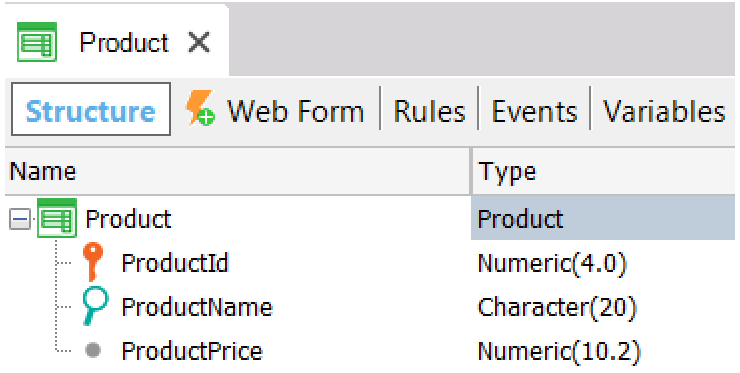
GeneXus Course

Conceptual Summary

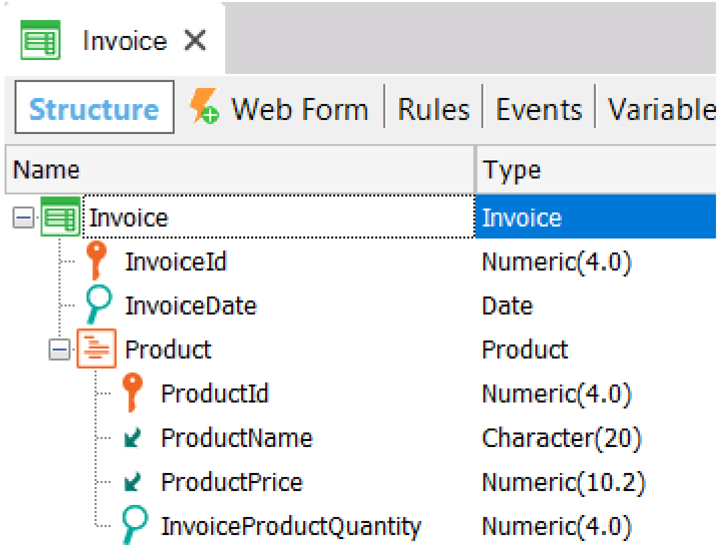
GeneXus™ 16

Transactions

What attribute names do we use?



Name	Type
Product	Product
ProductId	Numeric(4.0)
ProductName	Character(20)
ProductPrice	Numeric(10.2)



Name	Type
Invoice	Invoice
InvoiceId	Numeric(4.0)
InvoiceDate	Date
Product	Product
ProductId	Numeric(4.0)
ProductName	Character(20)
ProductPrice	Numeric(10.2)
InvoiceProductQuantity	Numeric(4.0)

GIK Naming convention

Entity + Category [+ Qualifier]

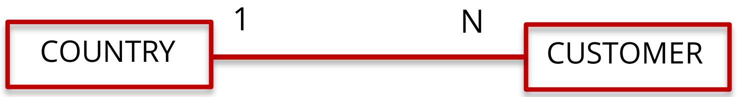
```
Product
{
    ProductId*  (PK)
    ProductName (S)
    ProductPrice (S)
}
```

```
Invoice
{
    InvoiceId* (PK)
    InvoiceDate (S)
    -----
    Product
    {
        ProductId* (PK, FK)
        ProductName  (I)
        ProductPrice  (I)
        InvoiceProductQuantity  (S)
        --------
    }
}
```

Transaction Design

Strong 1 – N

Each customer belongs to a country and a country has many customers



COUNTRY
CountryId*
CountryName

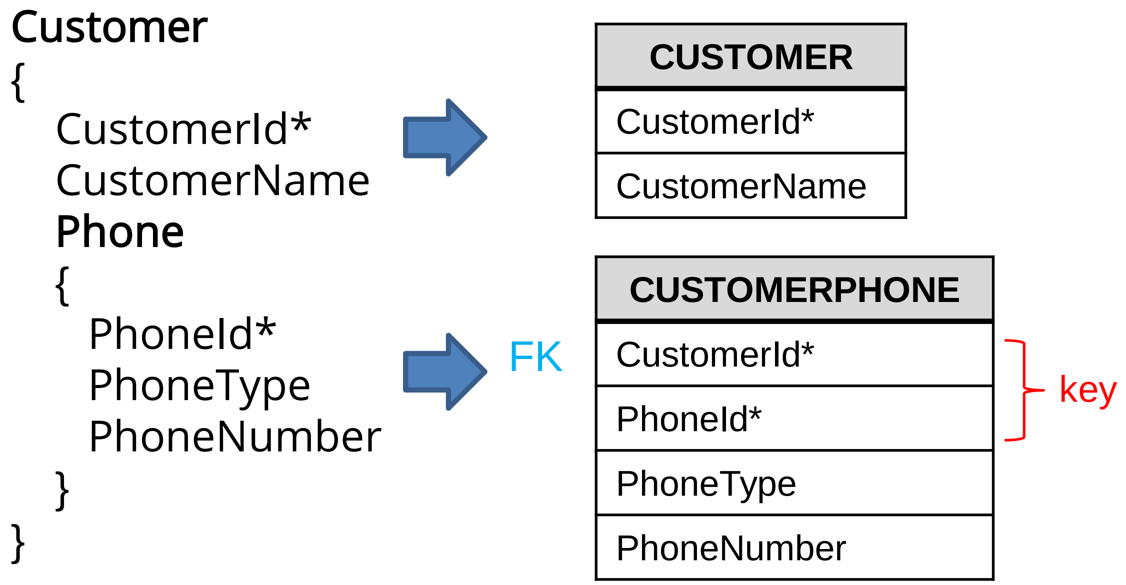
```
Country
{
   CountryId*   (PK)
   CountryName
}
```

```
Customer
{
   CustomerId*
   CustomerName
   CountryId   (FK)
   CountryName
}
```

CUSTOMER
CustomerId*
CustomerName
CountryId

Weak 1 – N

Each customer has many phones, and each phone belongs to a single customer



```
Customer
{
    CustomerId*
    CustomerName
    Phone
    {
       PhoneId*
       PhoneType
       PhoneNumber
    }
}
```

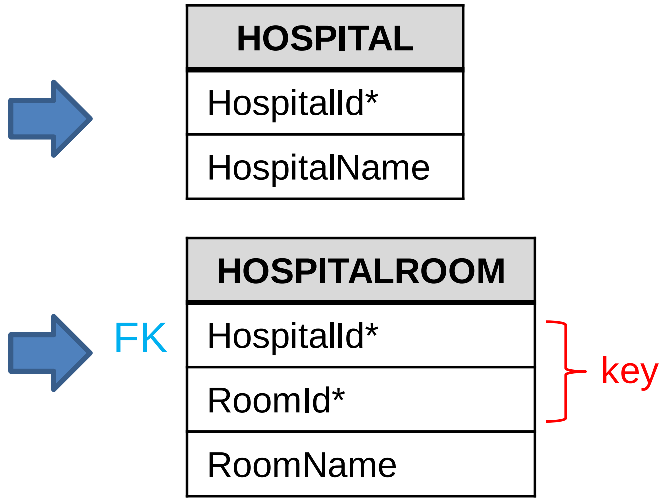
CUSTOMER
CustomerId*
CustomerName

CUSTOMERPHONE
CustomerId*
PhoneId*
PhoneType
PhoneNumber

Weak 1 – N

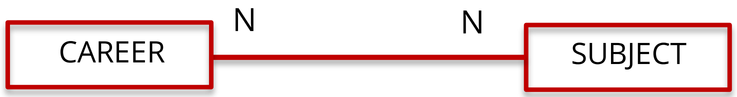
Each hospital has many rooms and each room belongs to a single hospital

```
Hospital
{
   HospitalId*
   HospitalName
   Room
   {
      RoomId*
      RoomName
   }
}
```



N – N (M)

Each degree program has many subjects and each subject can be included in many degree programs



N – N: Option 1 of 4

Each degree program has many subjects and each subject can be included in many degree programs

CAREER
CareerId*
CareerName

```
Career
{
    CareerId*
    CareerName
}
```

```
Subject
{
    SubjectId*
    SubjectName
}
```

SUBJECT
SubjectId*
SubjectName

CAREERSUBJ ECT
CareerId*
SubjectId*

```
CareerSubject
{
    CareerId*
    SubjectId*
}
```

N – N: Option 1 – Generated tables

CAREER

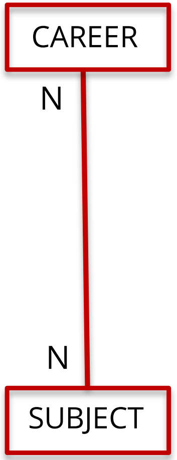
CareerId	CareerName
1	Computer Science
2	Data Science for Health

SUBJECT

SubjectId	SubjectName
1	Computer Logic
2	Programming Fundamentals

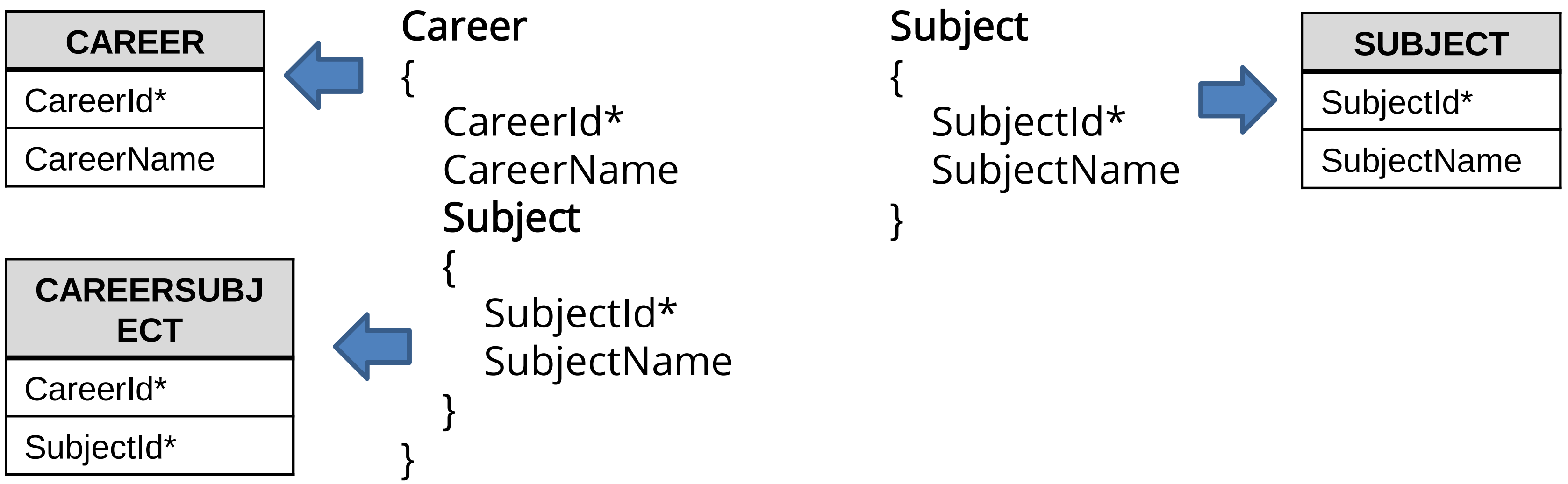
CAREERSUBJECT

CareerId	SubjectId
1	1
1	2
2	2



N – N: Option 2 of 4

Each degree program has many subjects and each subject can be included in many degree programs



CAREER
CareerId*
CareerName

CAREERSUBJECT
CareerId*
SubjectId*

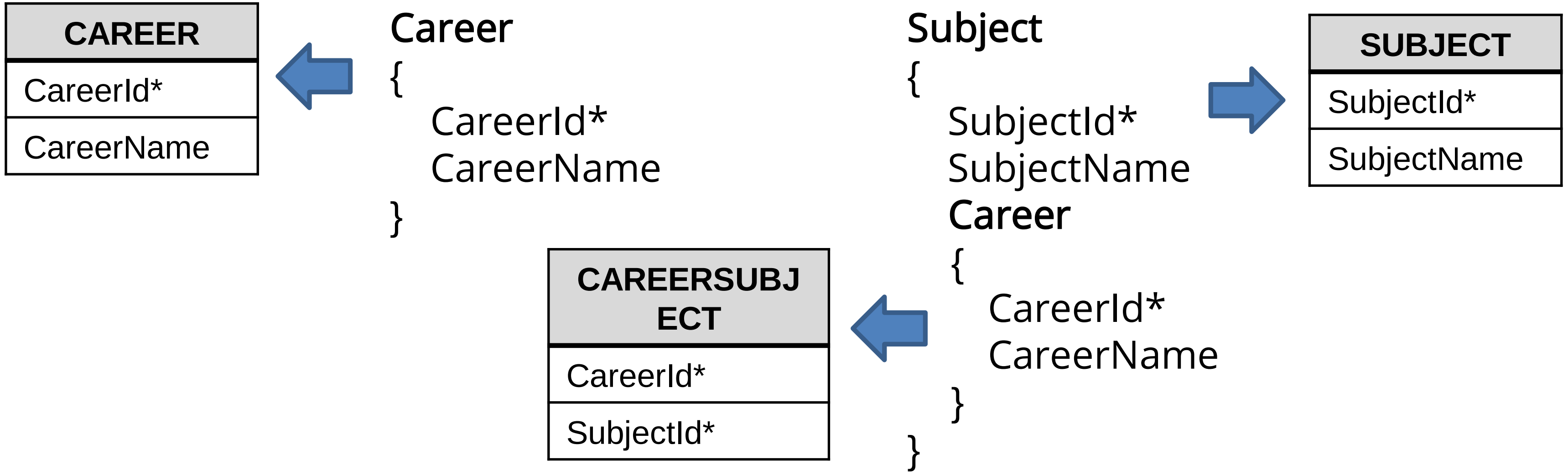
```
Career
{
    CareerId*
    CareerName
    Subject
    {
        SubjectId*
        SubjectName
    }
}
```

```
Subject
{
    SubjectId*
    SubjectName
}
```

SUBJECT
SubjectId*
SubjectName

N – N: Option 3 of 4

Each degree program has many subjects and each subject can be included in many degree programs



CAREER
CareerId*
CareerName

```
Career
{
    CareerId*
    CareerName
}
```

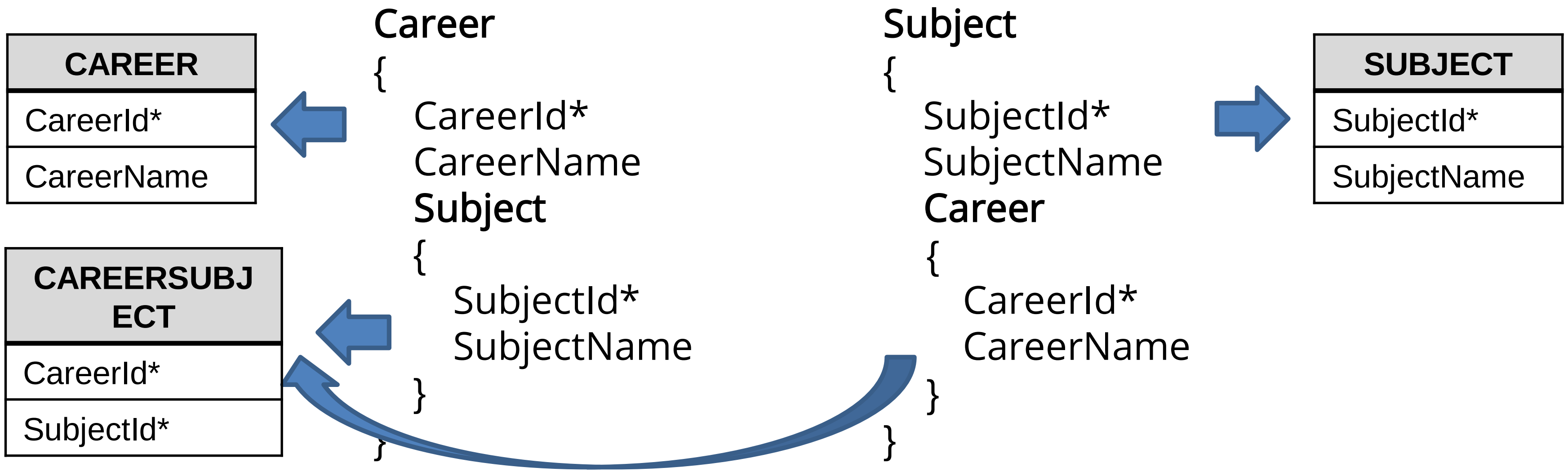
```
Subject
{
    SubjectId*
    SubjectName
    Career
    {
        CareerId*
        CareerName
    }
}
```

