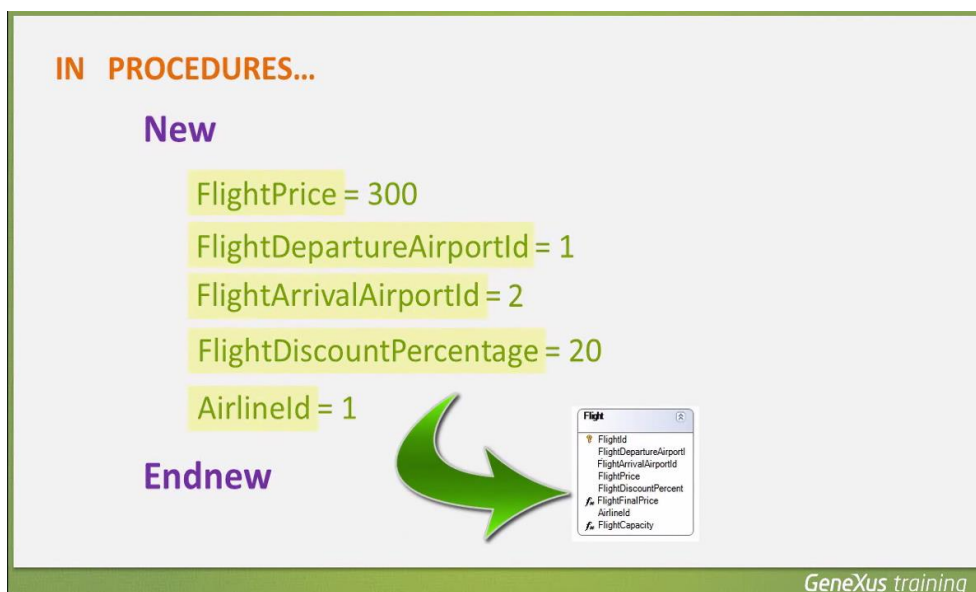
vemos estrictamente los atributos **pertenecientes a la tabla FLIGHT**.

A estos atributos, los podemos incluir dentro de un comando New, y asignarle valores

GeneXus determinará cuál es la tabla física donde insertará el registro, analizando los atributos que están a la izquierda del signo de igual



Si pertenecen todos a una misma tabla física, se realizará la inserción del registro en dicha tabla



y si no, se nos informará que no es posible determinar la tabla en la cual realizar la inserción.

La tabla que encuentra GeneXus se llama **tabla base del New**



## IN PROCEDURES...

New

FlightPrice = 300

FlightDepartureAirportId = 1

FlightArrivalAirportId = 2

FlightDiscountPercentage = 20

AirlineId = 1

Endnew



| Flight                   |
|--------------------------|
| FlightId                 |
| FlightDepartureAirportId |
| FlightArrivalAirportId   |
| FlightPrice              |
| FlightDiscountPercent    |
| FlightFinalPrice         |
| AirlineId                |
| FlightCapacity           |

New base table:  
FLIGHT

GeneXus training

Volvamos a la asignación de valores a los atributos.

Observemos que no le hemos asignado valor al atributo llave primaria FlightId.

Esto es porque FlightId tiene configurada su propiedad Autonumber con valor True, por lo que la base de datos se encargará de darle valor automática y correlativamente.

## IN PROCEDURES...

New

FlightId ?

FlightPrice = 300

FlightDepartureAirportId = 1

FlightArrivalAirportId = 2

FlightDiscountPercentage = 20

AirlineId = 1

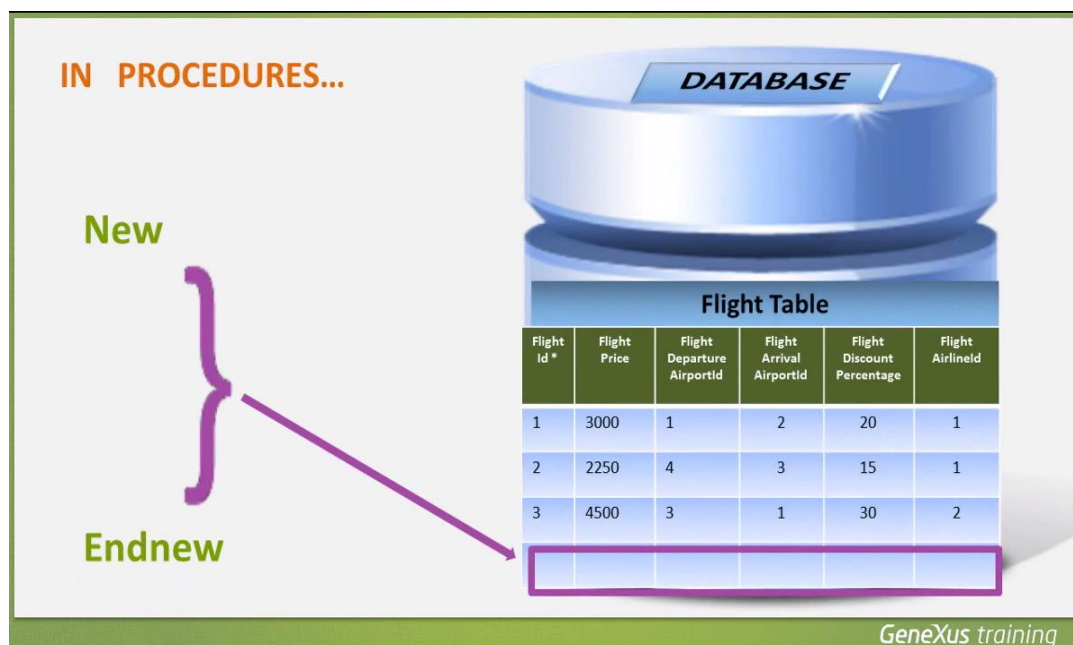
Endnew

| Properties                 |           |
|----------------------------|-----------|
| Filter                     |           |
| Attribute: FlightId        |           |
| Is Password                | False     |
| Auto Resize                | True      |
| Autonumber                 | True      |
| Autonumber for replication | True      |
| Autonumber start           | 1         |
| Autonumber step            | 1         |
| Back Color                 | Window    |
| Based on                   | Id        |
| Class                      | Attribute |
| Column title               | Flight Id |
| Contextual Title           | Id        |
| Control Type               | Edit      |
| Data Type                  | Numeric   |
| Decimals                   | 0         |
| Description                | Flight Id |
| Empty as null              | Yes       |

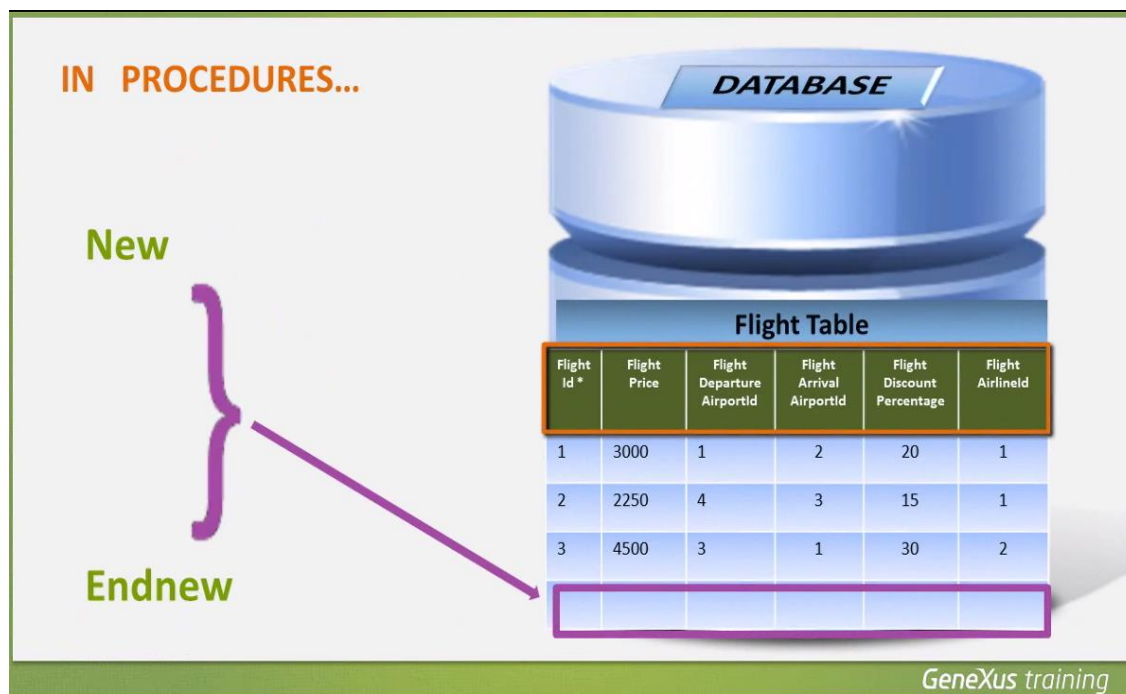
GeneXus training

Video filmado con GeneXus X Evolution 3

Dado que el comando New inserta **un** registro en **una** tabla,



solamente podremos asignarle valor, **a los atributos** que pertenezcan a esa **única tabla física**.



Es decir, **no** podremos asignarles valor a atributos que pertenezcan a distintas tablas físicas.

Sí podemos omitir asignarle un valor a algún atributo de la tabla en la cual estamos insertando, ya sea porque no es necesario (como a FlightId porque se autonumera),



## IN PROCEDURES...

New

FlightId ?