SUBJECT
SubjectId*
SubjectName

CAREERSUBJECT
CareerId*
SubjectId*

N – N: Option 4 of 4

Each degree program has many subjects and each subject can be included in many degree programs



```
Career
{
    CareerId*
    CareerName
    Subject
    {
        SubjectId*
        SubjectName
    }
}
```

```
Subject
{
    SubjectId*
    SubjectName
    Career
    {
        CareerId*
        CareerName
    }
}
```

1 - 1

Each course is taught in a classroom, and this classroom can only be used to teach this course

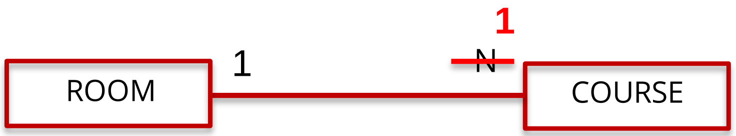
ROOM
RoomId*
RoomName

```
Room
{
    RoomId*
    RoomName
}
```

```
Course
{
    CourseId*
    CourseName
    RoomId
    RoomName
}
```

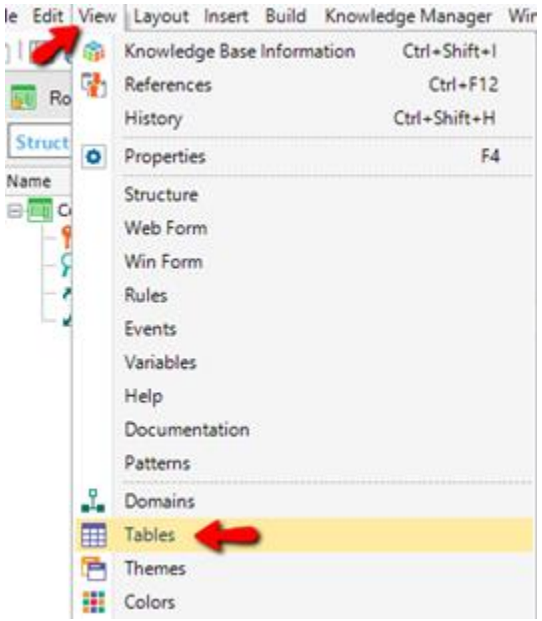
COURSE
CourseId*
CourseName
RoomId

Unique index

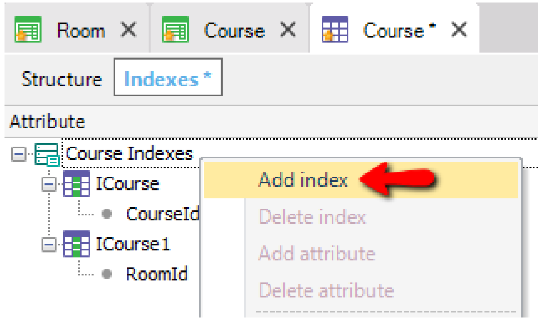


Creating an index

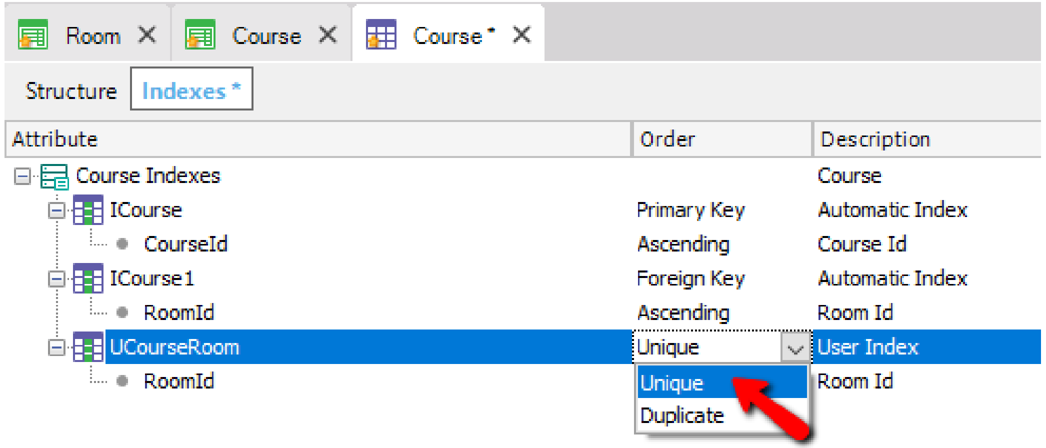
1



2



3



Attribute	Order	Description
Course Indexes		Course
ICourse	Primary Key	Automatic Index
CourseId	Ascending	Course Id
ICourse1	Foreign Key	Automatic Index
RoomId	Ascending	Room Id
UCourseRoom	Unique	User Index
RoomId		Room Id

Normalization

GeneXus normalize tables in Third Normal Form (3NF)

- Inferred attributes in a transaction, are not included in the generated table

```
Continent
{
    ContinentId*   (PK)
    ContinentName
}
```

CONTINENT
ContinentId*
ContinentName

```
Country
{
    CountryId*   (PK)
    CountryName
    ContinentId  (FK)
    ContinentName  (INF)

}
```

COUNTRY
CountryId*
CountryName
ContinentId

```
Customer
{
    CustomerId* (PK)
    CustomerName
    CountryId    (FK)
    CountryName  (INF)
    ContinentId  (INF)
    ContinentName  (INF)
}
```

CUSTOMER
CustomerId*
CustomerName
CountryId

Referential Integrity

Referential Integrity

```
Country
{
   CountryId*   (PK)
   CountryName
}
```

```
Customer
{
   CustomerId*
   CustomerName
   CountryId    (FK)
   CountryName
}
```

CountryId	CountryName
1	URUGUAY
2	ARGENTINA

CustomerId	CustomerName	CountryId	
1	ANA	1	✓
2	PEDRO	2	✓
3	LUIS	2	✓
4	JOSE	3	✗

The record is not inserted

Referential Integrity

Country
```
{
    CountryId*   (PK)
    CountryName
}
```

Customer
```
{
    CustomerId*
    CustomerName
    CountryId    (FK)
    CountryName
}
```

CountryId	CountryName
1	URUGUAY ✗
2	ARGENTINA

CustomerId	CustomerName	CountryId
1	ANA	$\boxed{1}$
2	PEDRO	2
3	LUIS	2
4	JOSE	3

The register is not deleted

Base Table and Extended Table

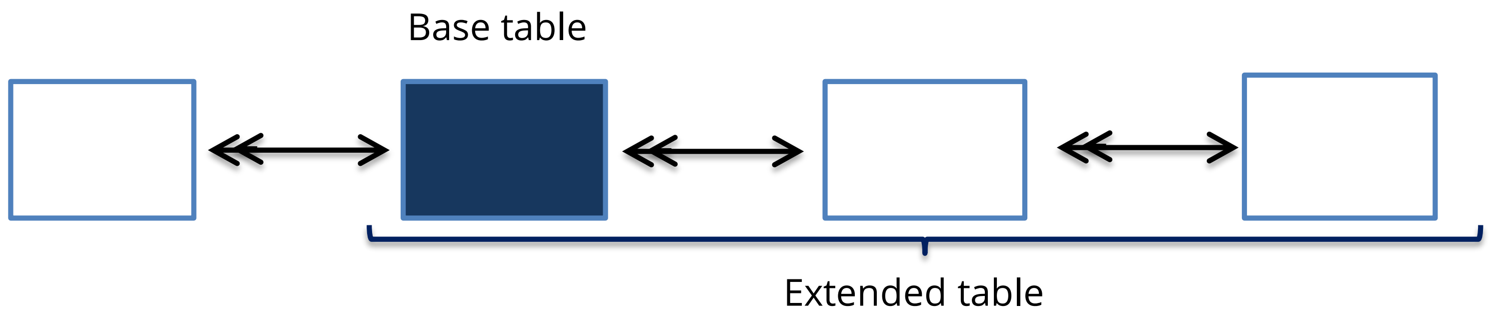
- **Base table**

  Any table in the database where we may be working at a given moment.

- **Extended table**

  For a given table, its extended table is a concept that allows us to consider all the information that we can access from it, using its foreign keys.

  It is the set of attributes of the table itself **+** all the attributes of the tables with which it has an N to 1 relation, either directly or indirectly.

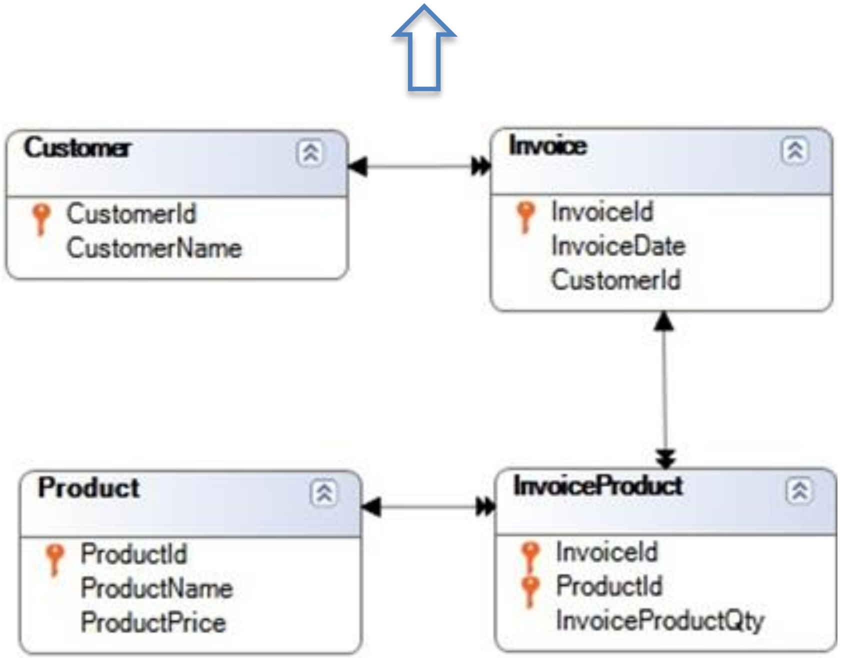


Example

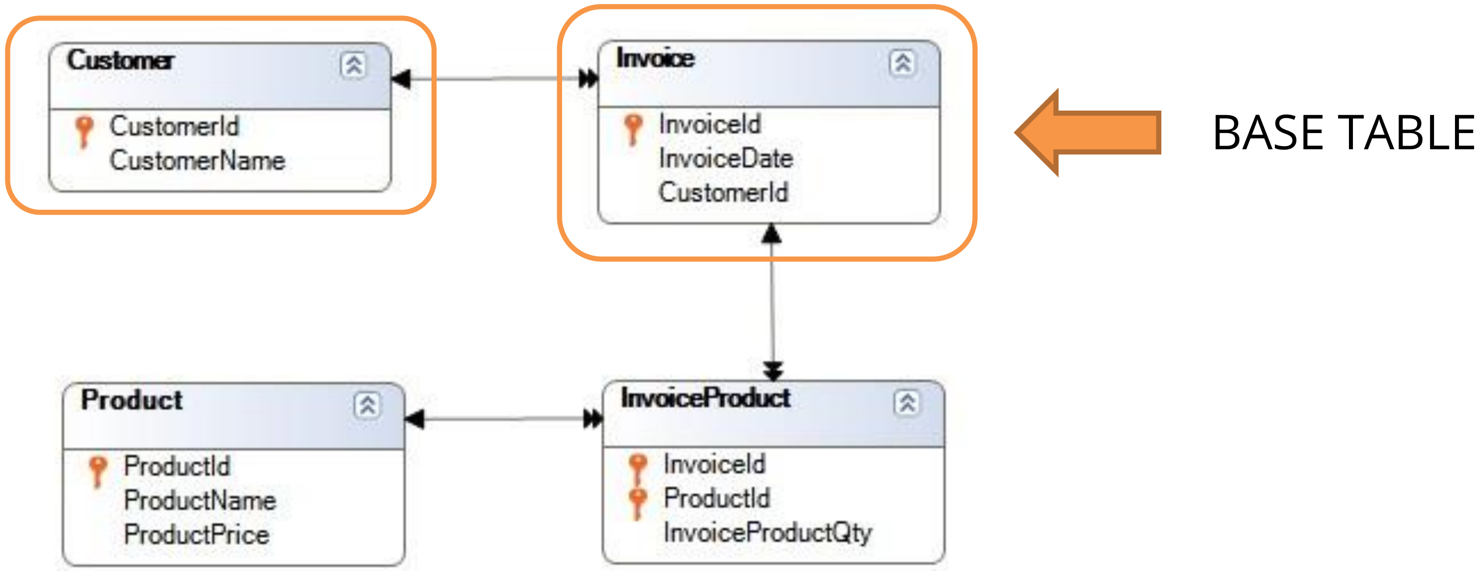
```
Customer
{
    CustomerId*
    CustomerName
}

Product
{
    ProductId*
    ProductName
    ProductPrice
}
```

```
Invoice
{
    InvoiceId*
    InvoiceDate
    CustomerId
    CustomerName
    Product
    {
        ProductId*
        ProductName
        ProductPrice
        InvoiceProductQuantity
        --------
    }
}
```

Table diagram
(Bachman diagram)