FlightPrice = 300

FlightDepartureAirportId = 1

FlightArrivalAirportId = 2

FlightDiscountPercentage = 20

AirlineId = 1

Endnew

GeneXus training

o porque queremos dejar algún atributo sin especificar.  
Por ejemplo, si omitimos asignarle valor a FlightPrice,

## IN PROCEDURES...

New

~~FlightPrice = 300~~

FlightDepartureAirportId = 1

FlightArrivalAirportId = 2

FlightDiscountPercentage = 20

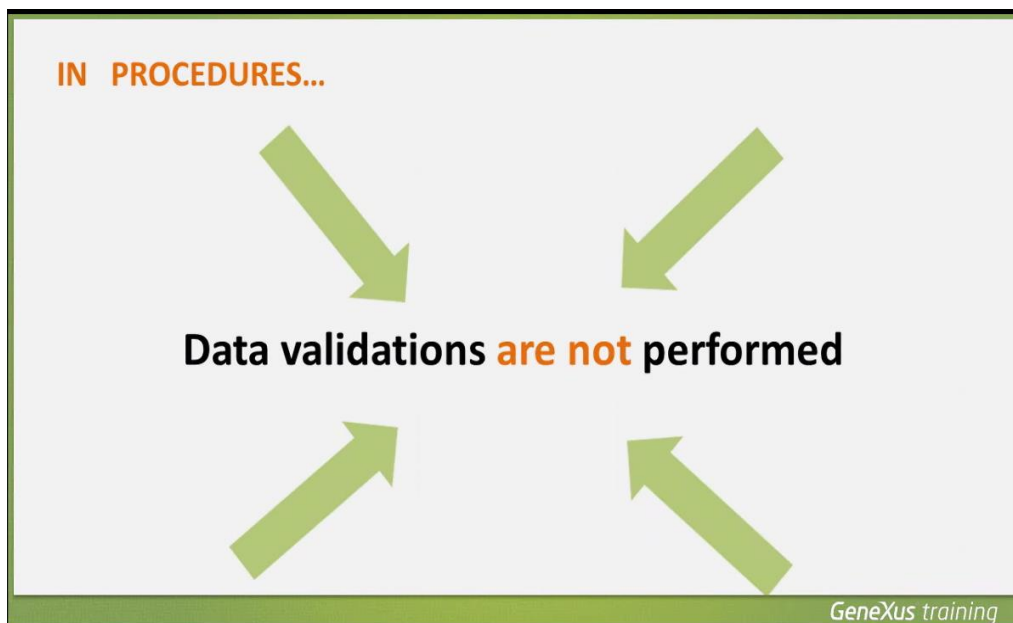
AirlineId = 1

Endnew

GeneXus training

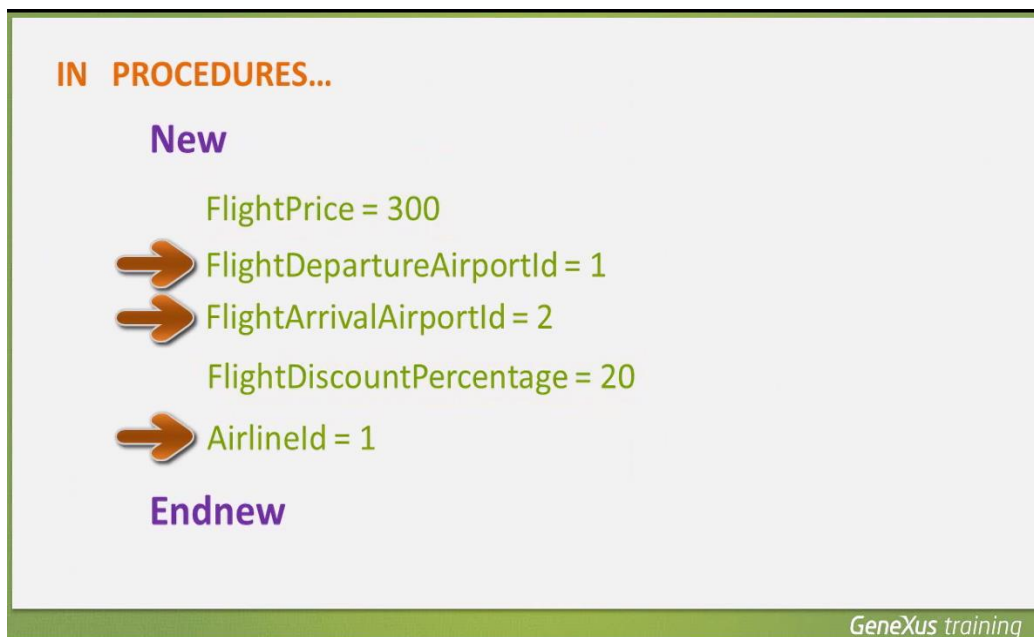
el registro insertado quedará sin precio -o dicho en otras palabras-, con precio vacío, es decir, sin especificar.

Algo importante que hay que saber, es que los procedimientos **no controlan que los datos que asignamos sean consistentes**.



Por ejemplo en el caso de estas asignaciones, podemos asignarle cualquier valor al precio del vuelo, ya que no es un dato relacionado con otras tablas.

Pero en cambio el resto de las asignaciones,



las estamos haciendo a identificadores de aeropuertos y aerolíneas... por lo que debemos ser cuidadosos de asignarle valores que existan en las tablas que almacenan los aeropuertos y las aerolíneas respectivamente.

En este ejemplo, hemos asignado el valor 1 de identificador al aeropuerto de partida

## IN PROCEDURES...

### New

FlightPrice = 300

FlightDepartureAirportId = 1

FlightArrivalAirportId = 2

FlightDiscountPercentage = 20

AirlineId = 1

### Endnew

GeneXus training

Y el identificador 2 al aeropuerto de llegada, sabiendo que tenemos en nuestra tabla de aeropuertos almacenado al aeropuerto Guarulhos con identificador: 1

## IN PROCEDURES...

### New

FlightPrice = 300

FlightDepartureAirportId = 1

FlightArrivalAirportId = 2

FlightDiscountPercentage = 20

AirlineId = 1

### Endnew

| Id  | Name              | Country Id | City Id |
|-----|-------------------|------------|---------|
| ✓ 1 | Guarulhos         | 1          | 1       |
| ✓ 2 | Charles de Gaulle | 2          | 1       |

GeneXus training

Y al aeropuerto Charles de Gaulle con identificador 2

La aerolínea con identificador 1 también la hemos registrado y corresponde a TAM.

**IN PROCEDURES...**

**New**

FlightPrice = 300  
FlightDepartureAirportId = 1  
FlightArrivalAirportId = 2  
FlightDiscountPercentage = 20  
AirlineId = 1

**Endnew**

| Id  | Name | Discount Percentage |
|-----|------|---------------------|
| ✓ 1 | TAM  | 0                   |

GeneXus training

Sin embargo, si hubiéramos asignado un valor de identificador de aeropuerto o aerolínea no registrado

el procedimiento no lo validaría, por lo que podríamos estar ingresando datos inconsistentes.

**IN PROCEDURES...**

**New**

FlightPrice = 300  
FlightDepartureAirportId = 1  
FlightArrivalAirportId = 2  
FlightDiscountPercentage = 20  
AirlineId = 9

**Endnew**

The procedure performs the New command without checking data consistency

GeneXus training

Ahora bien: como las bases de datos controlan la consistencia de los datos interrelacionados, cuando el usuario ejecuta la aplicación y se intenta asignar un valor no consistente, la base de datos rechazará la operación y la grabación inconsistente no se llevará a cabo.



Sin embargo el programa cancelará su ejecución y esto no es muy amigable para el usuario.

Por lo tanto, si usamos procedimientos para actualizar la base de datos, **será nuestra responsabilidad asignar datos válidos y bien relacionados.**

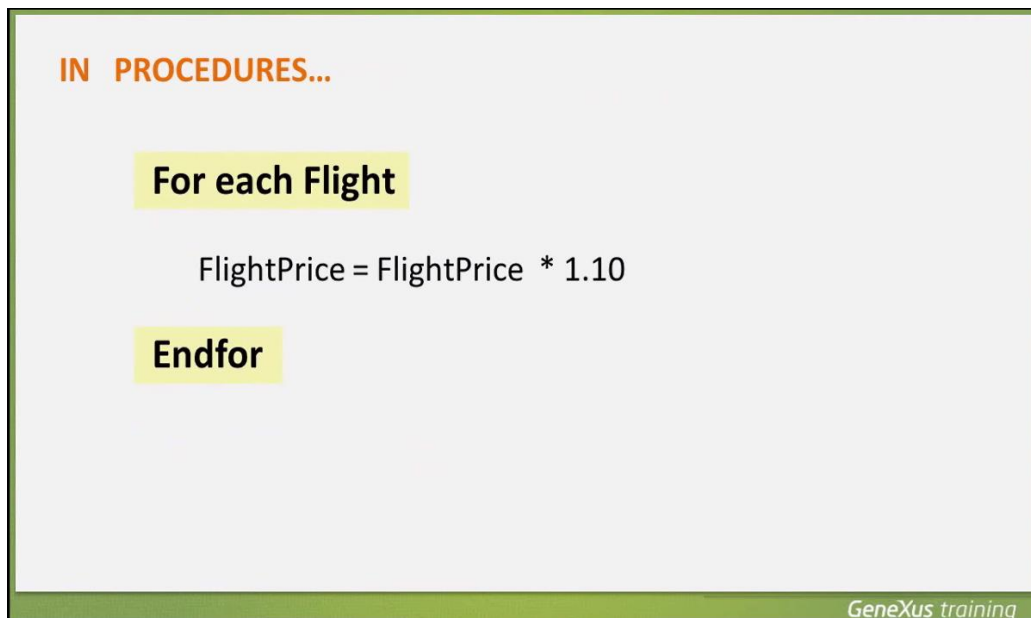


Aquí lo hemos explicado para una inserción empleando el comando New, pero lo mismo hay que tener en cuenta al actualizar datos, o eliminar registros.

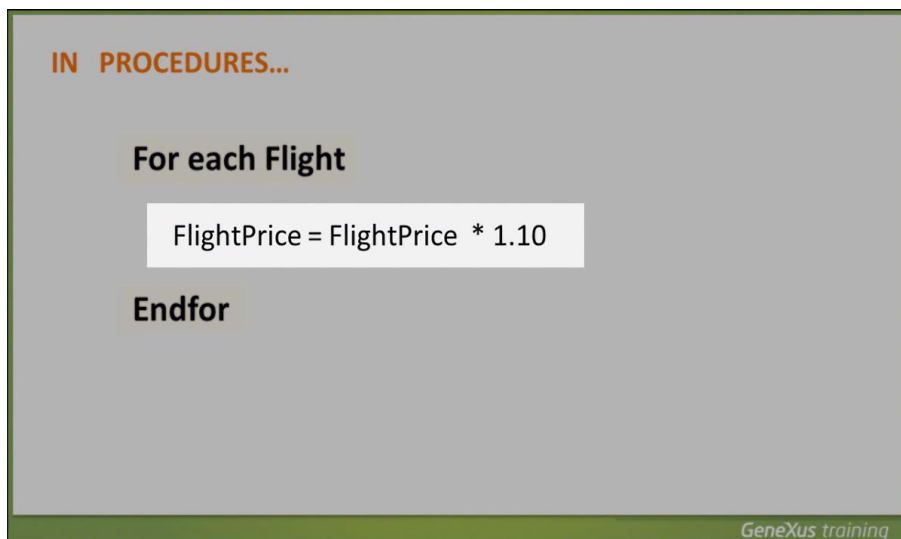


Vamos a ver ahora cómo podemos actualizar un valor existente en la base de datos.

Para cambiar un valor almacenado en un atributo, por otro valor, navegamos su tabla base mediante un For each



y mediante una asignación, damos el nuevo valor.



Podemos realizar asignaciones a los atributos de la tabla base que estamos navegando y a los de la tabla extendida.