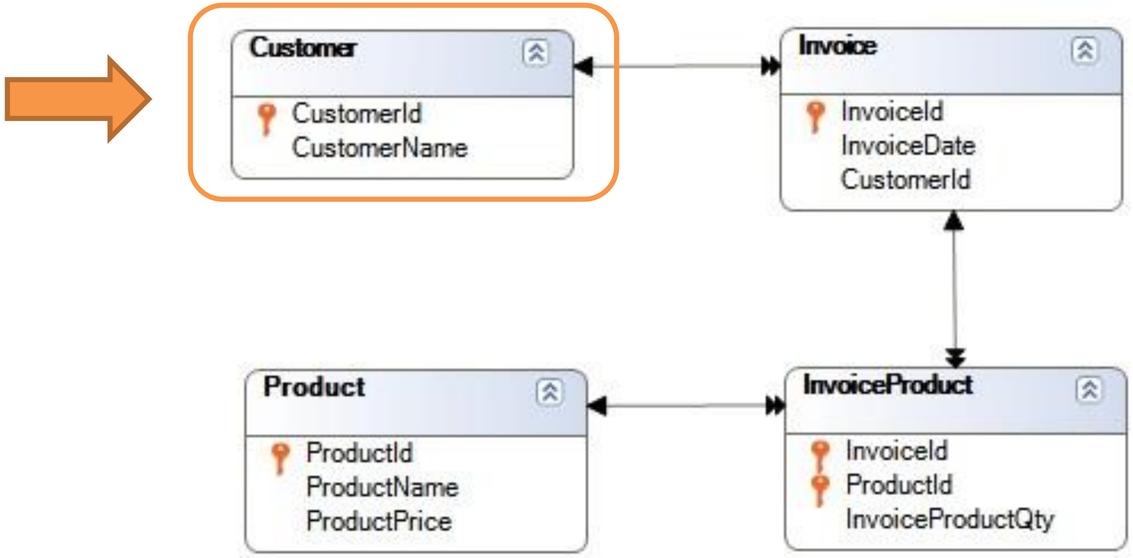


Example: Invoice Extended Table

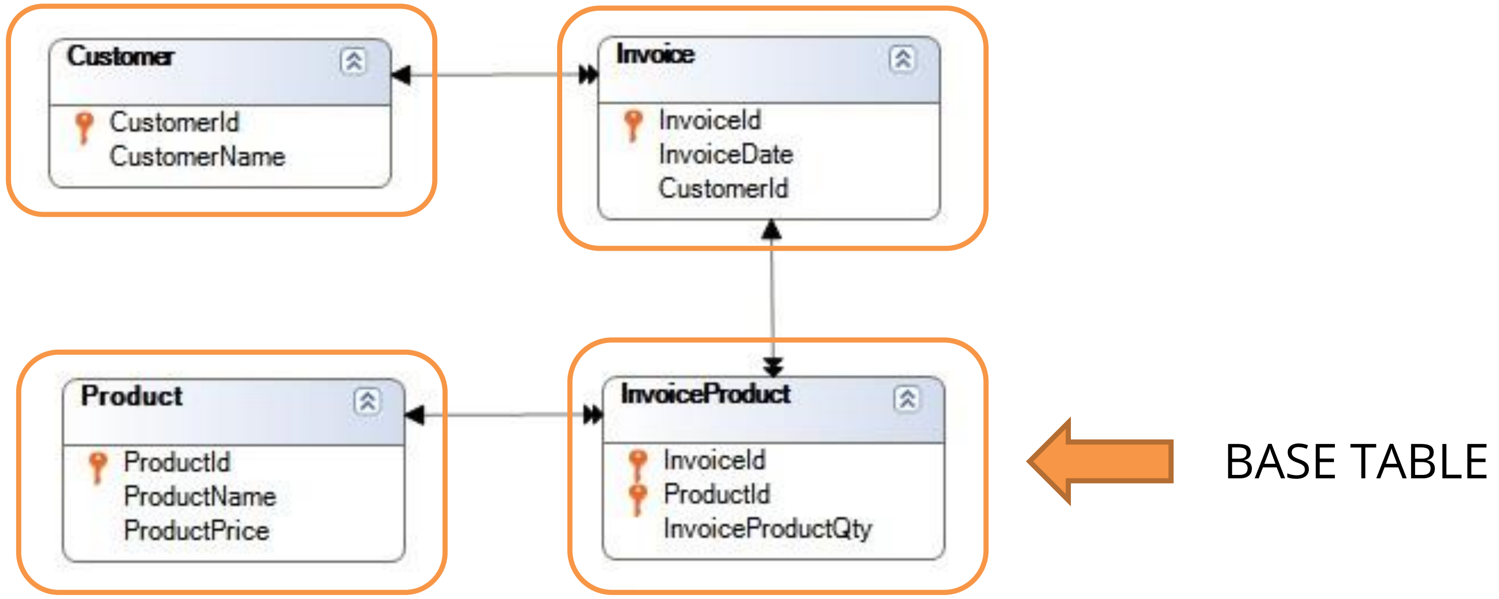


Example: Customer Extended Table

BASE TABLE

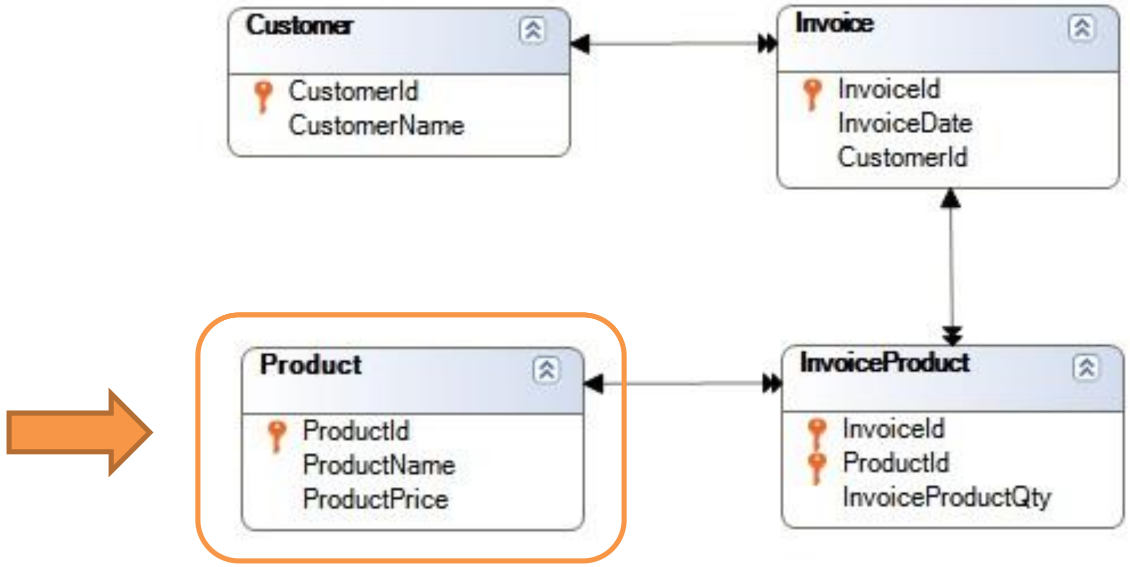


Example: InvoiceProduct Extended Table



Example: Product Extended Table

BASE TABLE



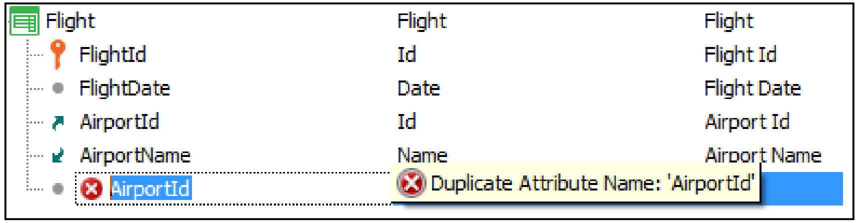
Subtypes

Multiple references:
For every flight, the departure and arrival airports must be saved

```
Airport
{
    AirportId*
    AirportName
}
```

```
Flight
{
    FlightId*
    FlightDate
    AirportId
    AirportName
    AirportId
    AirportName
}
```

Error due to duplicated attribute names



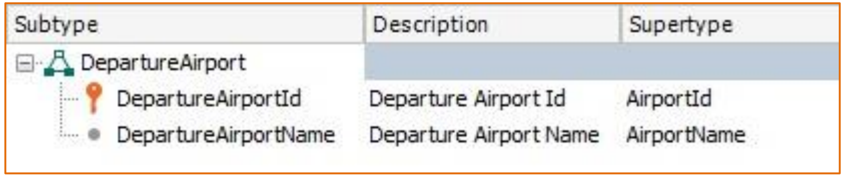
Flight	Flight	Flight
FlightId	Id	Flight Id
FlightDate	Date	Flight Date
AirportId	Id	Airport Id
AirportName	Name	Airport Name
AirportId		

Duplicate Attribute Name: 'AirportId'

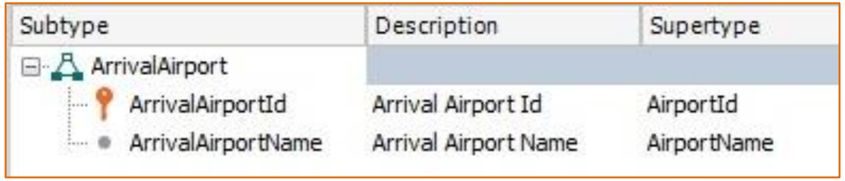
Solution 1 of 3: create two subtype groups, one for the departure airport and another for the arrival airport

```
Airport
{
    AirportId*
    AirportName
}
```

```
Flight
{
    FlightId*
    FlightDate
    DepartureAirportId      } Subtype group: DepartureAirport
    DepartureAirportName    }
    ArrivalAirportId        } Subtype group: ArrivalAirport
    ArrivalAirportName      }
}
```



Subtype	Description	Supertype
DepartureAirport		
DepartureAirportId	Departure Airport Id	AirportId
DepartureAirportName	Departure Airport Name	AirportName

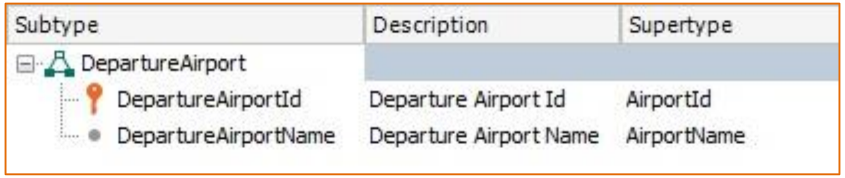


Subtype	Description	Supertype
ArrivalAirport		
ArrivalAirportId	Arrival Airport Id	AirportId
ArrivalAirportName	Arrival Airport Name	AirportName

Solution 2 of 3: create one subtype group for the departure airport only

Airport
```
{
    AirportId*
    AirportName
}
```

Flight
```
{
    FlightId*
    FlightDate
    DepartureAirportId     } Subtype group: DepartureAirport
    DepartureAirportName   }
    AirportId
    AirportName
}
```

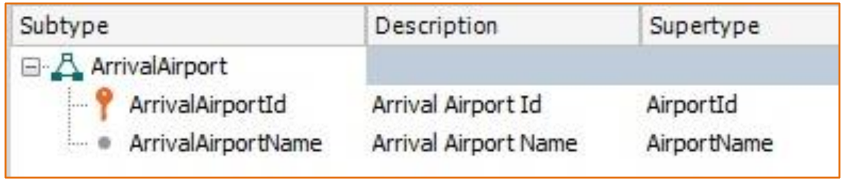


Subtype	Description	Supertype
DepartureAirport		
DepartureAirportId	Departure Airport Id	AirportId
DepartureAirportName	Departure Airport Name	AirportName

Solution 3 of 3: create one subtype group for the arrival airport only

```
Airport
{
    AirportId*
    AirportName
}
```

```
Flight
{
    FlightId*
    FlightDate
    AirportId
    AirportName
    ArrivalAirportId      } Subtype group: ArrivalAirport
    ArrivalAirportName    }
}
```



Subtype	Description	Supertype
ArrivalAirport		
ArrivalAirportId	Arrival Airport Id	AirportId
ArrivalAirportName	Arrival Airport Name	AirportName

Multiple references:
In addition to the customer's country, the country where the invoice was issued must also be saved

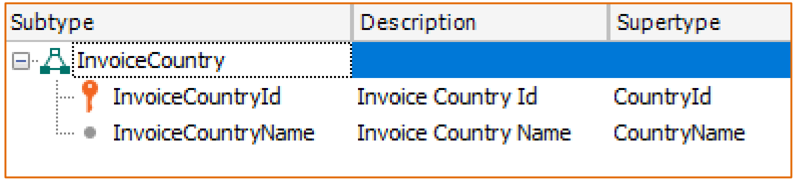
Country
{
  CountryId*
  CountryName
}

Customer
{
  CustomerId*
  CustomerName
  CountryId
  CountryName
}

Invoice
{
  InvoiceId*
  InvoiceDate
  CustomerId
  CustomerName
  CountryId
  CountryName
  InvoiceCountryId
  InvoiceCountryName
}

CustomerName, CountryId, CountryName: Inferred attributes

InvoiceCountryId, *InvoiceCountryName*: Subtype group: InvoiceCountry



Subtype	Description	Supertype
InvoiceCountry		
InvoiceCountryId	Invoice Country Id	CountryId
InvoiceCountryName	Invoice Country Name	CountryName

Multiple references: problem

```
Subject
{
   SubjectId*
   SubjectName
   TeacherId           ------- Permanent teacher
   TeacherName
   TeacherId           ------- Substitute teacher
   TeacherName
}
```

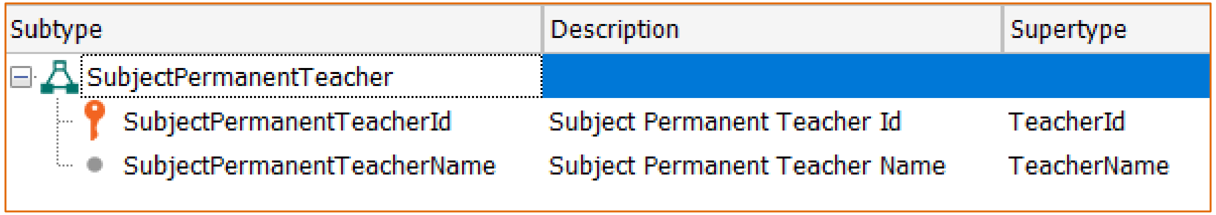
```
Teacher
{
   TeacherId*
   TeacherName
}
```

Error due to duplicated attribute names

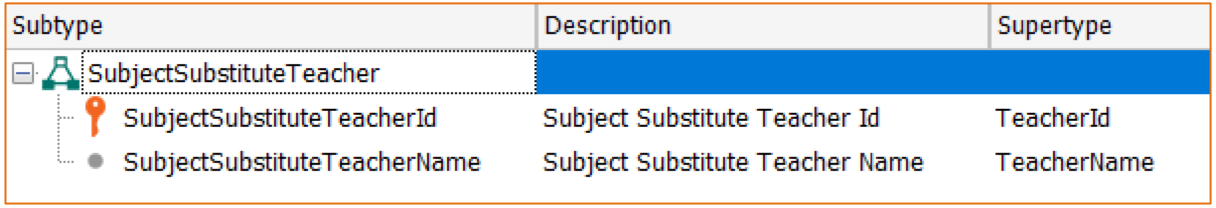
Multiple references: solution

Subject
```
{
    SubjectId*
    SubjectName
    SubjectPermanentTeacherId        ------- Permanent teacher
    SubjectPermanentTeacherName
    SubjectSubstituteTeacherId       ------- Substitute teacher
    SubjectSubstituteTeacherName
}
```

Teacher
```
{
    TeacherId*
    TeacherName
}
```



Subtype	Description	Supertype
SubjectPermanentTeacher		
SubjectPermanentTeacherId	Subject Permanent Teacher Id	TeacherId
SubjectPermanentTeacherName	Subject Permanent Teacher Name	TeacherName



Subtype	Description	Supertype
SubjectSubstituteTeacher		
SubjectSubstituteTeacherId	Subject Substitute Teacher Id	TeacherId
SubjectSubstituteTeacherName	Subject Substitute Teacher Name	TeacherName

Multiple references: problem

```
Subject
{
    SubjectId*
    SubjectName
}


Teacher
{
    TeacherId*
    TeacherName
    SubjectId
    SubjectName
}
```

```
Exam
{
    ExamId*
    ExamDate
    SubjectId
    SubjectName
    Teacher
    {
        TeacherId*
        TeacherName
        SubjectId
        SubjectName
    }
}
```

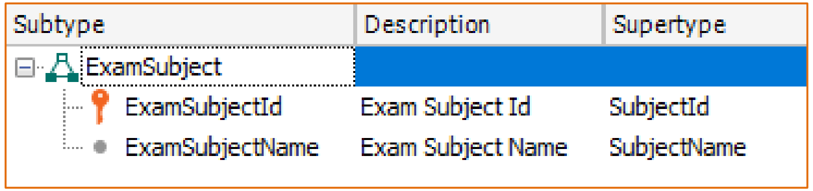
Inferred attributes ⇨ Error due to duplicated attribute names

Multiple references: solution

```
Subject
{
    SubjectId*
    SubjectName
}

Teacher
{
    TeacherId*
    TeacherName
    SubjectId
    SubjectName
}
```

```
Exam
{
    ExamId*
    ExamDate
    ExamSubjectId      }  Subtype group: ExamSubject
    ExamSubjectName    }
    Teacher
    {
        TeacherId*
        TeacherName  - - -
        SubjectId    - - -  Inferred attributes
        SubjectName  - - -
    }
}
```

ExamSubjectId
ExamSubjectName

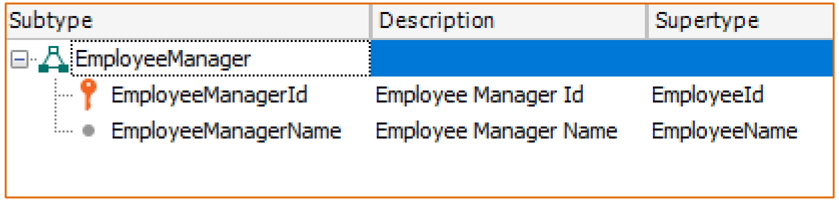


Subtype	Description	Supertype
ExamSubject		
ExamSubjectId	Exam Subject Id	SubjectId
ExamSubjectName	Exam Subject Name	SubjectName

Recursive subtypes

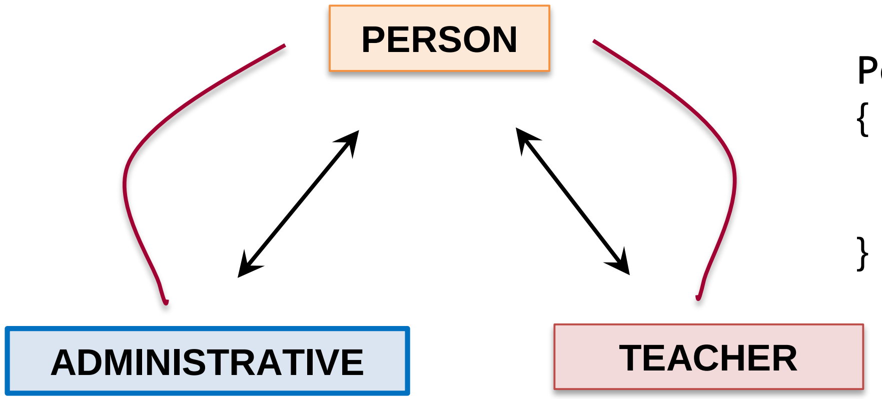
```
Employee
{
    EmployeeId*
    EmployeeName
    EmployeeManagerId
    EmployeeManagerName
}
```

Subtype group: EmployeeManager



Subtype	Description	Supertype
EmployeeManager		
EmployeeManagerId	Employee Manager Id	EmployeeId
EmployeeManagerName	Employee Manager Name	EmployeeName

Specialization

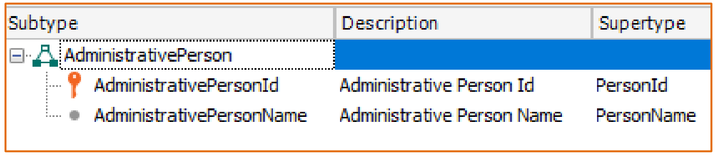


Person
{
    PersonId*
    PersonName
}

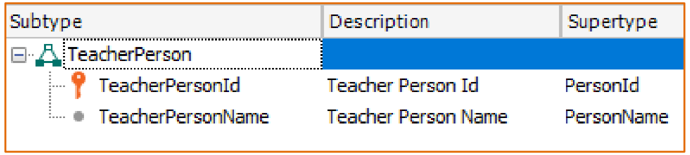
Correct:
More inf attrbs
Show tables

Administrative
{
    AdministrativeId*
    AdministrativeName
    AdministrativeLanguage
}

Teacher
{
    TeacherId*
    TeacherName
    TeacherTitle
}



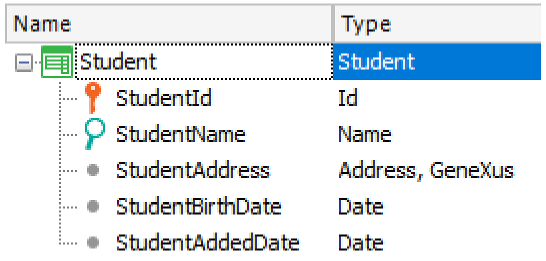
Subtype	Description	Supertype
AdministrativePerson		
AdministrativePersonId	Administrative Person Id	PersonId
AdministrativePersonName	Administrative Person Name	PersonName



Subtype	Description	Supertype
TeacherPerson		
TeacherPersonId	Teacher Person Id	PersonId
TeacherPersonName	Teacher Person Name	PersonName

Rules

Rules



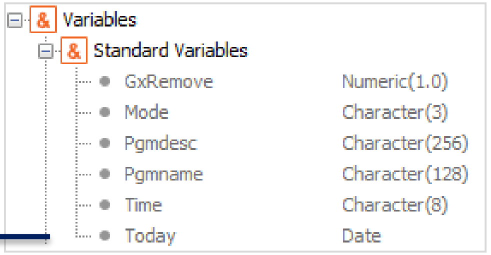
Name	Type
Student	Student
StudentId	Id
StudentName	Name
StudentAddress	Address, GeneXus
StudentBirthDate	Date
StudentAddedDate	Date

Error("Enter the student name") if StudentName.isEmpty();

Msg("The address is empty") if StudentAddress.isEmpty();

Default(StudentAddedDate, &Today);

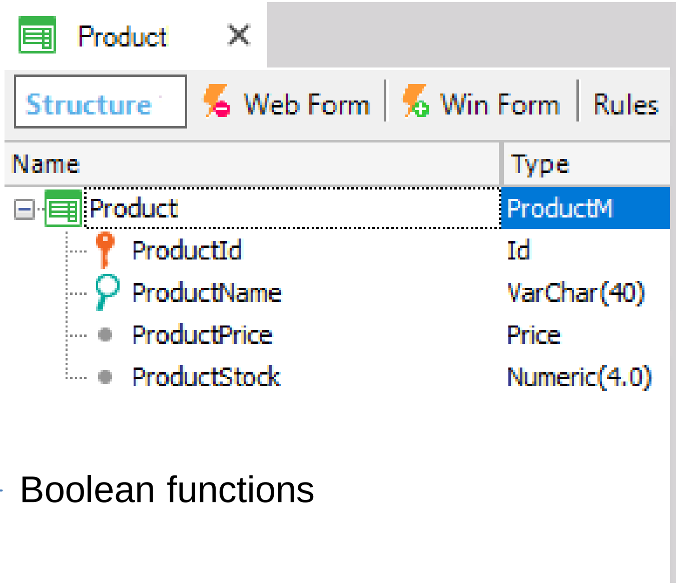
Noaccept(StudentAddedDate);



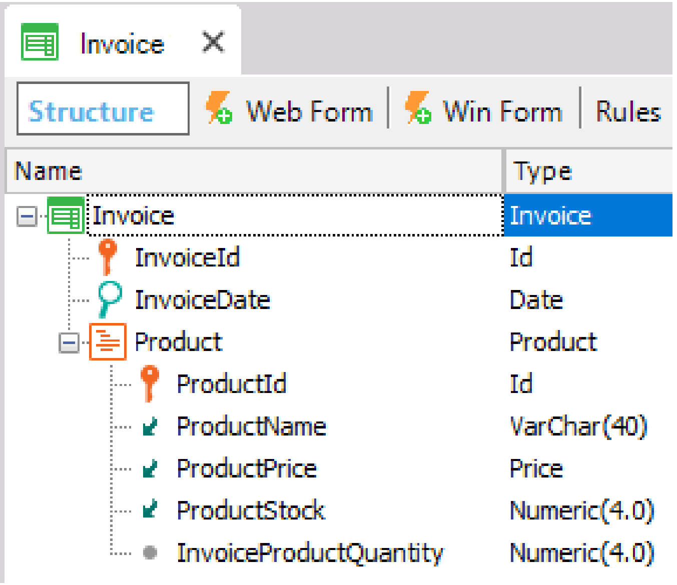
Variables
Standard Variables

GxRemove	Numeric(1.0)
Mode	Character(3)
Pgmdesc	Character(256)
Pgmname	Character(128)
Time	Character(8)
Today	Date

Rules



Name	Type
Product	ProductM
ProductId	Id
ProductName	VarChar(40)
ProductPrice	Price
ProductStock	Numeric(4.0)



Name	Type
Invoice	Invoice
InvoiceId	Id
InvoiceDate	Date
Product	Product
ProductId	Id
ProductName	VarChar(40)
ProductPrice	Price
ProductStock	Numeric(4.0)
InvoiceProductQuantity	Numeric(4.0)

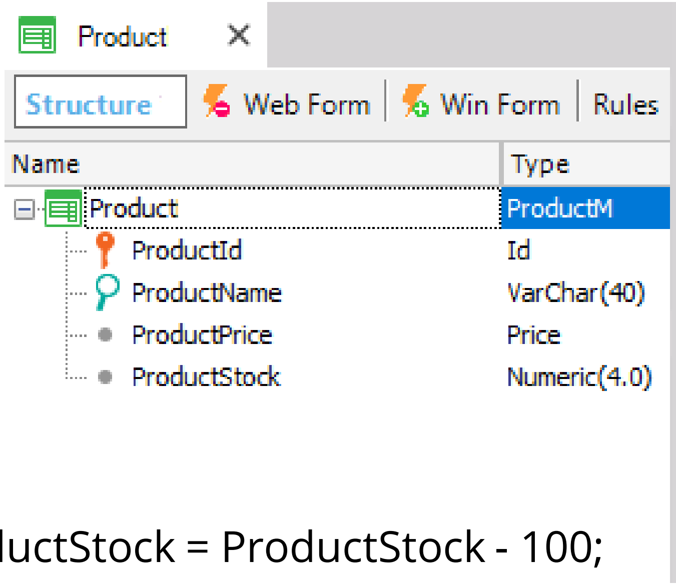
Insert
update
delete
display

Boolean functions

ProductStock = ProductStock - 100 if insert;

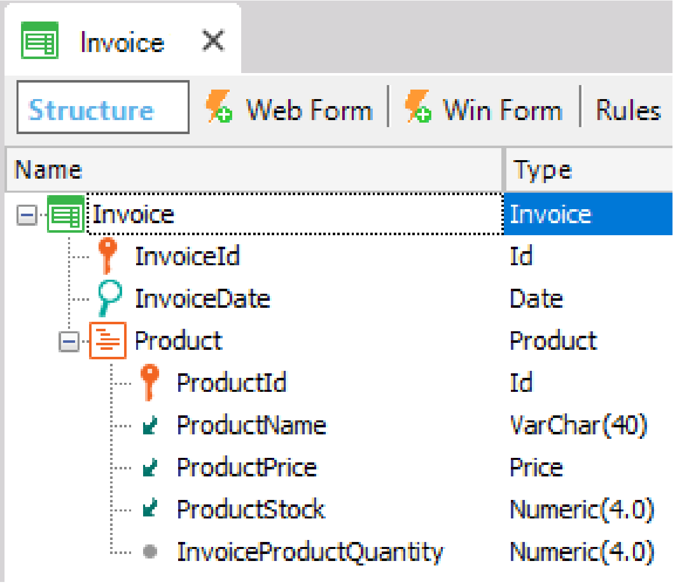
ProductStock = ProductStock + 100 if delete;

Rules


Product

Structure | Web Form | Win Form | Rules

Name	Type
Product	ProductM
ProductId	Id
ProductName	VarChar(40)
ProductPrice	Price
ProductStock	Numeric(4.0)


Invoice

Structure | Web Form | Win Form | Rules

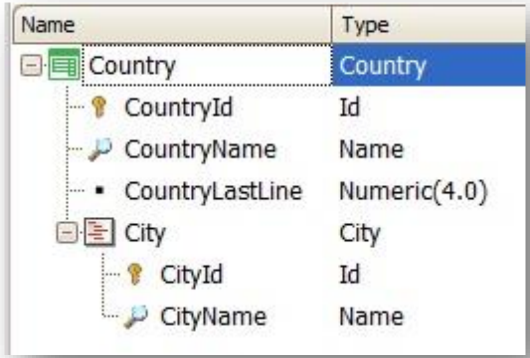
Name	Type
Invoice	Invoice
InvoiceId	Id
InvoiceDate	Date
Product	Product
ProductId	Id
ProductName	VarChar(40)
ProductPrice	Price
ProductStock	Numeric(4.0)
InvoiceProductQuantity	Numeric(4.0)

ProductStock = ProductStock - 100;

Subtract(InvoiceProductQuantity, ProductStock);

Add(500, ProductStock);

Rules



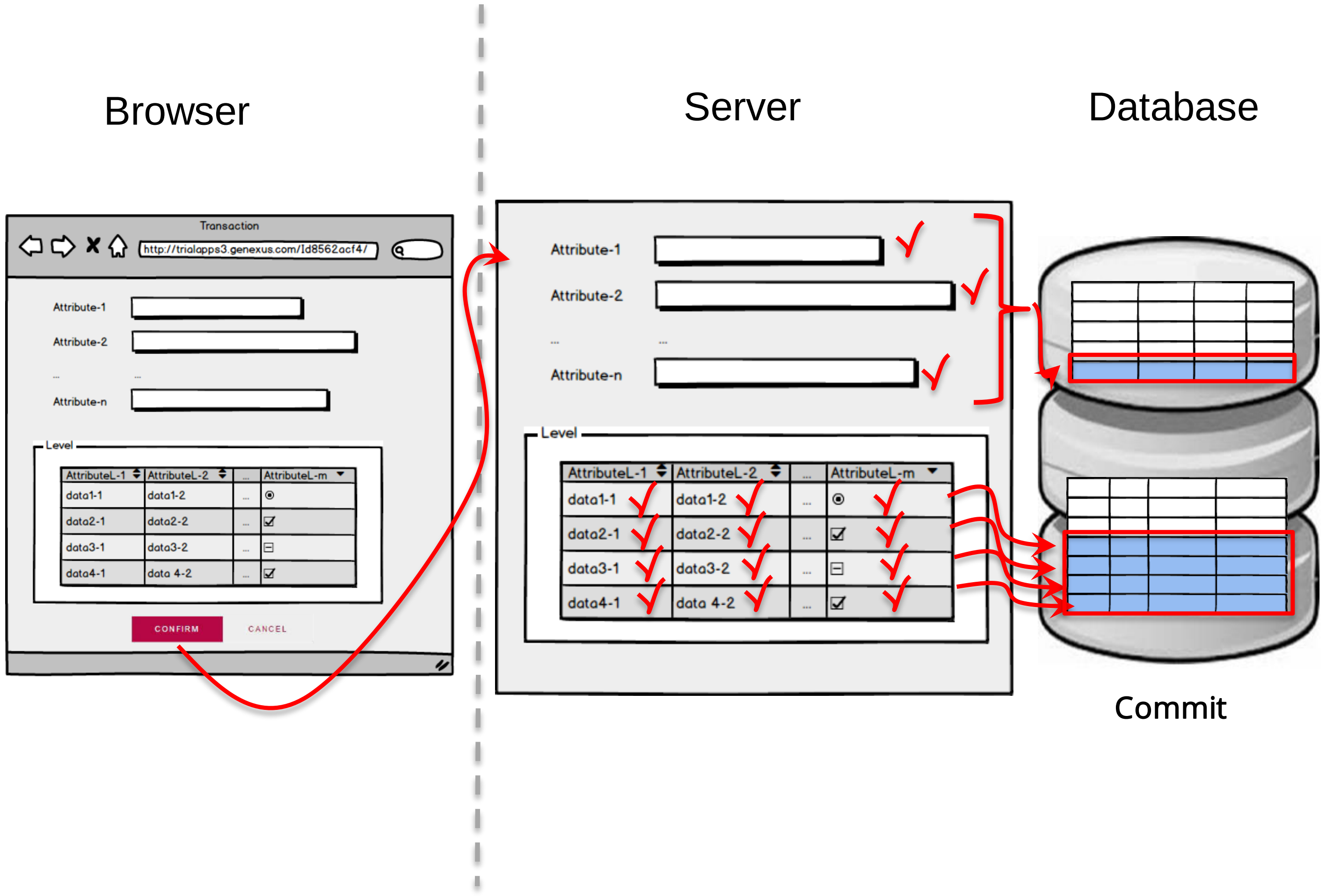
Name	Type
Country	Country
CountryId	Id
CountryName	Name
CountryLastLine	Numeric(4.0)
City	City
CityId	Id
CityName	Name

Serial(CityId, CountryLastLine, 1);

Parm(attribute1, &variable1, ….);

Variable: Space in memory that has a name and data type it can save. It is referenced using “&.”