En este ejemplo, dado que el único atributo presente dentro del For each es FlightPrice, la tabla base que GeneXus navegará es: FLIGHT

## IN PROCEDURES...

### For each Flight

FlightPrice = FlightPrice \* 1.10

### Endfor

Base Table: FLIGHT

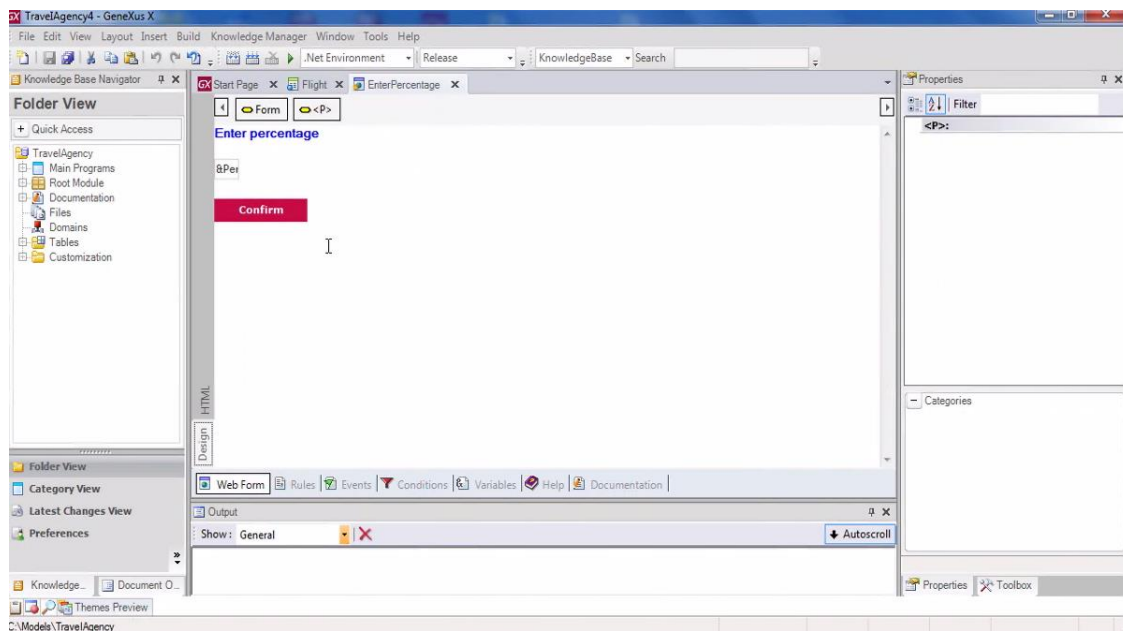
GeneXus training

Dado que no hay filtros definidos, se navegarán todos los registros de la tabla, y para cada vuelo, actualizamos su precio, en este caso, aumentándolo en un 10%.

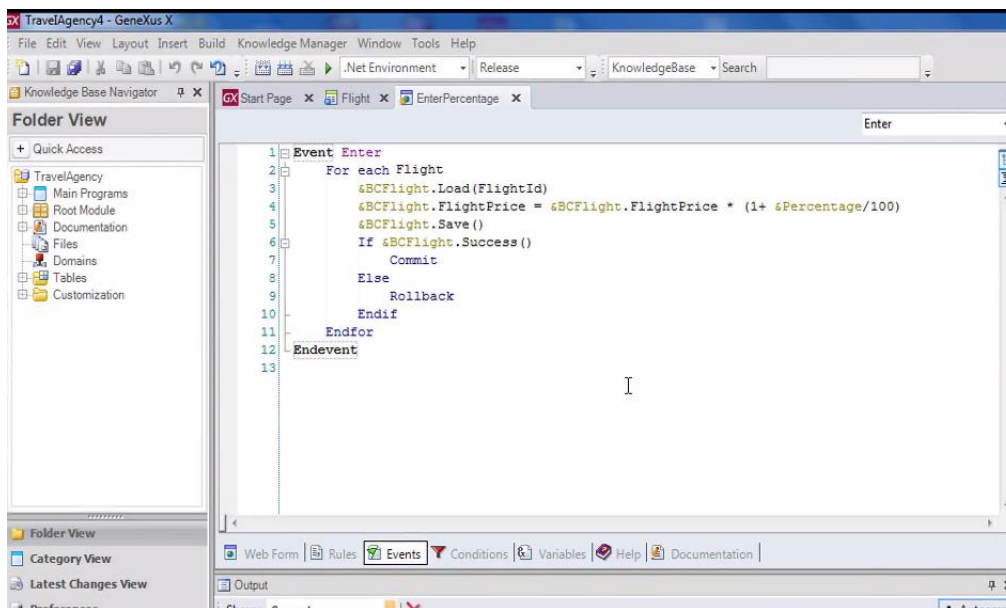
Vamos ahora a GeneXus, para poner en práctica esto que vimos.

Resolveremos la misma funcionalidad que habíamos implementado, empleando el concepto de business component y podremos comparar ambas soluciones.

Recordemos que teníamos un web panel de nombre: “EnterPercentage”

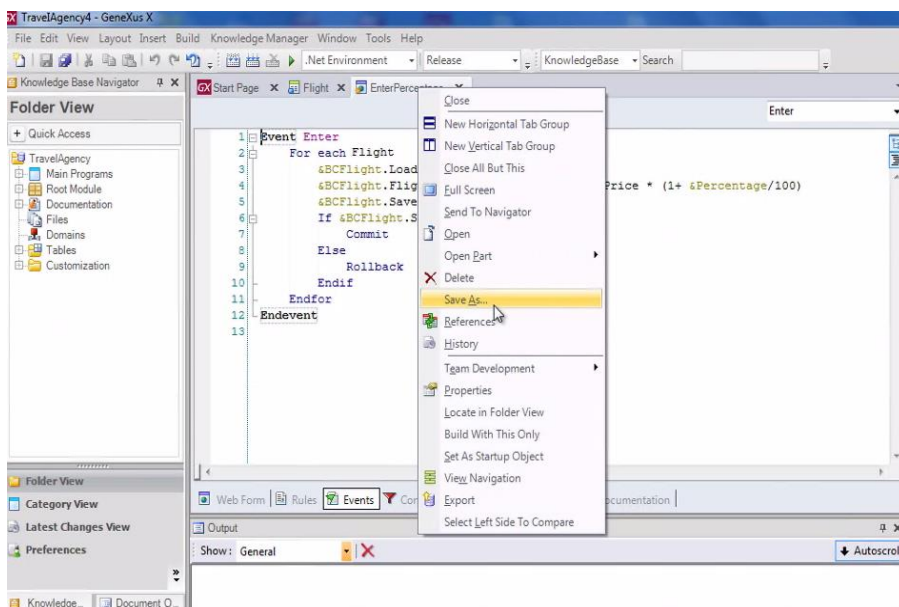


en el cual el usuario de la agencia de viajes podía digitar un porcentaje y al presionar el botón confirmar, se ejecuta **este código**



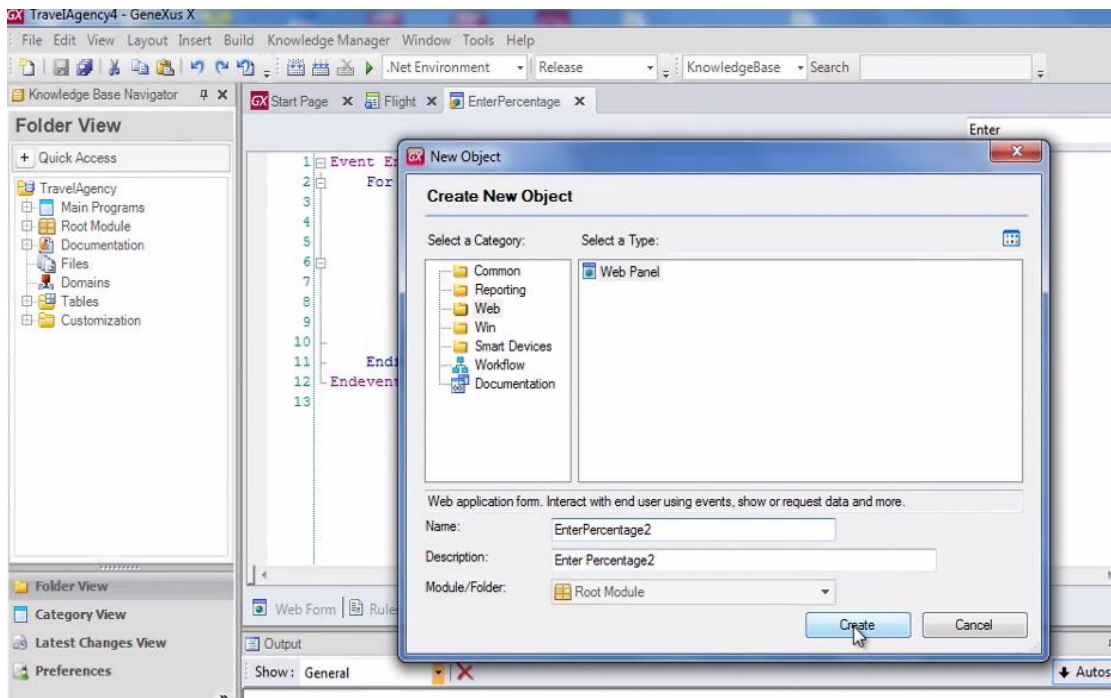
para navegar todos los vuelos e incrementar el precio de cada vuelo.

Vamos a presionar el botón derecho del mouse sobre la solapa con el nombre del objeto

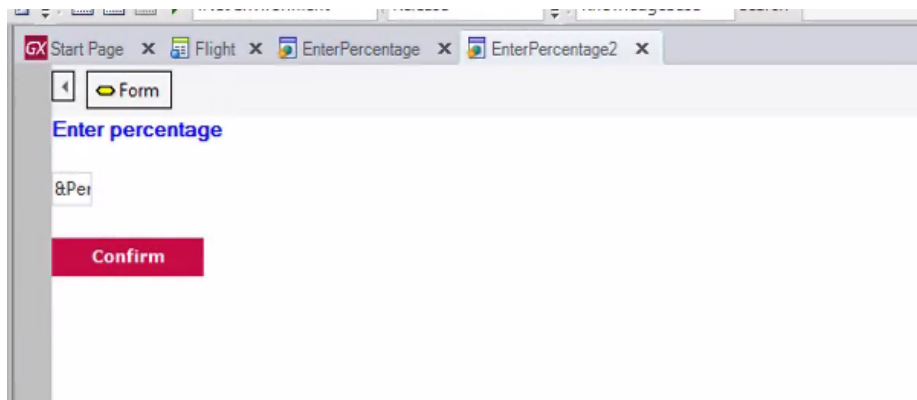


y elegimos “Save As” para obtener una copia del objeto con otro nombre.

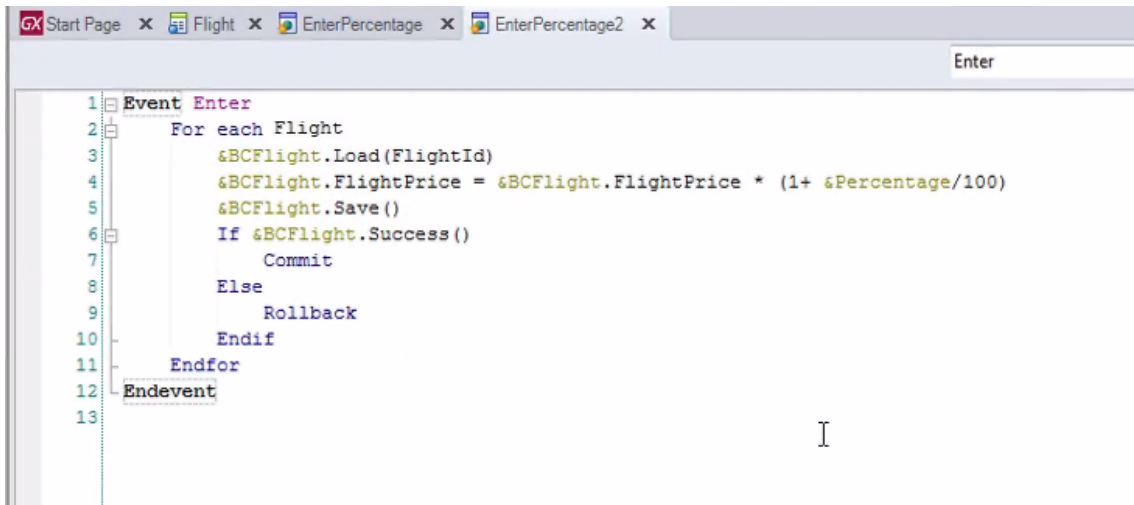
Le damos el nombre “EnterPercentage2”



Aquí nos quedó el nuevo web panel, hasta ahora idéntico al anterior



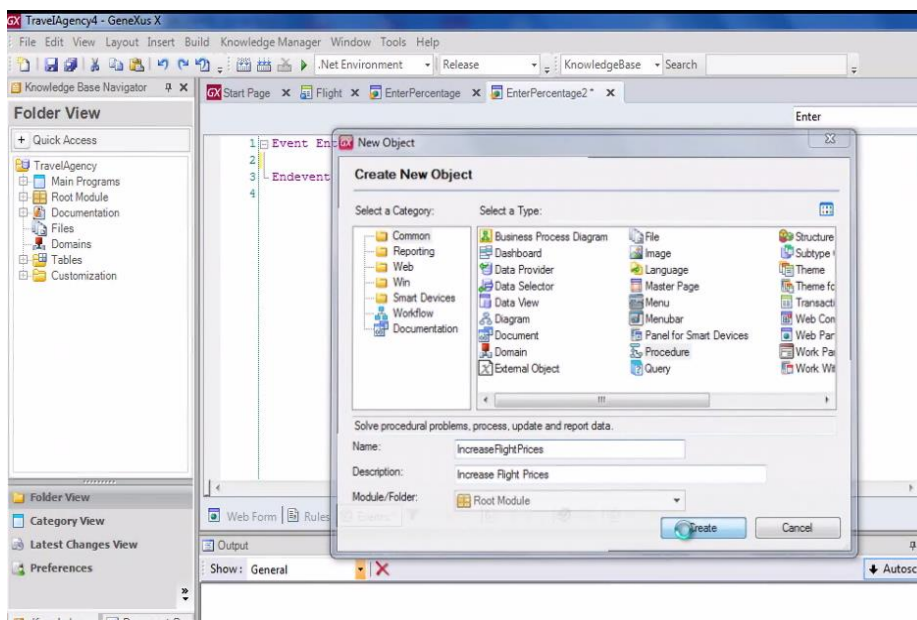
Vamos al evento asociado al botón Confirmar



Si bien en este evento hay un For each, la posibilidad de utilizar el mismo para actualizar atributos asignándole valores, solamente se permite en objetos procedimientos.

Así que vamos a borrar este código y en el evento Enter solamente incluiremos una llamada a un procedimiento, que es el que va a actualizar la base de datos.

Vamos a crear el objeto procedimiento. Le damos el nombre “IncreaseFlightPrices”



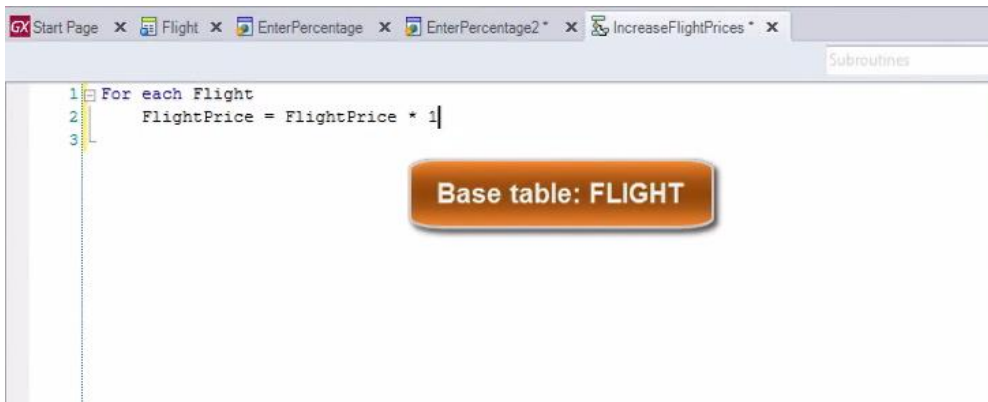
y en el source vamos a escribir el mismo código que propusimos en este ejemplo.

Escribimos For each... Flight... presionamos Control-Enter para autocompletar y elegimos FlightPrice.

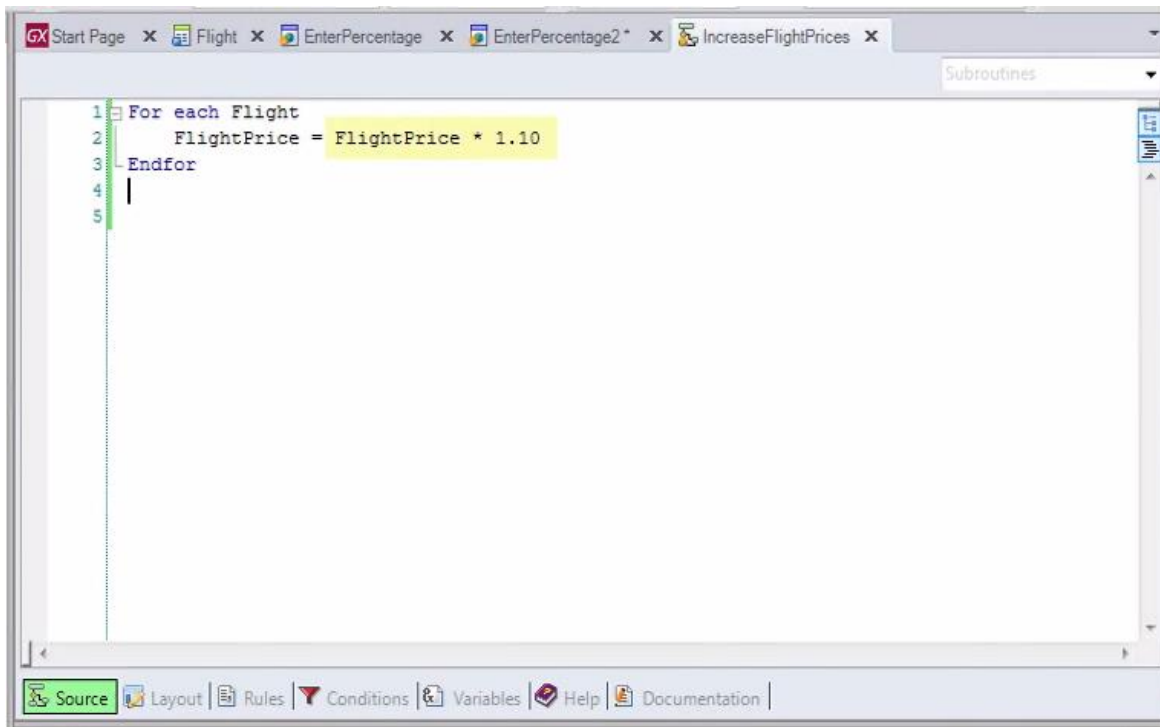
Ahora **le asignamos al atributo FlightPrice**, el valor que tenía el atributo multiplicado por 1.10.

Como ya hemos explicado, la tabla base de este For each es FLIGHT



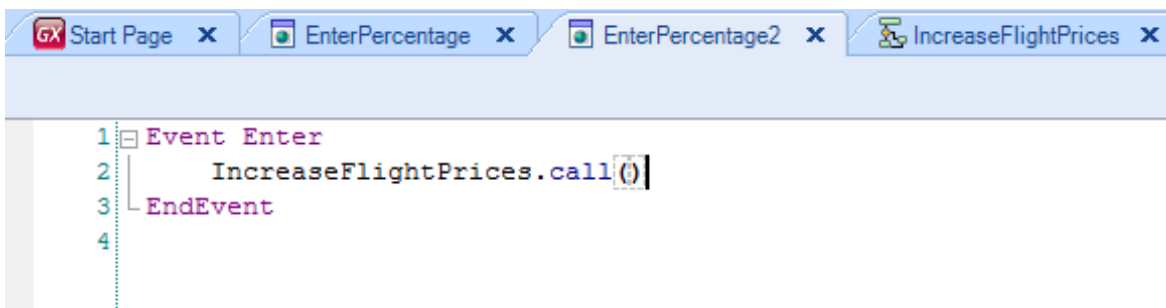


se navegan todos los vuelos y para cada uno de ellos estamos incrementando su precio en un 10%.

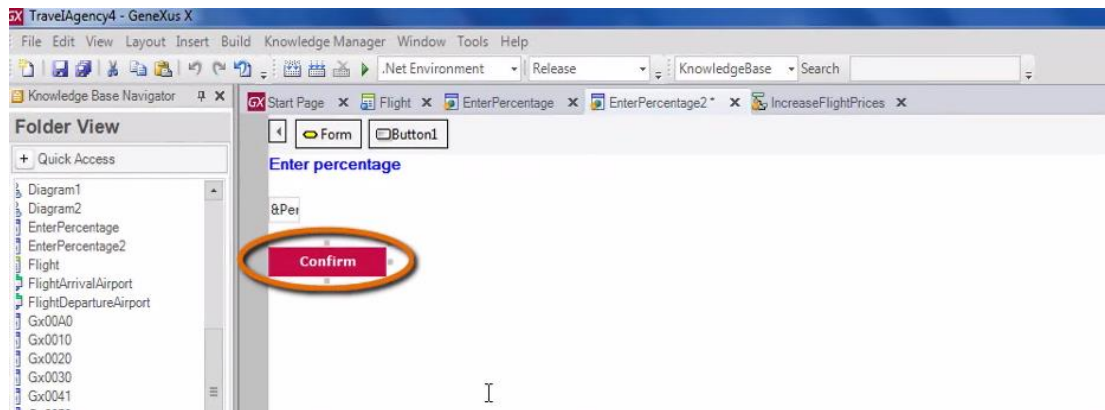


Ahora vamos a llamar a este procedimiento desde el web panel.