Triggering Moments


Browser
Server
Database
Transaction
http://trialapps3.genexus.com/Id8562acf4/
Attribute-1
Attribute-2
...
Attribute-n
Level
AttributeL-1
AttributeL-2
AttributeL-m
data1-1
data1-2
data2-1
data2-2
data3-1
data3-2
data4-1
data 4-2
CONFIRM
CANCEL
Commit

Rule triggering moments

In the server, after pressing Confirm

In single-level transactions:

On BeforeValidate
VALIDATION
On AfterValidate
On BeforeInsert/BeforeUpdate/ BeforeDelete
SAVING
On AfterInsert/AfterUpdate/ AfterDelete

On BeforeComplete
COMMIT
On AfterComplete

In two-level transactions:

On BeforeValidate
VALIDATION of the header
On AfterValidate
On BeforeInsert/BeforeUpdate/BeforeDelete
SAVING the header
On AfterInsert/AfterUpdate/AfterDelete

For each line $\left\{ \begin{array}{l} \text{On BeforeValidate} \\ \text{VALIDATION of the line} \\ \text{On AfterValidate} \\ \text{OnBeforeInsert/BeforeUpdate/BeforeDelete} \\ \text{SAVING the line} \\ \text{On AfterInsert/AfterUpdate/AfterDelete} \end{array} \right.$

On AferLevel Level Line attribute
On BeforeComplete
COMMIT
On AfterComplete

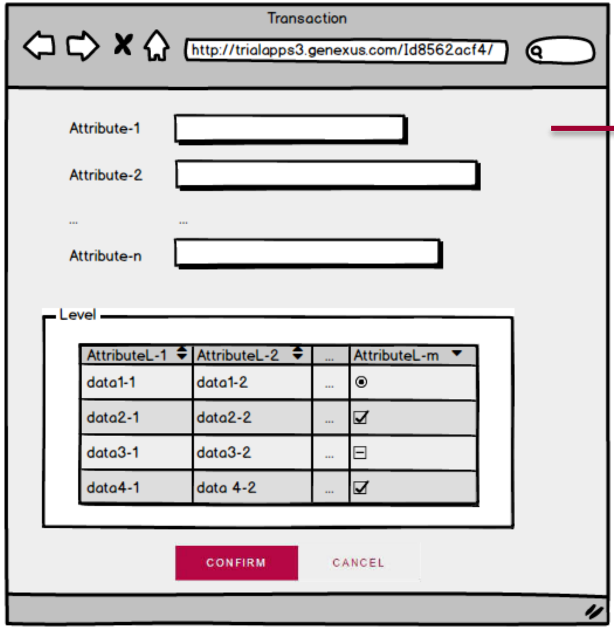
GeneXus™

Rule triggering moments

PrintCustomer(CustomerId) on AfterValidate;

Is it correct or not? ✗

It is not correct because it is invoked BEFORE saving and the table will not reflect the changes made to the customer.



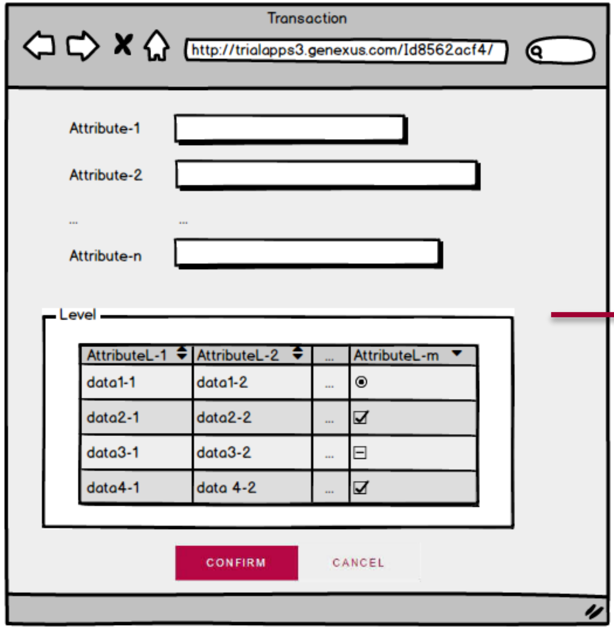
VALIDATION
On AfterValidate

SAVING
On AfterInsert / On AfterUpdate / On AfterDelete

Rule triggering moments

PrintCustomer(CustomerId) on AfterInsert, AfterUpdate;

Is it correct or not? ✓ It is correct!



AttributeL-1	AttributeL-2	...	AttributeL-m
data1-1	data1-2	...	◉
data2-1	data2-2	...	☑
data3-1	data3-2	...	⊟
data4-1	data 4-2	...	☑

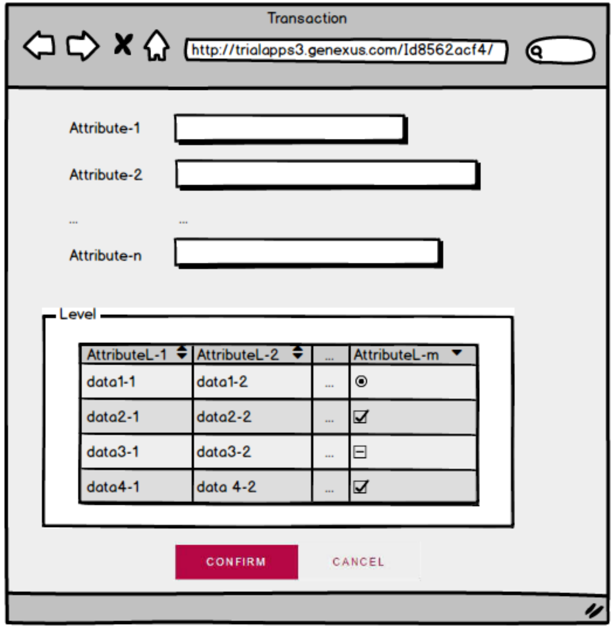
VALIDATION
On Aftervalidate

SAVING
On AfterInsert / On AfterUpdate / On AfterDelete

Rule triggering moments

PrintCustomer(CustomerId) on AfterDelete; Is it correct or not? ✗

It is not correct because it is invoked AFTER the deletion and the customer will not be found with that ID in the table.

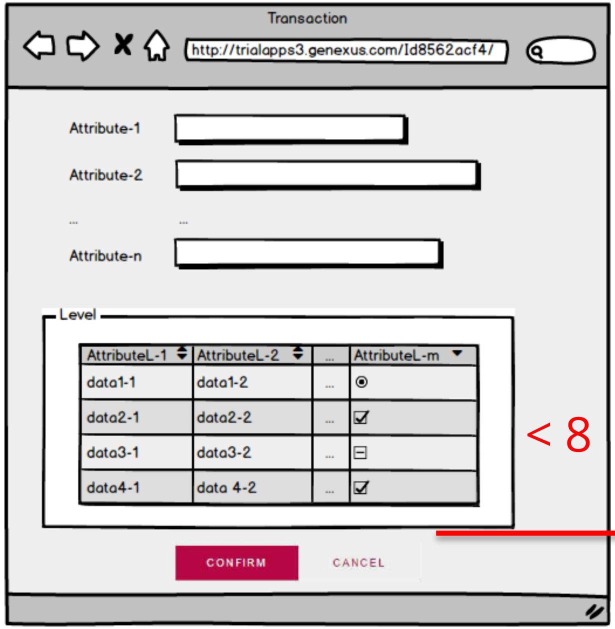


VALIDATION
On Aftervalidate

SAVING
On AfterInsert / On AfterUpdate / On AfterDelete

Rule triggering moments

```
Error('The seat quantity should be equal or greather than 8') if FlightCapacity<8
    on AfterLevel
    Level FlightSeatChar;
```



< 8

In two-level transactions:

On BeforeValidate
VALIDATION of the header
On AfterValidate
On BeforeInsert/BeforeUpdate/BeforeDelete
SAVING the header
On AfterInsert/AfterUpdate/AfterDelete

For each line:
- On BeforeValidate
- VALIDATION of the line
- On AfterValidate
- OnBeforeInsert/BeforeUpdate/BeforeDelete
- SAVING the line
- On AfterInsert/AfterUpdate/AfterDelete

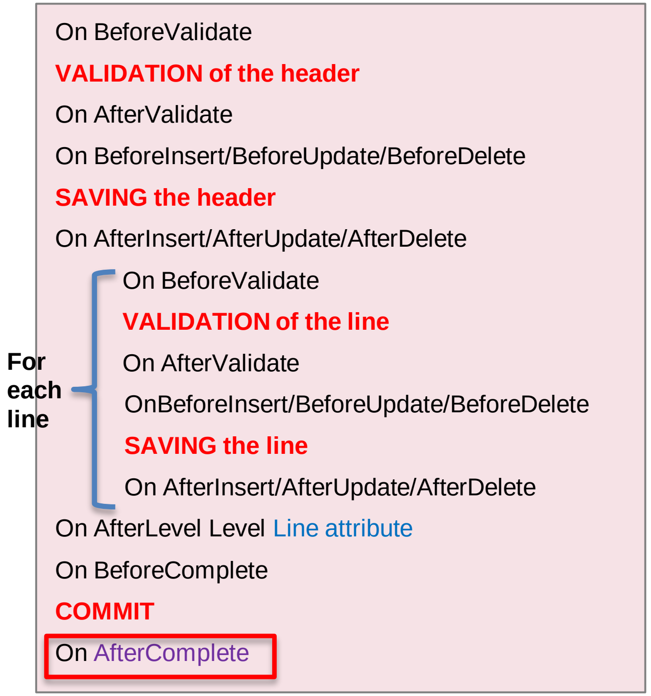
On AferLevel Level Line attribute
On BeforeComplete
COMMIT
On AfterComplete

Rule triggering moments

```
PrintFlight(FlightId) on AfterComplete;
```

✓on AfterComplete: Right after **Commit** is performed in the database.

In two-level transactions:


On BeforeValidate
VALIDATION of the header
On AfterValidate
On BeforeInsert/BeforeUpdate/BeforeDelete
SAVING the header
On AfterInsert/AfterUpdate/AfterDelete

For each line
- On BeforeValidate
- **VALIDATION of the line**
- On AfterValidate
- OnBeforeInsert/BeforeUpdate/BeforeDelete
- **SAVING the line**
- On AfterInsert/AfterUpdate/AfterDelete

On AfterLevel Level Line attribute
On BeforeComplete
COMMIT
On AfterComplete

Examples

Determine if is it correct or not:

```
Invoice
{
   InvoiceId*
   InvoiceDate
   -------
   Product
   {
      ProductId*
      ProductName
      ProductPrice
      InvoiceProductQuantity
      --------
    }
 }
```

PrintInvoiceDetail(InvoiceId) on **AfterComplete**; ✓

ProductControl(ProductId) **on BeforeInsert**; ✗

ProductControl(ProductId) **on AfterComplete**; ✗

Can I assign a value to an attribute on AfterInsert? ⇨ NO

Value assignment examples

```
Product
{
   ProductId*
   ProductName
   ProductPrice
}
```

On BeforeValidate
VALIDATION
On AfterValidate
On BeforeInsert/BeforeUpdate/ BeforeDelete
SAVING
On AfterInsert/AfterUpdate/ AfterDelete
On BeforeComplete
COMMIT
On AfterComplete

Determine if is it correct or not:

ProductPrice = 100 **on BeforeInsert;** ✓

ProductPrice = 100 **on BeforeComplete;** ✗

ProductPrice = 100 **on AfterValidate;** ✓

ProductPrice = 100 **on AfterInsert;** ✗

Formulas

Global Formulas

- It's a calculation associated with an attribute in a transaction
- They are known throughout the KB

```
Product
{
    ProductId*
    ProductName
    ProductPrice
}
```

```
Invoice
{
    InvoiceId*
    InvoiceDate
    InvoiceAmount
    Product
    {
        ProductId*
        ProductName
        ProductPrice
        InvoiceProductQuantity
        InvoiceLineAmount
    }
}
```

InvoiceAmount → Sum(InvoiceProductAmount)

"Virtual" attributes (not saved in the DB)

InvoiceLineAmount:

```
ProductPrice*InvoiceProductQuantity*0.9 if ProductId = 1;
ProductPrice*InvoiceProductQuantity*0.8 if ProductId = 3;
ProductPrice*InvoiceProductQuantity otherwise;
```

```
Country
{
   CountryId*
   CountryName
   CountryCustomersQuantity
}
```

```
Customer
{
   CustomerId*
   CustomerName
   CountryId
   CountryName
}
```

$$\downarrow$$

$\boxed{\text{Count(CustomerName)}}$

Will this formula count the customers by country or the total number of customers?

It will count the customers by country because GeneXus applies an <u>automatic filter</u> by the common attribute (*CountryId*).

```
Customer
{
    CustomerId*
    CustomerName
    CustomerTotal
}
```

```
Invoice
{
    InvoiceId*
    InvoiceDate
    InvoiceType
    CustomerId
    CustomerName
    InvoiceAmount
}
```

Domain that provides two Enum Values

credit

cash

Sum(InvoiceAmount, InvoiceType=InvoiceType.Credit)

Calculation condition

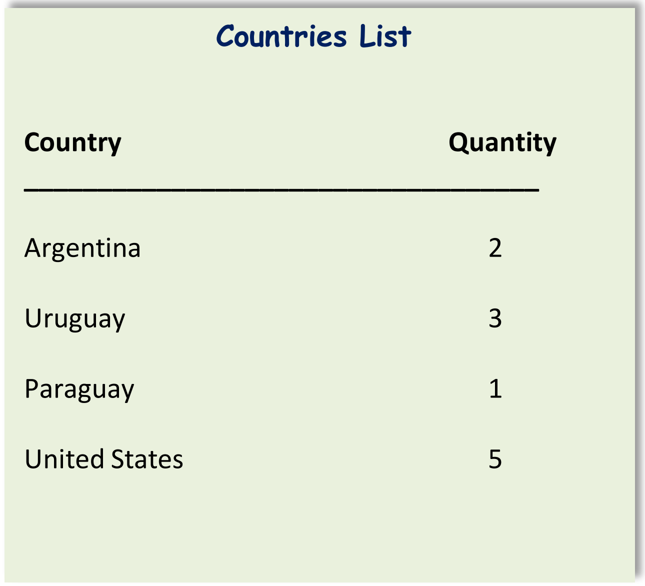
If CustomerId = 3

Triggering condition

Inline Formulas

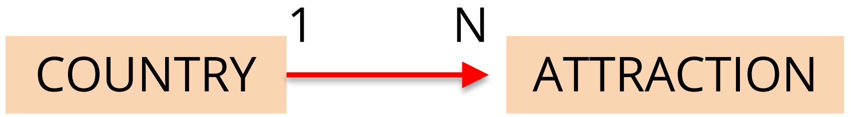
- They are formulas defined in the code section of an object
- They are only known in the object in which they have been defined

Requirement: A list of countries with the number of attractions in each one of them



Countries List

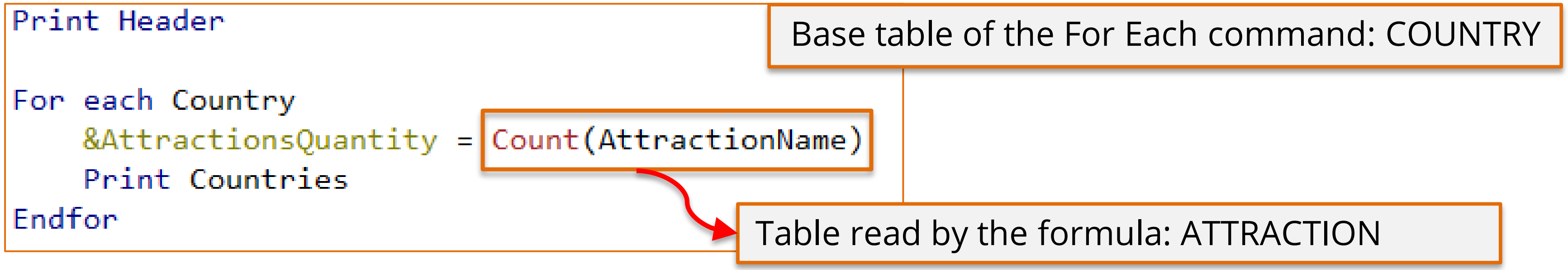
Country	Quantity
Argentina	2
Uruguay	3
Paraguay	1
United States	5



```
Country
{
   CountryId*
   CountryName
}
```

```
Attraction
{
   AttractionId*
   AttractionName
   CountryId
   CountryName
}
```

Inline Formula in the code of a Procedure object



```
Print Header

For each Country
    &AttractionsQuantity = Count(AttractionName)
    Print Countries
Endfor
```

Base table of the For Each command: COUNTRY
Table read by the formula: ATTRACTION

Will this formula count the attractions by country or the total number of attractions?

It will count the attractions by country because it is applied an <u>automatic filter</u> by the common attribute *CountryId* (both tables are related).