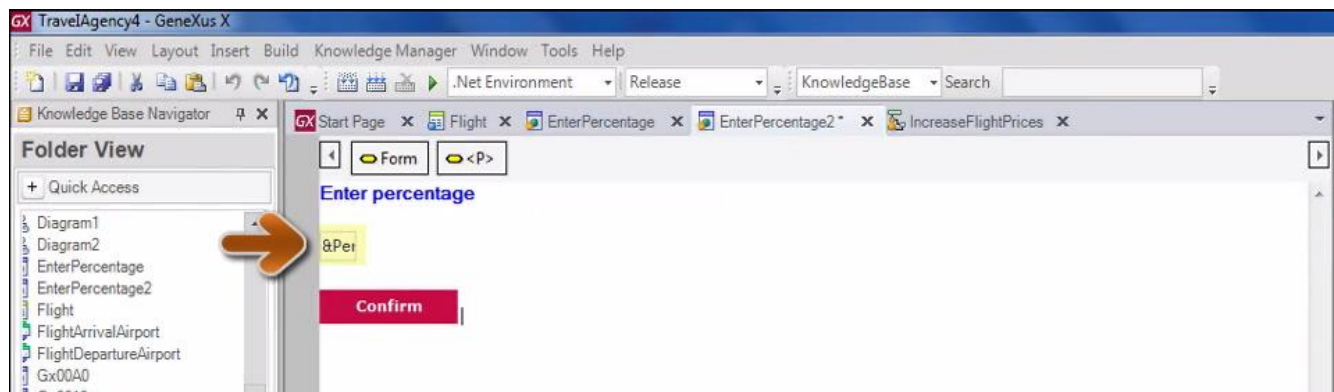
Localizamos al procedimiento en el Folder View y lo arrastramos hasta el webpanel. Digitamos punto y seleccionamos el call que se nos ofrece en este menú contextual.



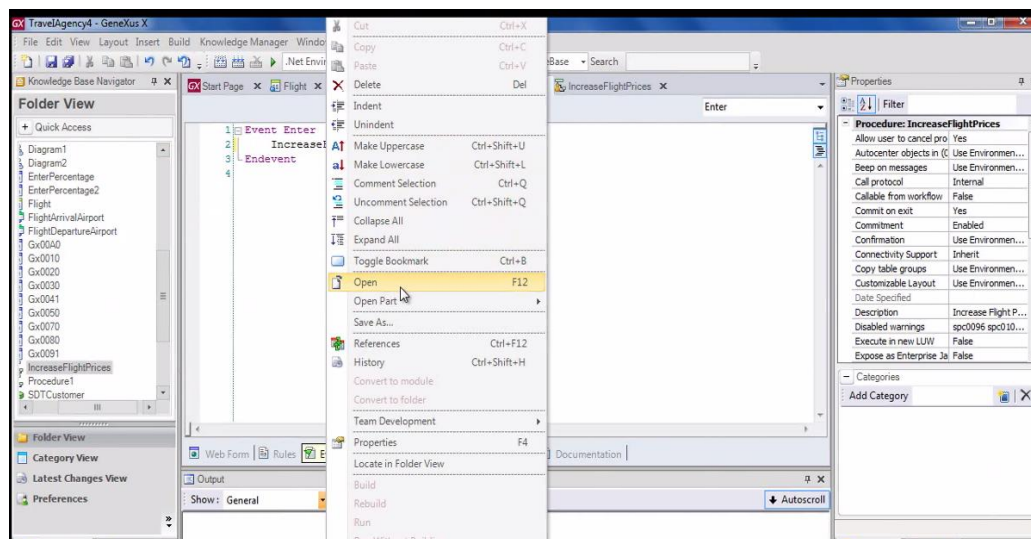
Este evento Enter se ejecutará cuando el usuario presione el botón que tenemos asociado al mismo.



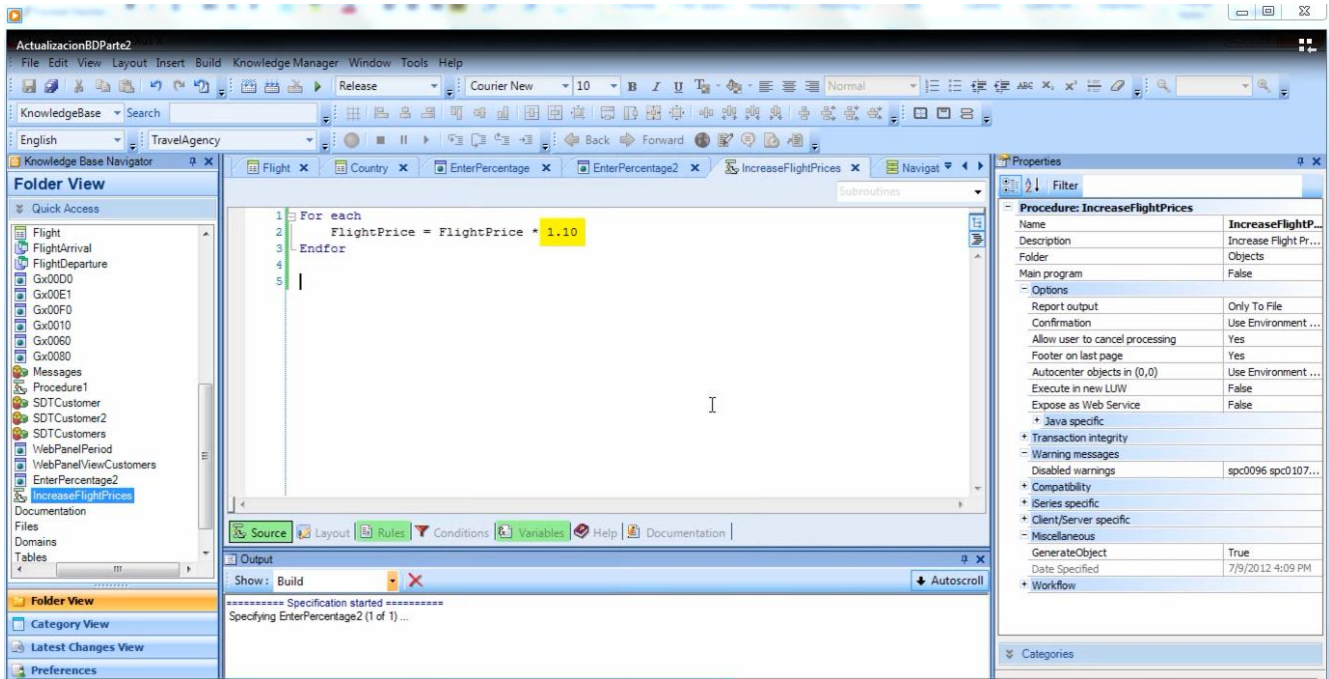
Pero antes de ejecutar esto que hicimos, observemos un pequeño detalle: En el web panel, el usuario podrá digitar en esta variable



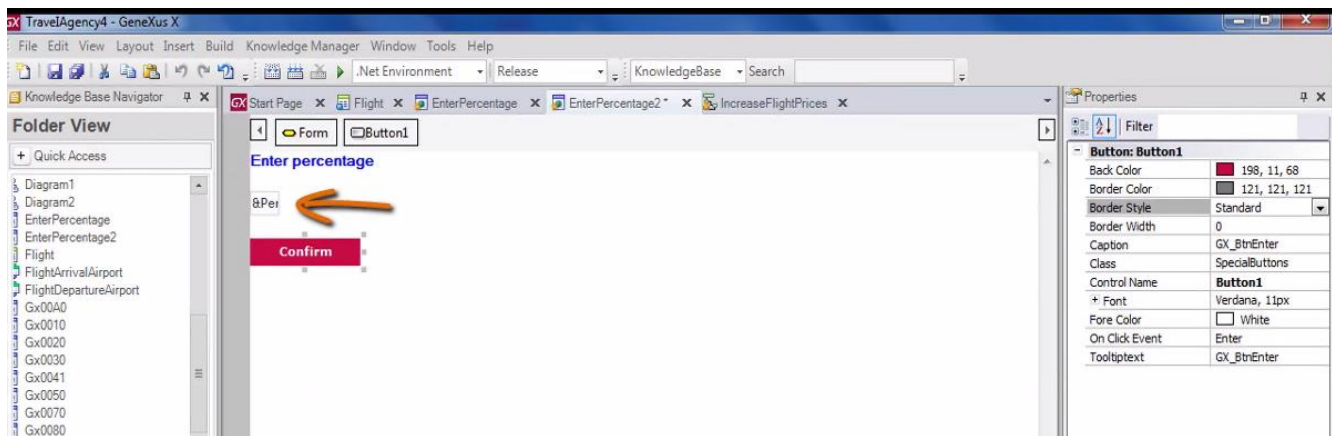
un determinado porcentaje de aumento, y al presionar el botón estamos llamando al procedimiento



que aumenta los precios en un porcentaje fijo de un 10%



El porcentaje fijo del 10% lo pusimos en una primera instancia como ejemplo... pero ahora queremos considerar el valor del porcentaje digitado por el usuario en el web panel.

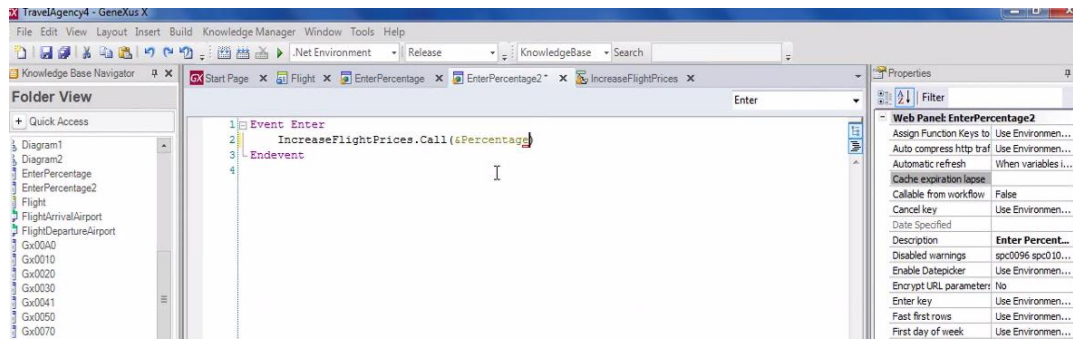


¿Cómo hacemos para que **el procedimiento** conozca el valor digitado **en el web panel**?

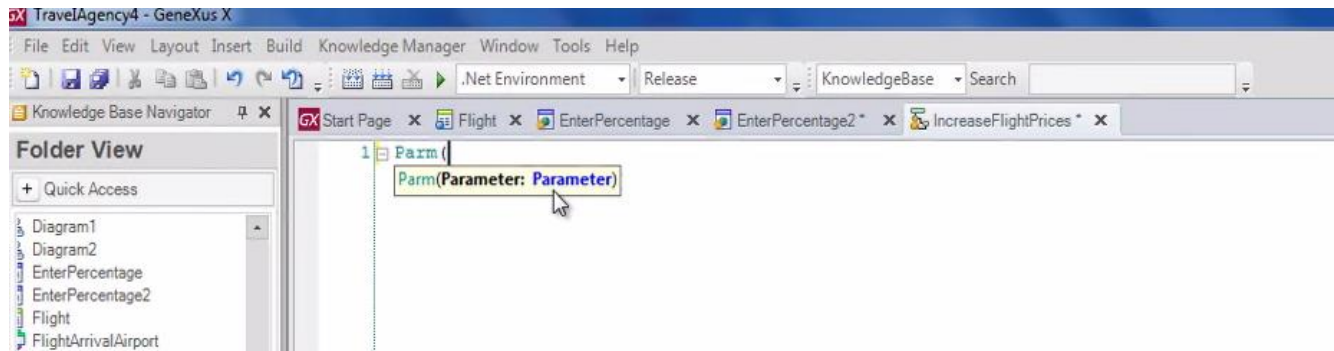
En el web panel se conoce el valor de la variable **&Percentage**, pero no en el procedimiento.

Lo que haremos es enviarle el valor guardado en la variable, al momento de llamar al procedimiento.

Para lograr esto, dentro del paréntesis del call, incluimos a la variable &Percentage



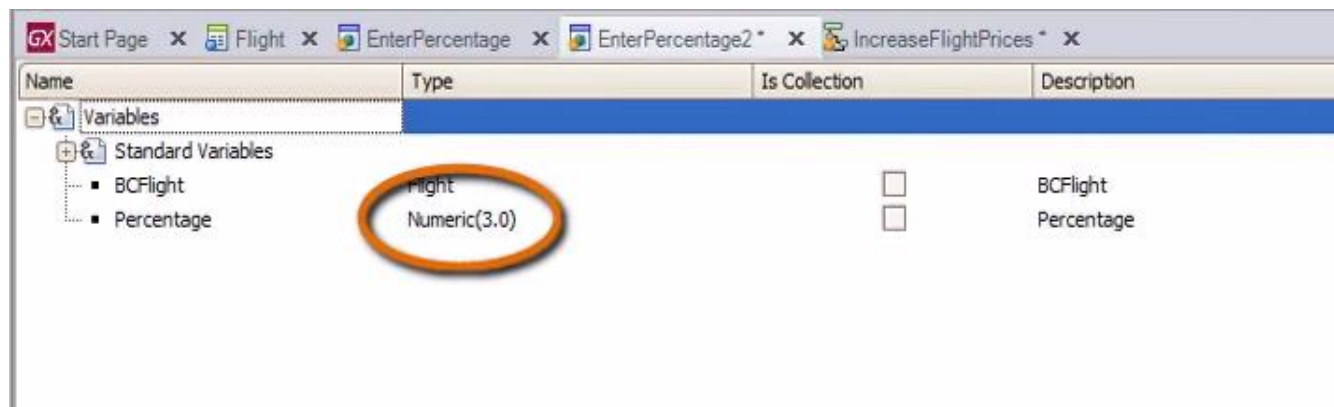
Ahora en el procedimiento, debemos recibirla. Vamos al procedimiento y en la sección rules escribimos parm seguido de paréntesis.



Dentro del paréntesis tenemos que recibir a la variable, pero la variable aquí no está definida.

Podemos definirla con el mismo nombre que está definida en el web panel o con otro nombre. Lo importante es que el tipo de datos sea idéntico al tipo de datos de la variable enviada desde el web panel al procedimiento.

Vemos que el tipo de datos de la variable &percentage en el web panel es: Numeric(3,0)



así que en el procedimiento, vamos a definirla así

| Name               | Type         | Is Collection            | Description |
|--------------------|--------------|--------------------------|-------------|
| Variables          |              |                          |             |
| Standard Variables |              |                          |             |
| Percentage         | Numeric(3.0) | <input type="checkbox"/> | Percentage  |

Ahora en la sección de rules, dentro del paréntesis de la regla Parm, incluimos a la variable que recibimos

Todas las reglas deben finalizar con punto y coma en todo objeto, así que lo ponemos y salvamos.

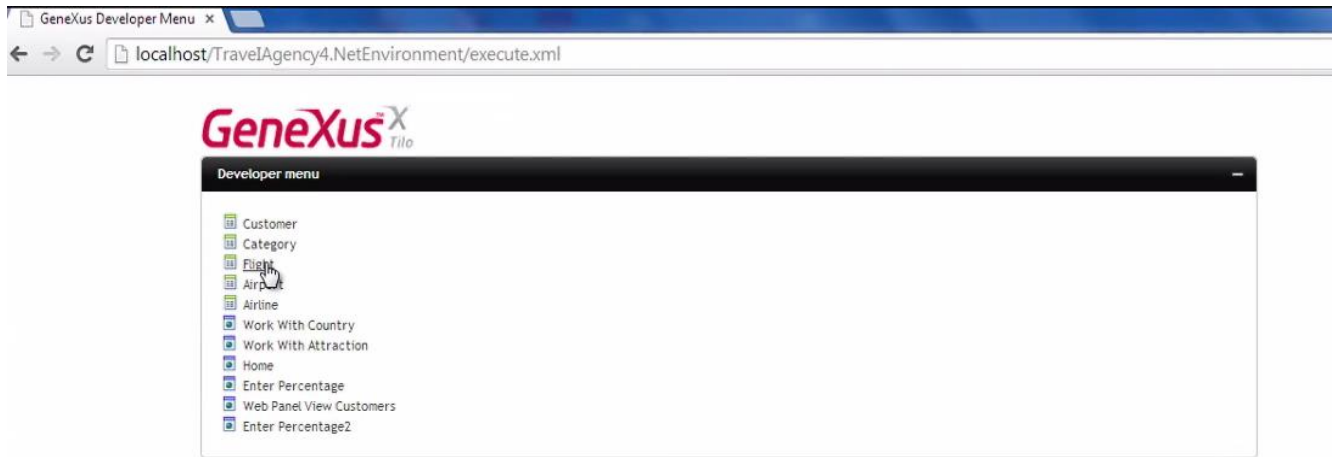
```
1 Parm(&Percentage);
```

Y ahora vayamos al source para incluir la variable &Percentage en el cálculo.

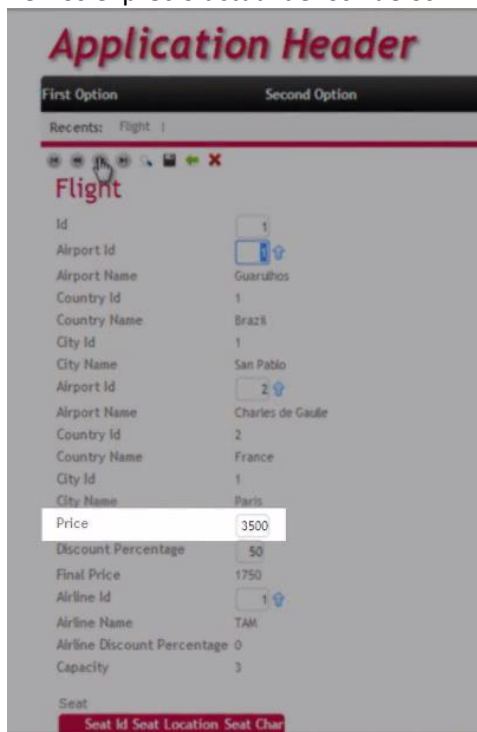
```
1 For each Flight
2   FlightPrice = FlightPrice * (1+ &Percentage/100)
3 Endfor
4
5
6 |
```

Presionemos F5...





Vemos el precio actual de los vuelos...



Ejecutamos el web panel EnterPercentage2,

# Application Header

| First Option                           | Second Option |
|--|---------------|
| Recents: Flight   Enter Percentage2    |               |
| Enter percentage                       |               |
| <input type="text" value="0"/>         |               |
| <input type="button" value="Confirm"/> |               |

Ingresamos 50% de aumento

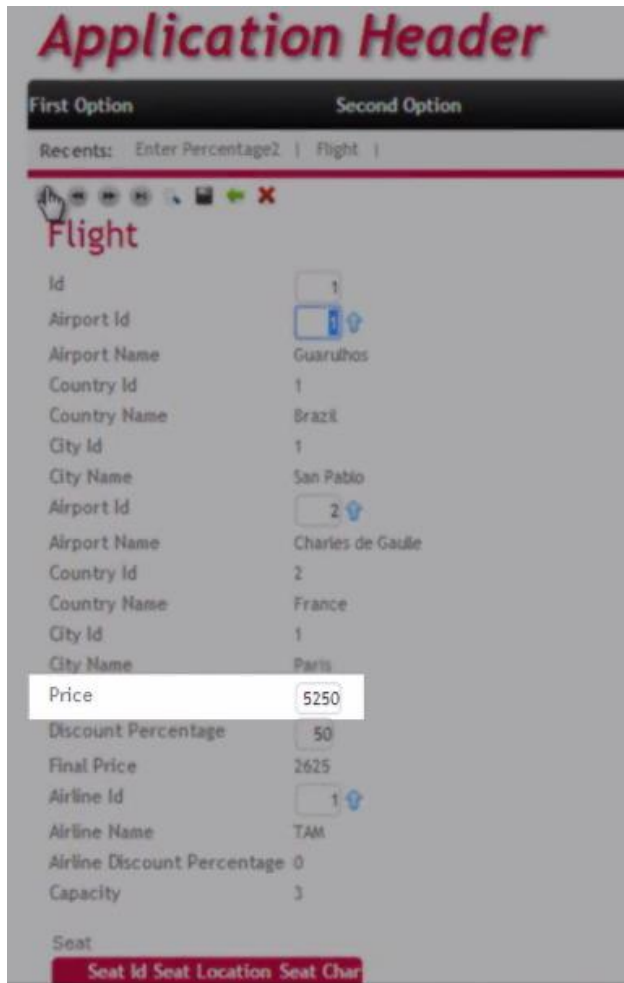
# Application Header

| First Option                           | Second Option | Third |
|--|---------------|-------|
| Recents: Flight   Enter Percentage2    |               |       |
| Enter percentage                       |               |       |
| <input type="text" value="50"/>        |               |       |
| <input type="button" value="Confirm"/> |               |       |

Footer Info

y confirmamos.

Y ahora vamos a ver los precios de los vuelos y vemos que se incrementaron en un 50%.



Algo que es importante saber es que en un mismo For each, podemos actualizar **varias tablas físicas**.

Concretamente, un For each tiene siempre una tabla base que navega y puede modificar

**IN PROCEDURES...**

**For each Flight**

FlightPrice = FlightPrice \* 1.10

**Endfor**

**Base Table: FLIGHT**

GeneXus training

pero también puede modificar toda la tabla extendida de dicha tabla base.

**IN PROCEDURES...**

**For each Flight**

$$\text{FlightPrice} = \text{FlightPrice} * 1.10$$

**Endfor**

Base Table: FLIGHT

FLIGHT extended table can be updated inside this For each.

GeneXus training

En este ejemplo, la tabla base del For each es Flight.

Como sabemos, cada vuelo tiene 1 aeropuerto de origen, 1 aeropuerto de destino, y una aerolínea que lo opera. Por lo tanto en este For each podríamos modificar los datos de dichos aeropuertos y aerolínea, si fuera necesario.