For Each command

Base Transaction

```
Flight
{
   FlightId*
   FlightDate
   -----
   Seat
   {
     FlightSeatId*
     FlightSeatChar
   }
}
```

```
For each Flight
     ---------
Endfor
```

```
For each Flight.Seat
     ---------
Endfor
```

Base Transaction

Name of the transaction whose associated physical table is to be run through

Order

```
Customer
{
   CustomerId*
   CustomerName
   ------
}
```

Requirement: A list of all customers in alphabetical order by name.

```
For each Customer order CustomerName
    -------
Endfor
```

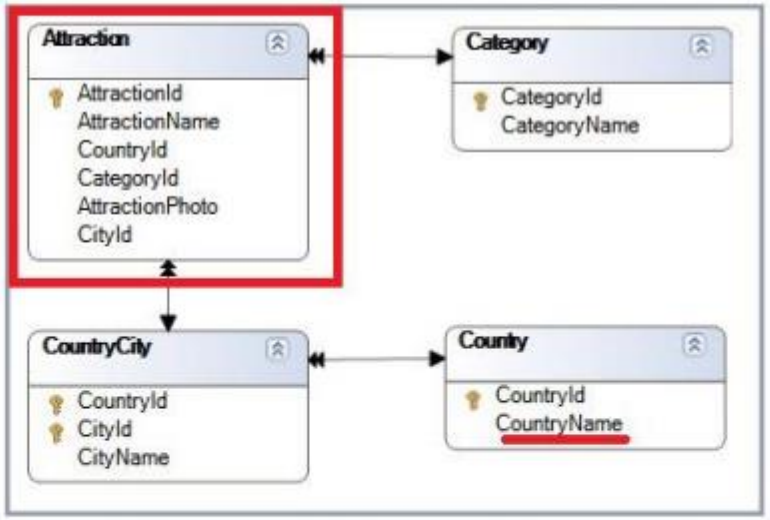
Requirement: A list of all customers in descending order by name.

```
For each Customer  order (CustomerName)
    -------
Endfor
```

Order

GeneXus allows ordering by the value of an attribute not included in the table being run through, but in its extended table.

```
Print Header
For each Attraction order CountryName
    Print Attractions
Endfor
```



Filters

```
Flight
{
    FlightId*
    FlightDate
    -----
    Seat
    {
      FlightSeatId*
      FlightSeatChar
    }
}
```

```
For each Flight
      Where FlightDate = Today()
            ---------
Endfor
```

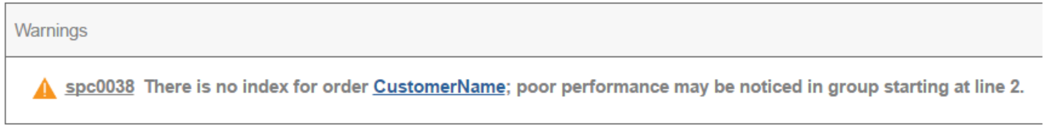
```
For each Flight.Seat
      Where FlightId = 1
            --------
Endfor
```

Filters

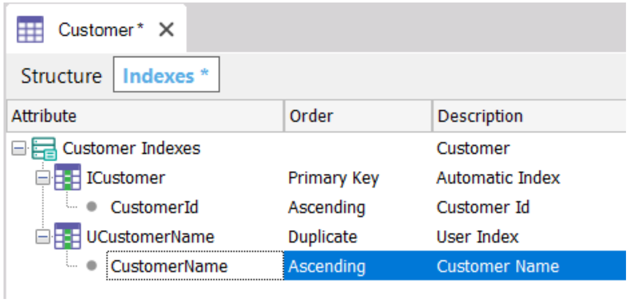
```
Customer
{
   CustomerId*
   CustomerName
   CustomerAddress
}
```

Index

```
For each Customer order CustomerName
    Where CustomerName >= &NameFrom
          ---------
Endfor
```


Warnings

spc0038 There is no index for order CustomerName; poor performance may be noticed in group starting at line 2.



Attribute	Order	Description
Customer Indexes		Customer
ICustomer	Primary Key	Automatic Index
CustomerId	Ascending	Customer Id
UCustomerName	Duplicate	User Index
CustomerName	Ascending	Customer Name

The query has been optimized! ✓

For Each command syntax

```
For each BaseTransaction
    skip expression1 count expression2
    order att1, att2, … , attn [when condition]
    order att1, att2, … , attn [when condition]
    unique att1, att2, … , attn
    using DataSelector( parm1, parm2, … , parmn)
    where condition [when condition]
    where condition [when condition]
    where att IN DataSelector( parm1, parm2, … , parmn)

      main code

When none
      …
Endfor
```

Nested For Each commands + Different base table + Tables NOT related = CARTESIAN PRODUCT

```
Country
{
   CountryId*
   CountryName
}


Room
{
   RoomId*
   RoomName
}
```

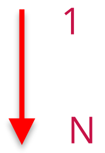
Cartesian Product

```
For each Country
      Print Country

   For each Room
      Print Room
   Endfor

Endfor
```

```
1 - Brazil
               RoomA
               RoomB
               RoomC
2 -Uruguay
               RoomA
               RoomB
               RoomC
3 - Argentina
               RoomA
               RoomB
               RoomC
4 - United States
               RoomA
               RoomB
               RoomC
```

Nested For Each commands + Different base table + Related tables = JOIN

```
Country
{
   CountryId*
   CountryName
}
```



```
Customer
{
   CustomerId*
   CustomerName
   CountryId
   CountryName
}
```

Join

```
For each Country
     Print Country

   For each Customer
      Print Customer
   Endfor

Endfor
```

```
1 - Brazil
              LUIS
              JORGE

2 - Uruguay


3 - Argentina


4 - United States
              ANA
```

Nested For each commands + Same base table + Related tables = CONTROL BREAK

```
Country
{
   CountryId*
   CountryName
}


Customer
{
   CustomerId*
   CustomerName
   CountryId
   CountryName
}
```

Control Break

```
For each Customer order CountryId
      Print Country

    For each Customer
        Print Customer
    Endfor

Endfor
```

```
1 - Brazil
              LUIS
              JORGE

4 - United States
              ANA
```

Summary

Cartesian Product

```
For each Country
    Print Country

    For each Room
        Print Room
    Endfor

Endfor
```

Different tables, with no relation between them

Join

```
For each Country
    Print Country

    For each Customer
        Print Customer
    Endfor

Endfor
```

Different tables which are related

Control Break

```
For each Customer order CountryId
    Print Country

    For each Customer
        Print Customer
    Endfor

Endfor
```

Same table, grouped with order

Summary

Cartesian Product	Join	Control Break

1 - Brazil
RoomA
RoomB
RoomC
2 -Uruguay
RoomA
RoomB
RoomC
3 - Argentina
RoomA
RoomB
RoomC

"Both entities are not related; show all possibilities for each country"

1 - Brazil
LUIS
JORGE
2 - Uruguay
3 - Argentina
4 - United States
ANA

"All countries and their customers, regardless if they have customers or not"

1 - Brazil
LUIS
JORGE
4 - United States
ANA

"Only those countries that have customers"

Communication between objects

Example: sending parameters

```
Country
{
   CountryId*
   CountryName
}


Customer
{
   CustomerId*
   CustomerName
   CountryId
   CountryName
}
```

For example, in the Rules of the Country trn:

```
CustomerList(CountryId) on AfterComplete;
```

$\downarrow$

```
CustomerList

Parm(in: &CountryId);

For each Customer
   Where CountryId = &CountryId
      ---------
Endfor
```

Variable
Explicit filter

Example: sending parameters

```
Country
{
    CountryId*
    CountryName
}
```

```
Customer
{
    CustomerId*
    CustomerName
    CountryId
    CountryName
}
```

For example, in the Rules of the Country trn :

```
CustomerList(CountryId) on AfterComplete;
```

CustomerList

```
Parm(in: CountryId);

For each Customer
   ~~Where CountryId = &CountryId~~
        ---------
Endfor
```

Attribute
Implicit filter

Example: returning a value

```
Country
(
   CountryId*
   CountryName
)


Customer
(
   CustomerId*
   CustomerName
   CountryId
   CountryName
)
```

For example, in the Rules of the Customer trn:

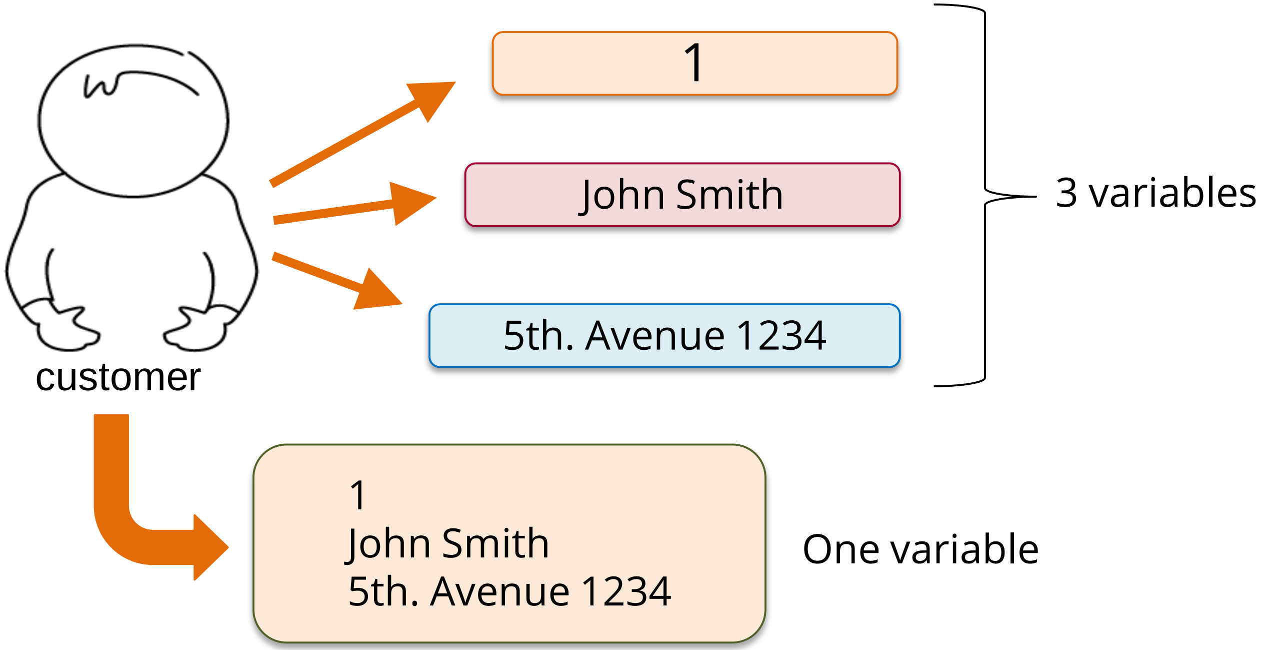
```
&Control = CustomerControl(CustomerId);
```

CustomerControl

```
Parm(in: &CustomerId, out: &Control);

For each Customer
   Where CustomerId = &CustomerId
      ---------
      &Control = True
Endfor
```

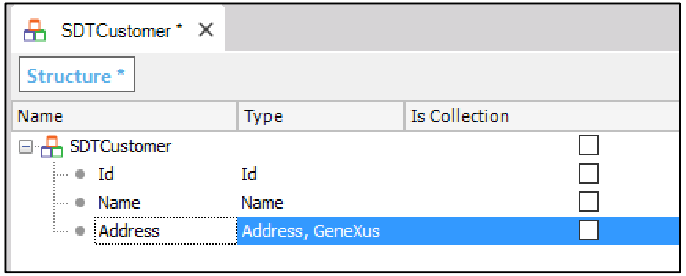
Structured Data Types

Concept



customer
1
John Smith
5th. Avenue 1234
3 variables
1
John Smith
5th. Avenue 1234
One variable

Definition



SDTCustomer *

Structure *

Name	Type	Is Collection
SDTCustomer		☐
Id	Id	☐
Name	Name	☐
Address	Address, GeneXus	☐

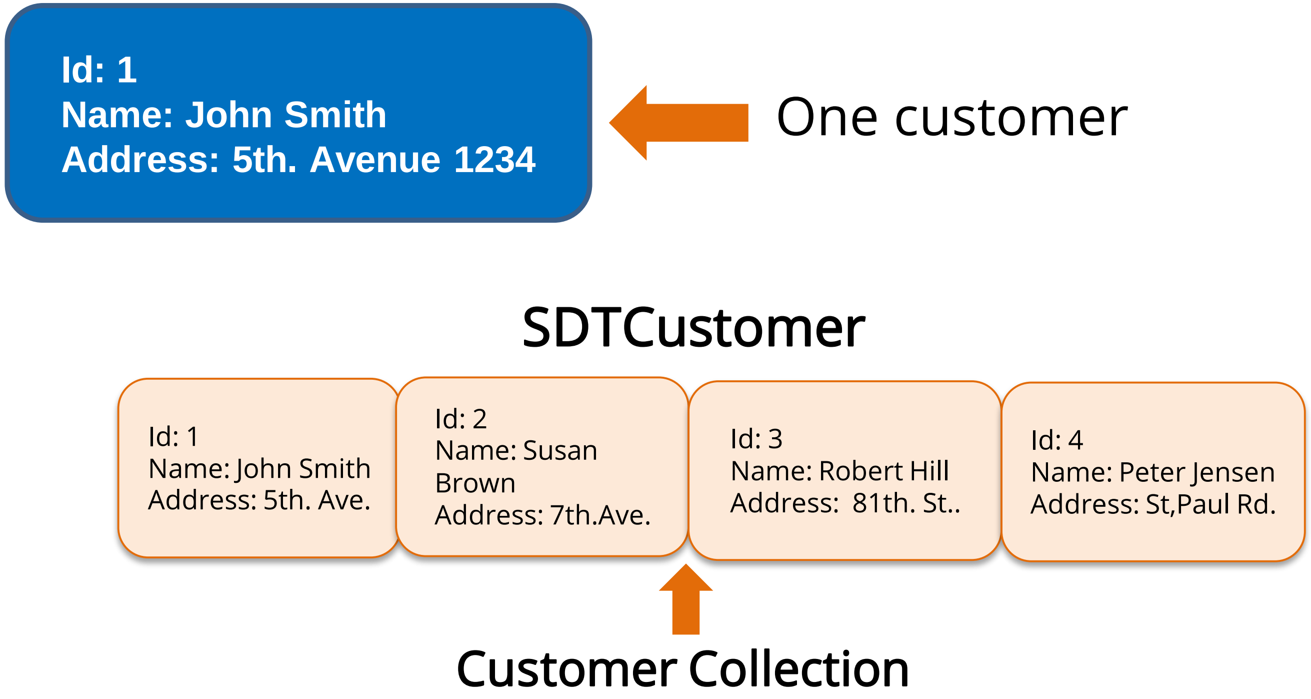
Structured Data Type object

customer

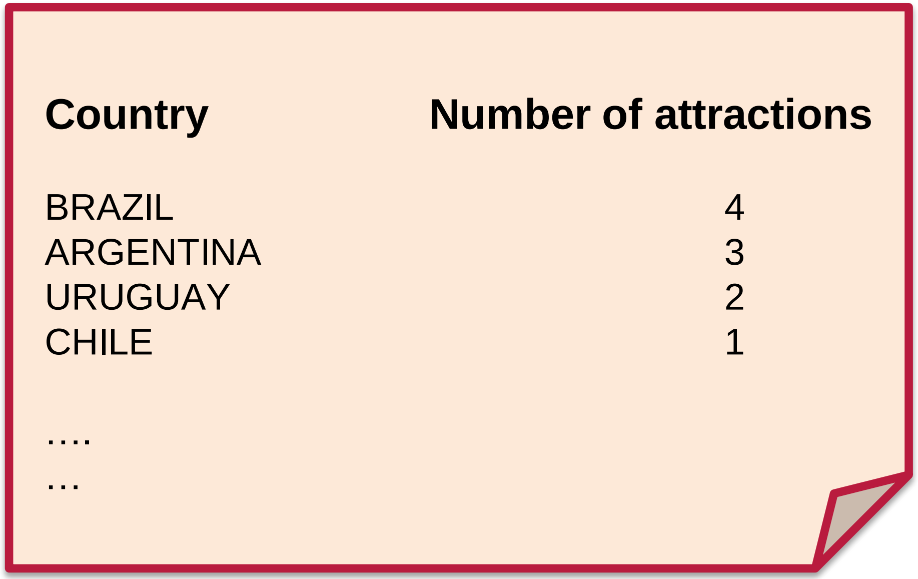
&OneCustomer: SDTCustomer

```
&OneCustomer.Id = 1
&OneCustomer.Name = 'John Smith'
&OneCustomer.Address = '5th. Avenue 1234'
```

Data Providers


Id: 1
Name: John Smith
Address: 5th. Avenue 1234
One customer
SDTCustomer
Id: 1
Name: John Smith
Address: 5th. Ave.
Id: 2
Name: Susan Brown
Address: 7th.Ave.
Id: 3
Name: Robert Hill
Address: 81th. St..
Id: 4
Name: Peter Jensen
Address: St,Paul Rd.
Customer Collection