**IN PROCEDURES...**

**For each Flight**

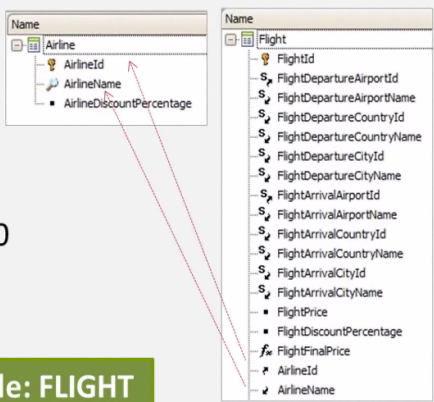
$$\text{FlightPrice} = \text{FlightPrice} * 1.10$$

$$\text{AirlineName} = \text{'...'}'$$

**Endfor**

Base Table: FLIGHT

AIRLINE is included in FLIGHT extended table.

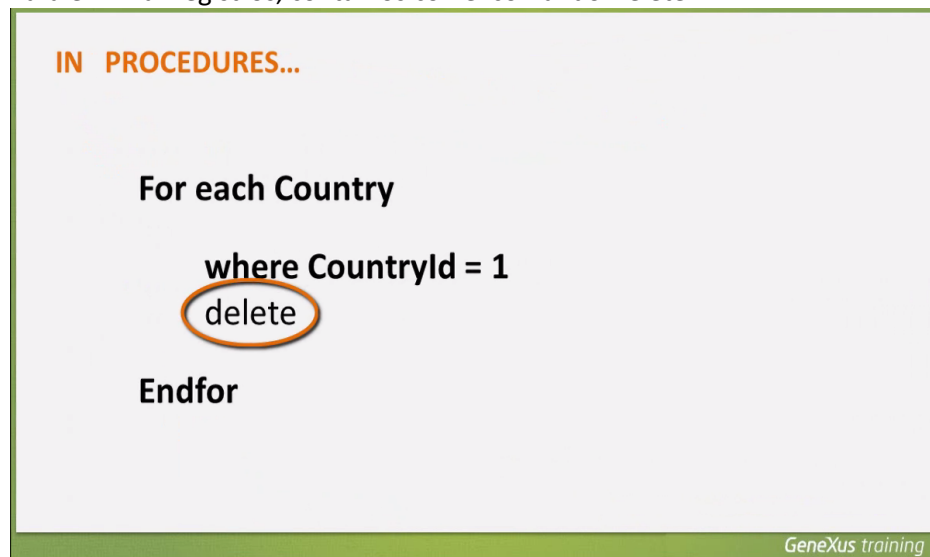


GeneXus training

Por ejemplo, podríamos modificar dentro de este For each, para cada vuelo, los datos de la aerolínea asociada al vuelo.

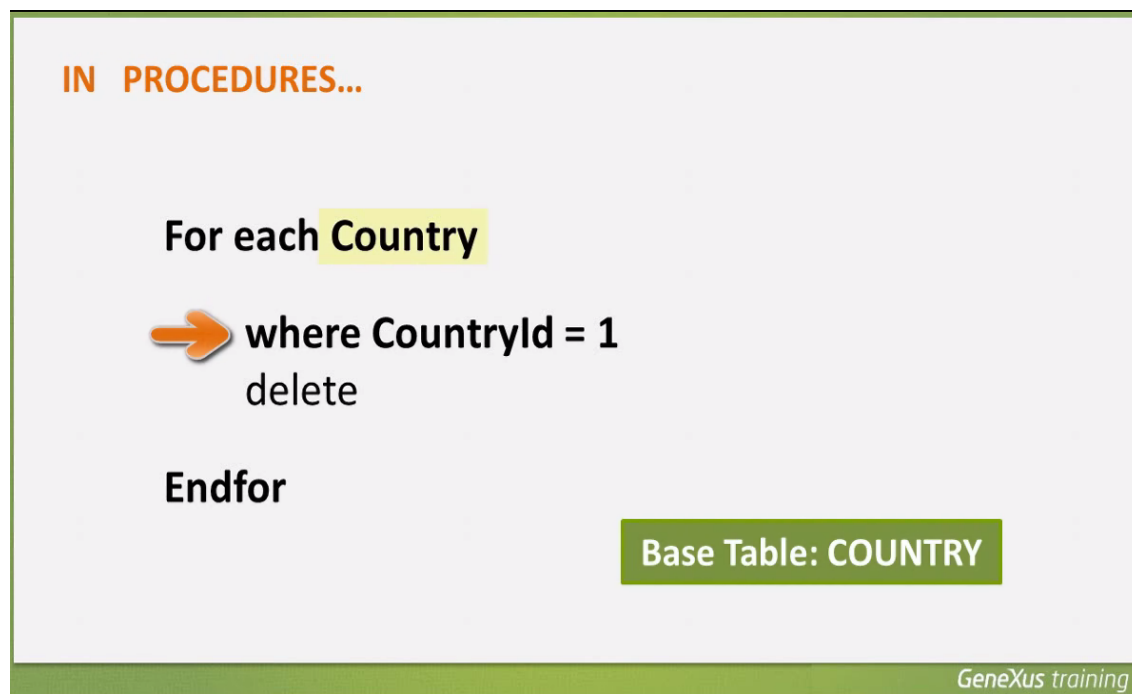
Ahora bien. Ya hemos visto cómo los procedimientos nos permiten insertar y actualizar registros en la base de datos. Vamos a ver ahora cómo eliminar registros.

Para eliminar registros, contamos con el comando Delete



que se usa dentro de un comando For each.

Básicamente hay que navegar la tabla en la cual queremos eliminar uno o varios registros, e incluir dentro del For each el comando delete.



En el ejemplo vemos que estamos navegando la tabla Country, filtrando por el valor de un determinado país y por lo tanto eliminando puntualmente 1 registro, con el comando Delete. Pero también podríamos haber

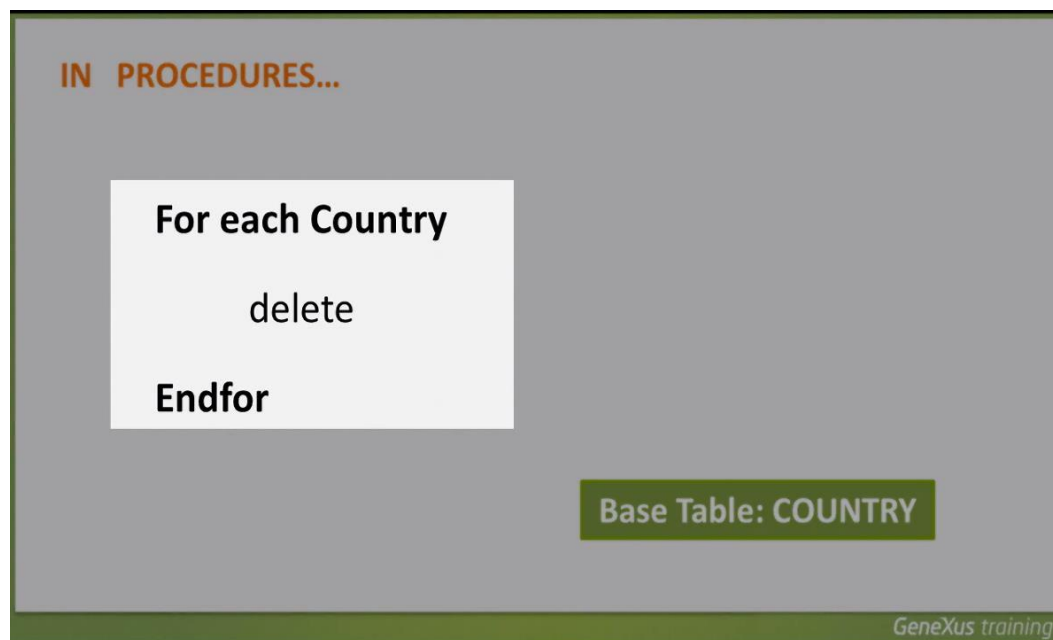
*Video filmado con GeneXus X Evolution 3*



eliminado a todos los países ingresados en la tabla Country.

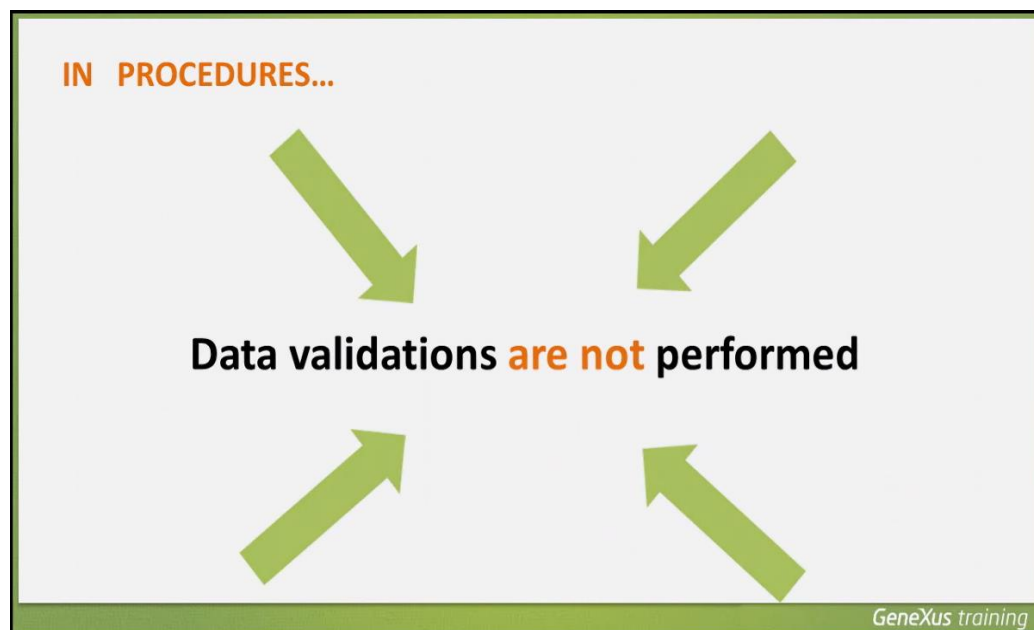
Como ya hemos explicado, los procedimientos no tienen en cuenta los datos relacionados en otras tablas.

Así, que este For each que elimina todos los países



al ejecutarse, podría dejar atracciones turísticas almacenadas referenciando países que han sido borrados.

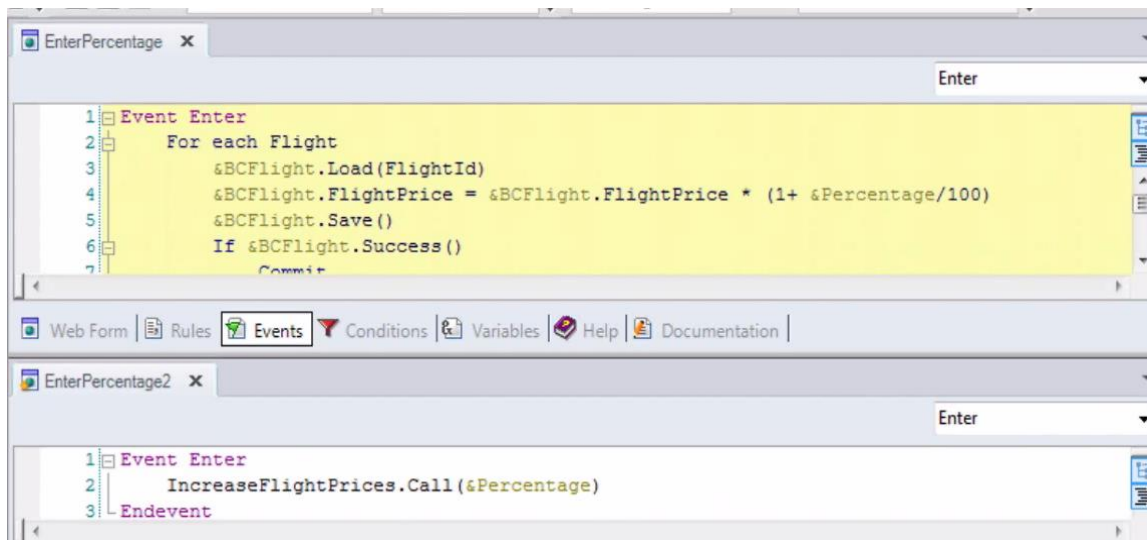
Como la base de datos controla la consistencia de los datos interrelacionados, rechazará la operación y el programa cancelará su ejecución.



Por lo tanto, es nuestra responsabilidad al usar procedimientos, borrar, asignar o insertar datos que sean coherentes con el resto de la información almacenada.

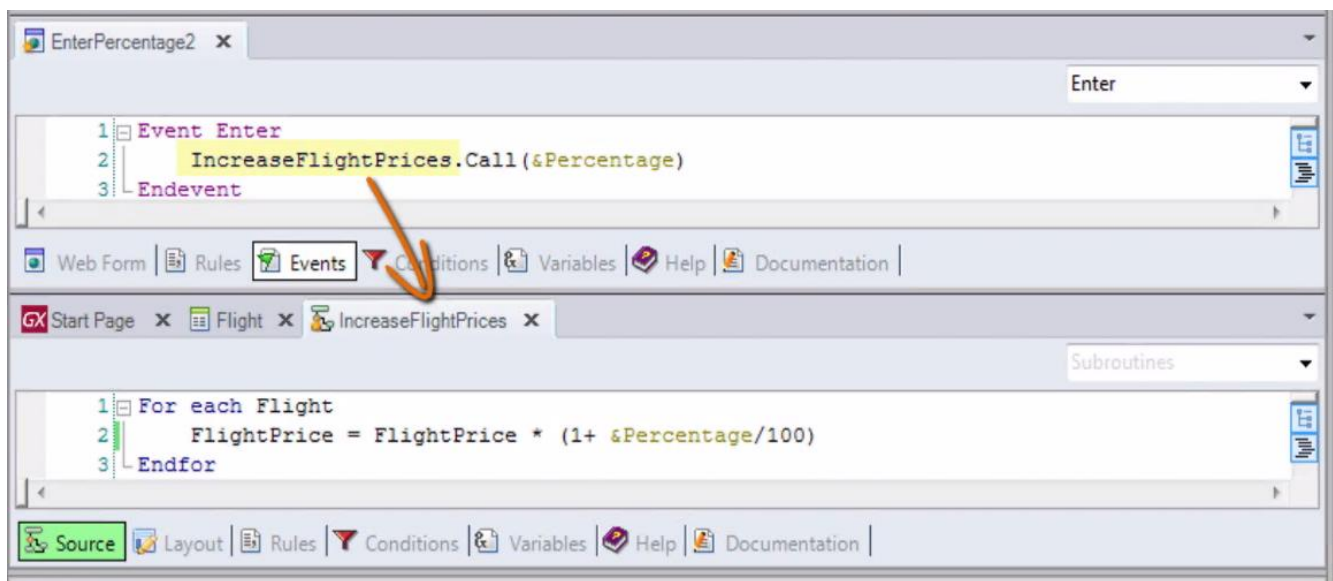
Para finalizar, comparemos las 2 alternativas que utilizamos para actualizar la base de datos.

En la primera alternativa que implementamos,

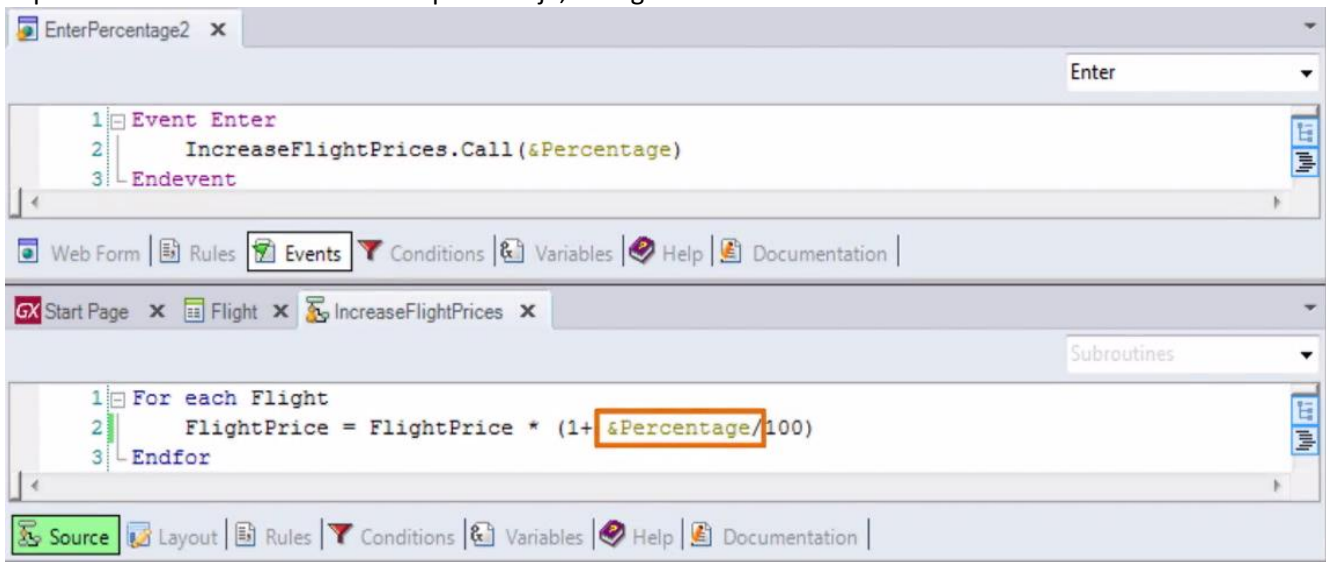


en el evento enter del webpanel EnterPercentage, escribimos el For each y actualizamos la base de datos empleando a la transacción Flight como Business Component.

En la 2da solución, el evento enter del webpanel EnterPercentage2, solamente contiene una llamada a un procedimiento

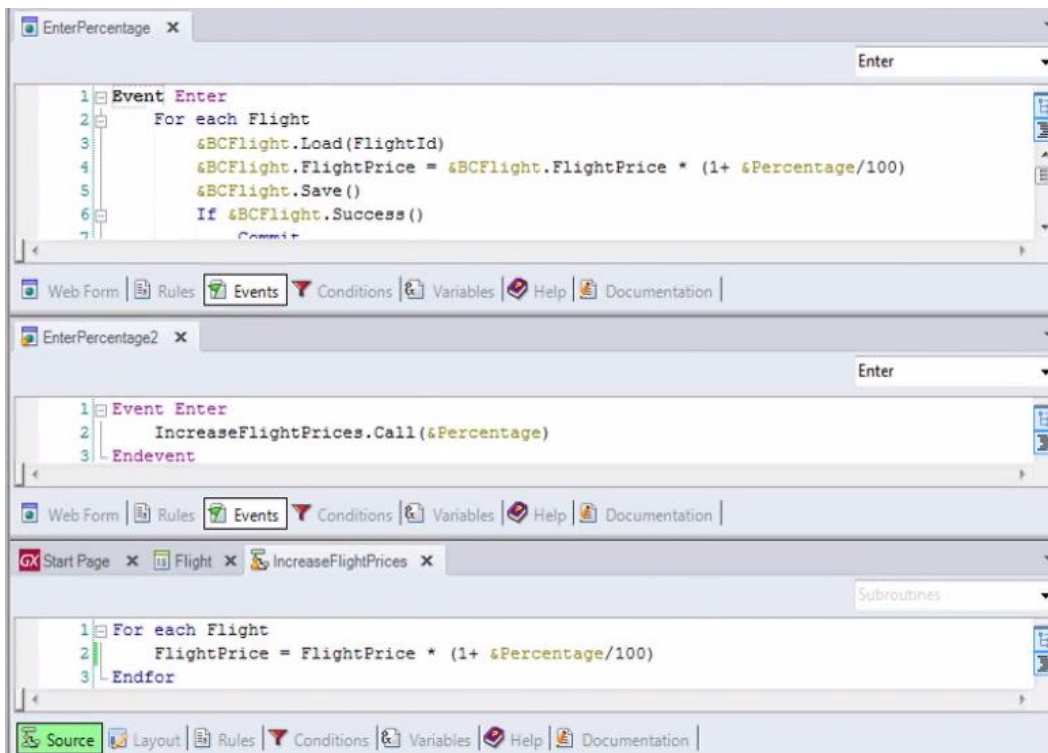


el procedimiento recibe el valor del porcentaje, navega con For each



todos los vuelos y a cada uno de ellos, le calcula y asigna el nuevo precio

**¿Cuáles son las diferencias, ventajas y desventajas entre resolverlo de una manera u otra?**



Algo que hemos visto y es oportuno repasar, es que si bien el comando For each puede ser usado en un web panel, no es posible utilizarlo para modificar la base de datos directamente asignándole valores a los atributos, y que esto solamente puede llevarse a cabo desde un objeto procedimiento. Tampoco es posible codificar un

comando New en un web panel, ni incluir un comando delete dentro del For each. **Esto sólo es válido en procedimientos.**

**Sí en cambio es posible, en cualquier objeto,** modificar la base de datos con business components.

Además, cuando empleamos business components

### USING BUSINESS COMPONENTS:

- ✓ Data validations are performed

GeneXus training

se valida la consistencia de los datos que se intentan actualizar en la base de datos

### USING BUSINESS COMPONENTS:

- ✓ Data validations are performed
- ✓ Transaction rules are triggered

GeneXus training

y se ejecutan las reglas definidas en la transacción que se está ejecutando como business component.

Las reglas que generan mensajes como msg y error, también se disparan y los mensajes correspondientes quedan guardados en una colección que se puede recorrer e imprimir.

En un procedimiento, ninguna de estas cosas se realiza.

De esta manera, hemos visto distintas alternativas para actualizar la base de datos, en una aplicación GeneXus.