Example: Ranking of attractions per country

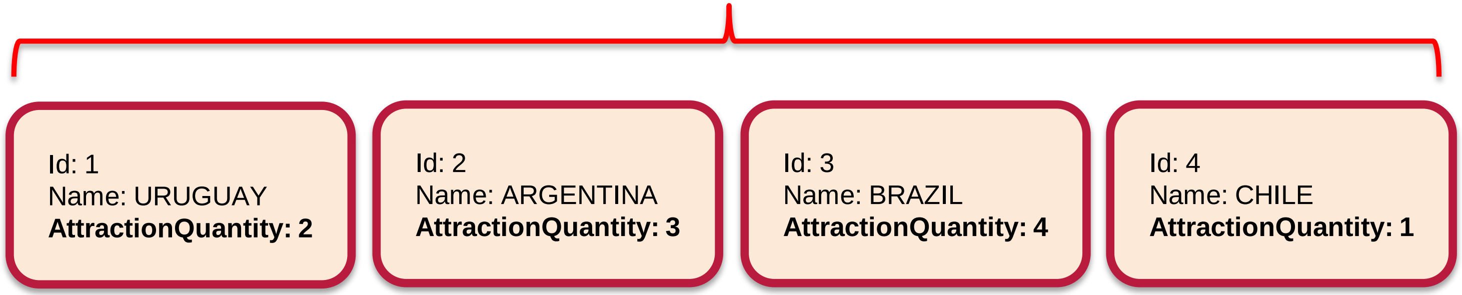


Country	**Number of attractions**
BRAZIL	4
ARGENTINA	3
URUGUAY	2
CHILE	1
….	
…	

Example: Ranking of attractions per country

SDTCountries

A collection of countries



Id: 1
Name: URUGUAY
AttractionQuantity: 2

Id: 2
Name: ARGENTINA
AttractionQuantity: 3

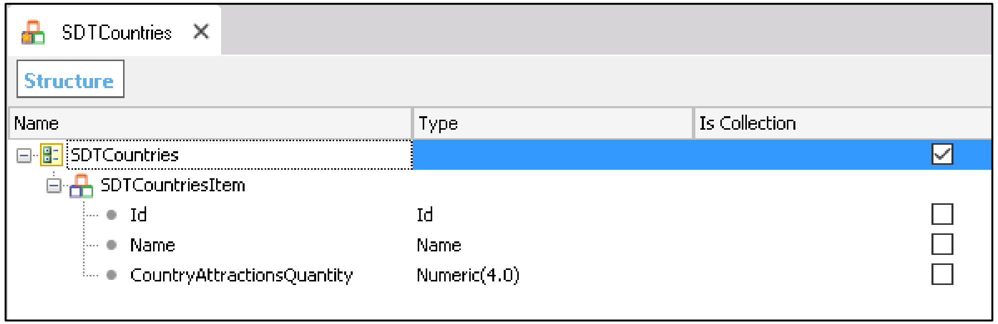
Id: 3
Name: BRAZIL
AttractionQuantity: 4

Id: 4
Name: CHILE
AttractionQuantity: 1

Example: Ranking of attractions per country

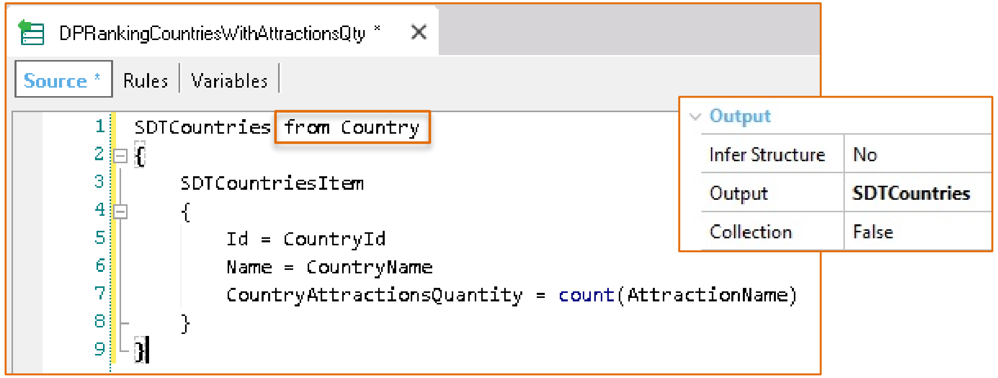
```
Country
{
   CountryId*
   CountryName
}
```

```
Attraction
{
   AttractionId*
   AttractionName
   CountryId
   CountryName
}
```


SDTCountries

Structure

Name	Type	Is Collection
SDTCountries		☑
SDTCountriesItem		
Id	Id	☐
Name	Name	☐
CountryAttractionsQuantity	Numeric(4.0)	☐


DPRankingCountriesWithAttractionsQty *

Source * | Rules | Variables

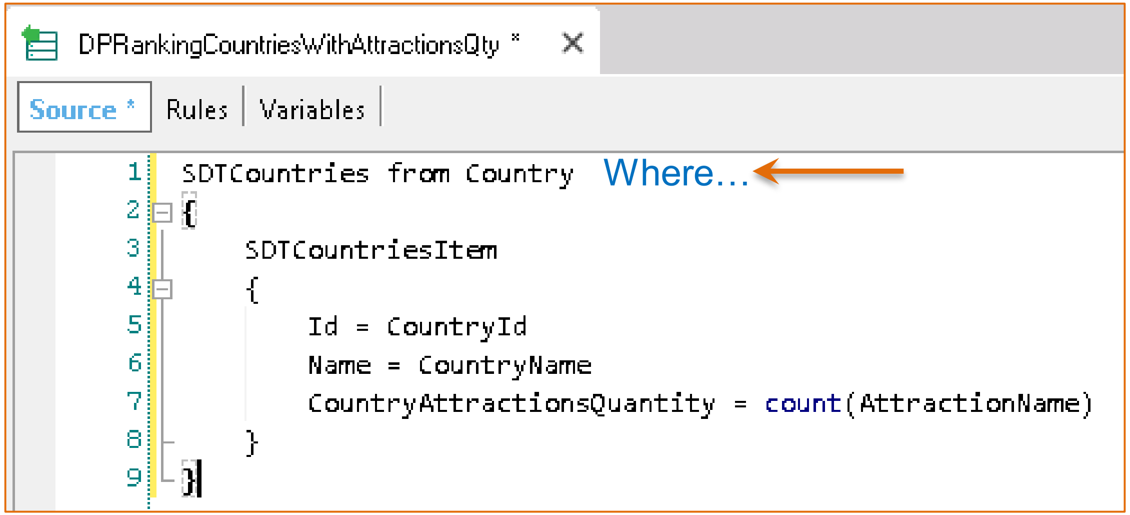
```
SDTCountries from Country
{
    SDTCountriesItem
    {
        Id = CountryId
        Name = CountryName
        CountryAttractionsQuantity = count(AttractionName)
    }
}
```

Output

Infer Structure	No
Output	SDTCountries
Collection	False

GeneXus

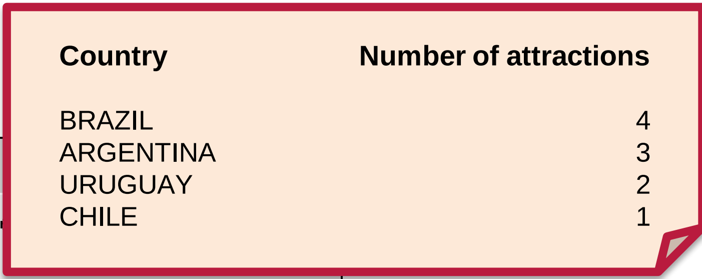
Example: Ranking of attractions per country



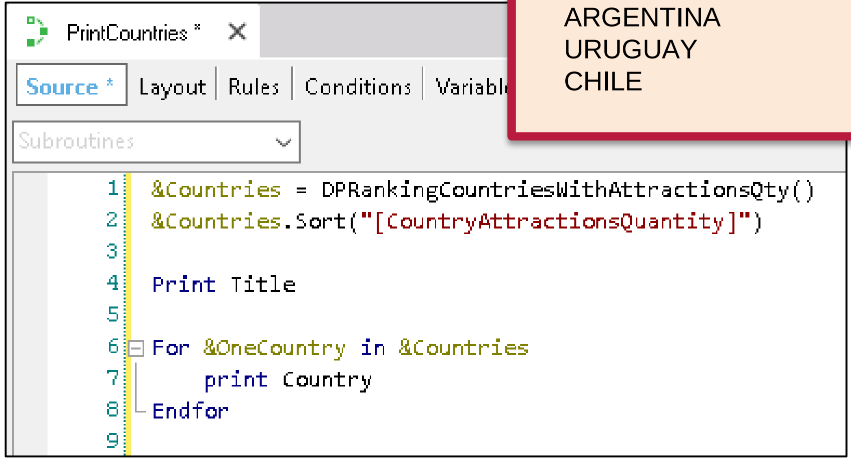
```
SDTCountries from Country
{
    SDTCountriesItem
    {
        Id = CountryId
        Name = CountryName
        CountryAttractionsQuantity = count(AttractionName)
    }
}
```

Example: Ranking of attractions per country

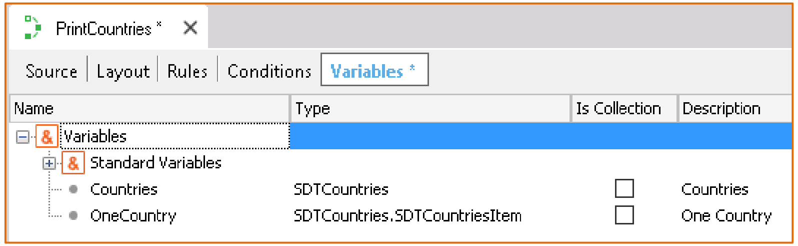
```
Country
{
   CountryId*
   CountryName
}

Attraction
{
   AttractionId*
   AttractionName
   CountryId
   CountryName
}
```



Country	Number of attractions
BRAZIL	4
ARGENTINA	3
URUGUAY	2
CHILE	1



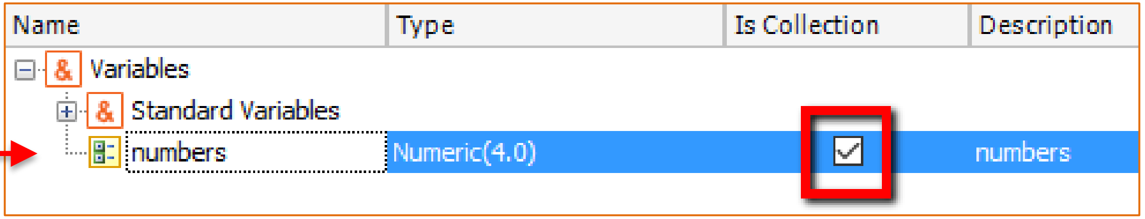
```
&Countries = DPRankingCountriesWithAttractionsQty()
&Countries.Sort("[CountryAttractionsQuantity]")

Print Title

For &OneCountry in &Countries
     print Country
Endfor

```



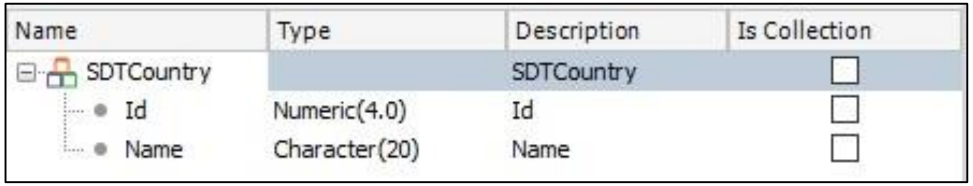
Name	Type	Is Collection	Description
Variables			
Standard Variables			
Countries	SDTCountries	☐	Countries
OneCountry	SDTCountries.SDTCountriesItem	☐	One Country

Collection Variables

&numbers

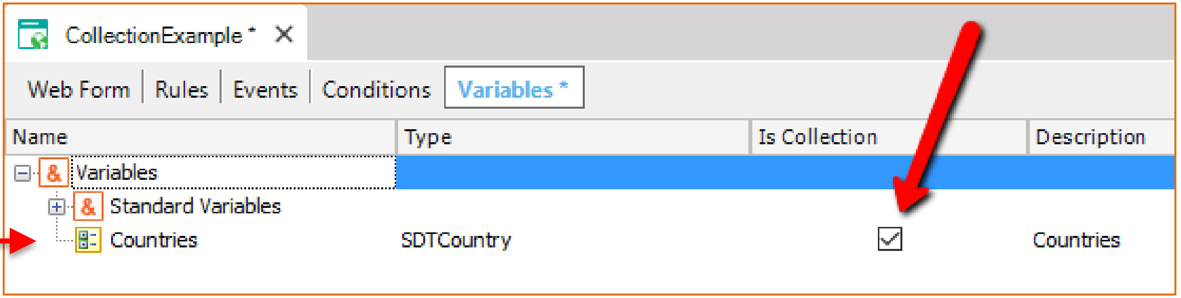


	Name	Type	Is Collection	Description
⊟	Variables			
⊞	Standard Variables			
	numbers	Numeric(4.0)	☑	numbers



Name	Type	Description	Is Collection
SDTCountry		SDTCountry	☐
Id	Numeric(4.0)	Id	☐
Name	Character(20)	Name	☐

&Countries



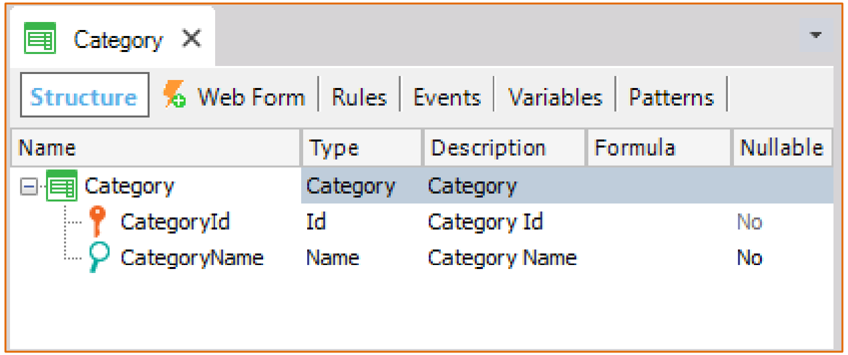
CollectionExample *

Web Form | Rules | Events | Conditions | Variables *

Name	Type	Is Collection	Description
Variables			
Standard Variables			
Countries	SDTCountry	☑	Countries

Business Components

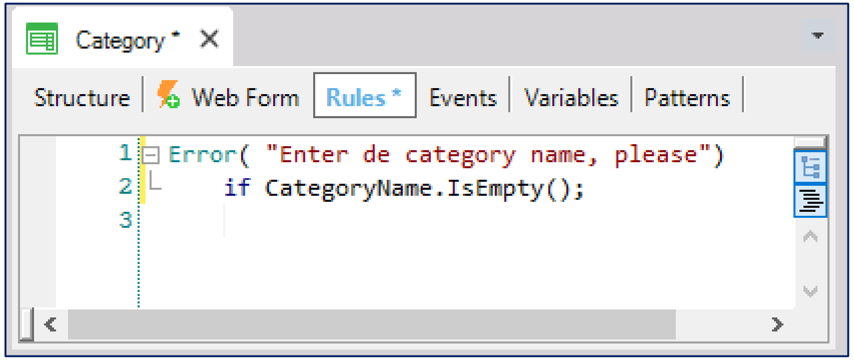
Concept: special data type based on a transaction



Category ✕

Structure | Web Form | Rules | Events | Variables | Patterns

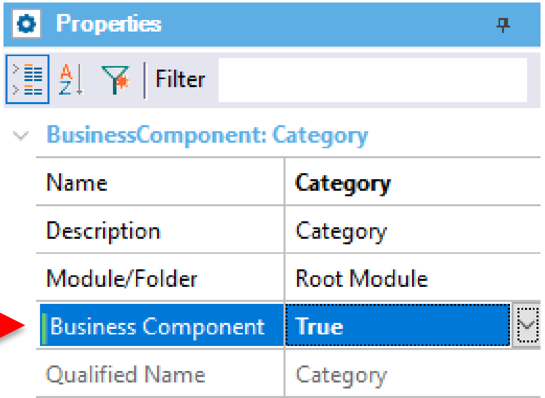
Name	Type	Description	Formula	Nullable
Category	Category	Category		
CategoryId	Id	Category Id		No
CategoryName	Name	Category Name		No



Category * ✕

Structure | Web Form | Rules * | Events | Variables | Patterns

```
Error( "Enter de category name, please")
    if CategoryName.IsEmpty();
```



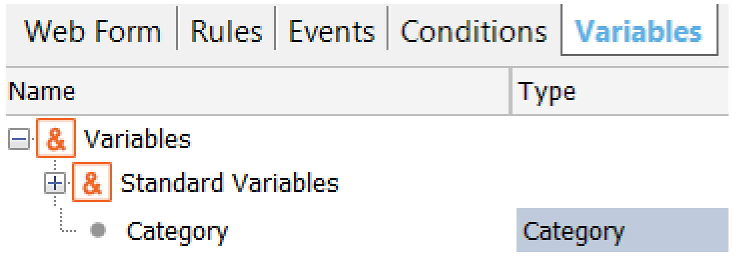
Properties

Filter

BusinessComponent: Category

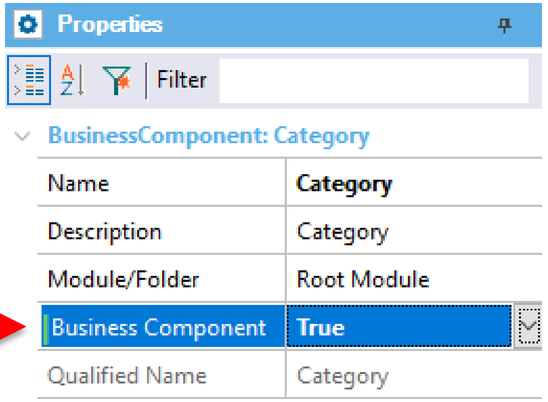
Name	**Category**
Description	Category
Module/Folder	Root Module
Business Component	True
Qualified Name	Category

Concept: special data type based on a transaction



Web Form | Rules | Events | Conditions | Variables

Name	Type
Variables	
Standard Variables	
Category	Category

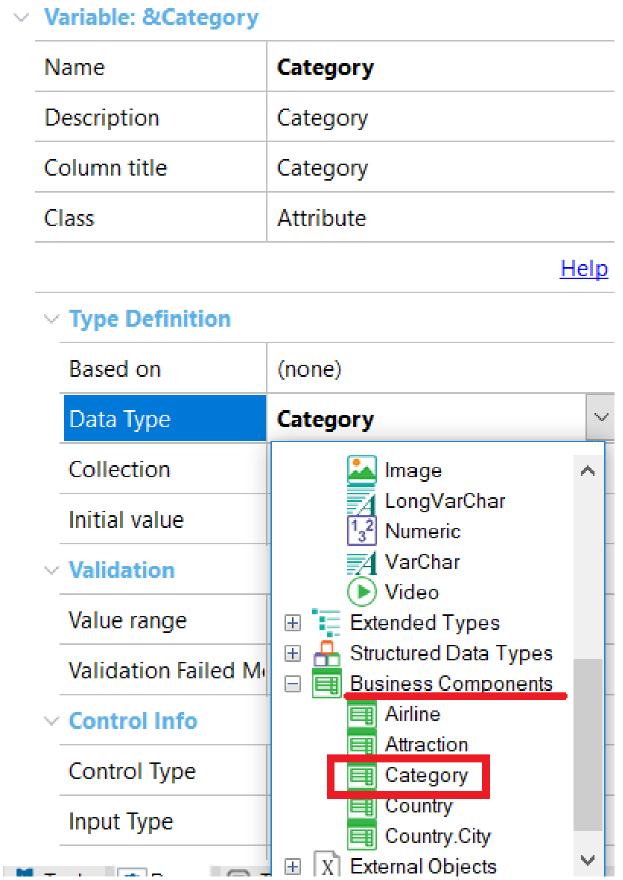


Properties

Filter

BusinessComponent: Category

Name	Category
Description	Category
Module/Folder	Root Module
Business Component	True
Qualified Name	Category



Variable: &Category

Name	Category
Description	Category
Column title	Category
Class	Attribute

Help

Type Definition

Based on	(none)
Data Type	Category
Collection	
Initial value	

Validation

Value range
Validation Failed M

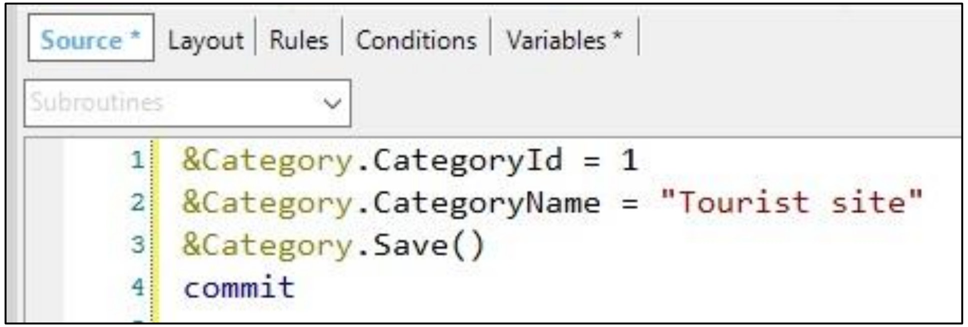
Control Info

Control Type
Input Type

Examples: insertion and modification

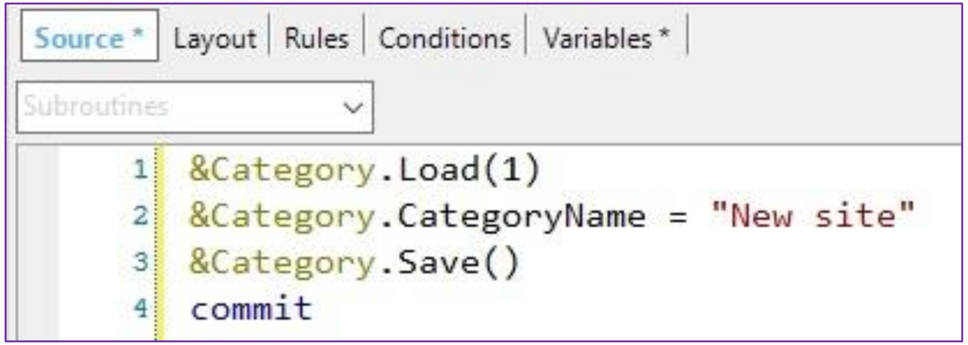
```
Category
{
   CategoryId*
   CategoryName
}
```

Insert



```
&Category.CategoryId = 1
&Category.CategoryName = "Tourist site"
&Category.Save()
commit
```

Update

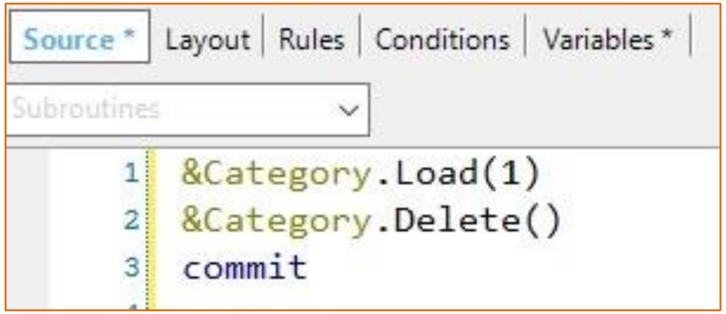


```
&Category.Load(1)
&Category.CategoryName = "New site"
&Category.Save()
commit
```

Example: deletion

Category
{
  CategoryId*
  CategoryName
}

Delete

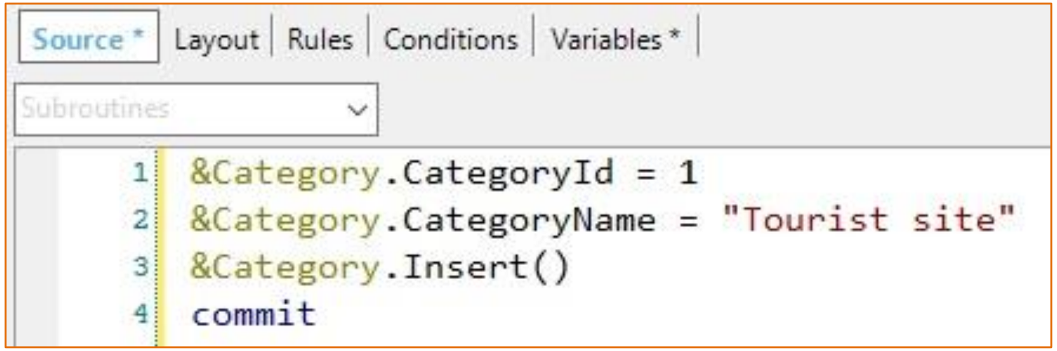


```
&Category.Load(1)
&Category.Delete()
commit
```

Insert and Update Methods

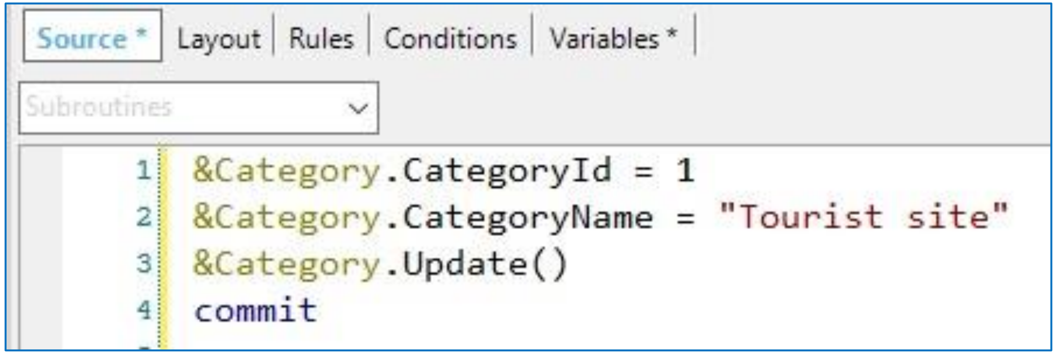
Category
{
  CategoryId*
  CategoryName
}

Insert



```
&Category.CategoryId = 1
&Category.CategoryName = "Tourist site"
&Category.Insert()
commit
```

Update

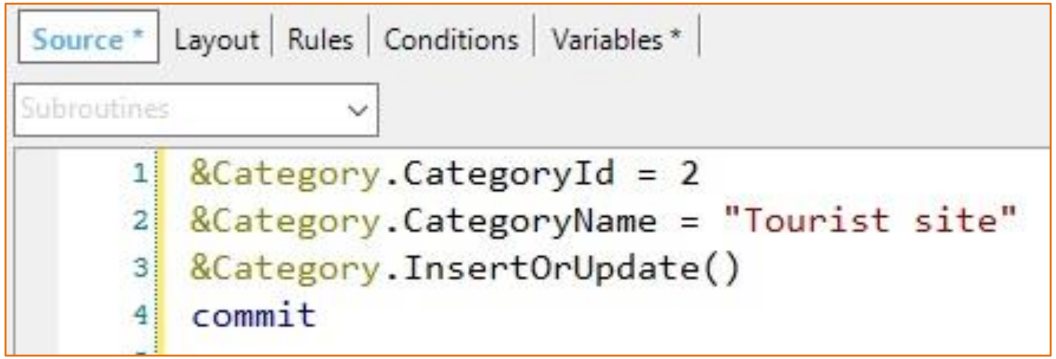


```
&Category.CategoryId = 1
&Category.CategoryName = "Tourist site"
&Category.Update()
commit
```

InsertOrUpdate Method

Category
{
  CategoryId*
  CategoryName
}

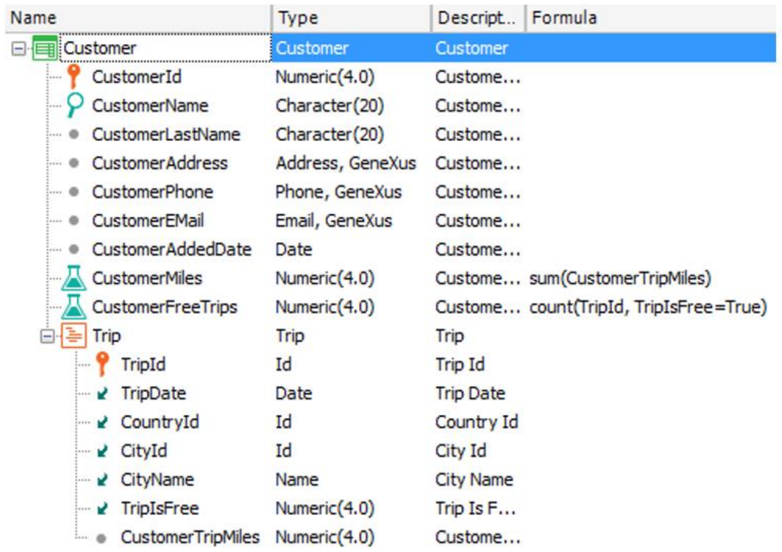
InsertOrUpdate



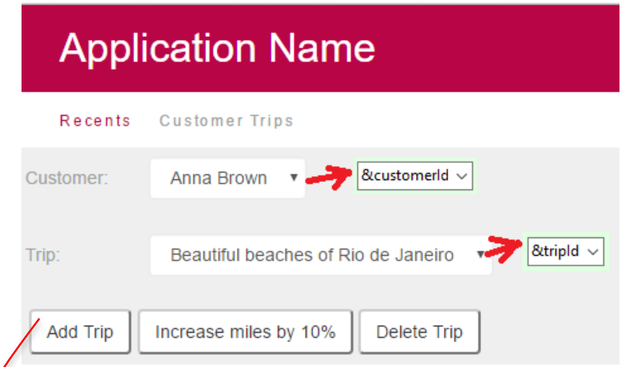
```
&Category.CategoryId = 2
&Category.CategoryName = "Tourist site"
&Category.InsertOrUpdate()
commit
```

Insert / Update in two-level transaction

Insert



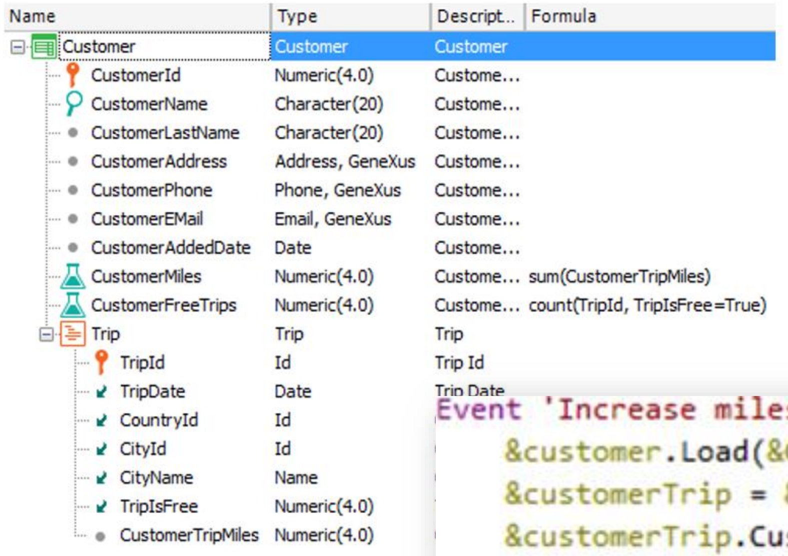
Name	Type	Descript...	Formula
Customer	Customer	Customer	
CustomerId	Numeric(4.0)	Custome...	
CustomerName	Character(20)	Custome...	
CustomerLastName	Character(20)	Custome...	
CustomerAddress	Address, GeneXus	Custome...	
CustomerPhone	Phone, GeneXus	Custome...	
CustomerEMail	Email, GeneXus	Custome...	
CustomerAddedDate	Date	Custome...	
CustomerMiles	Numeric(4.0)	Custome...	sum(CustomerTripMiles)
CustomerFreeTrips	Numeric(4.0)	Custome...	count(TripId, TripIsFree=True)
Trip	Trip	Trip	
TripId	Id	Trip Id	
TripDate	Date	Trip Date	
CountryId	Id	Country Id	
CityId	Id	City Id	
CityName	Name	City Name	
TripIsFree	Numeric(4.0)	Trip Is F...	
CustomerTripMiles	Numeric(4.0)	Custome...	



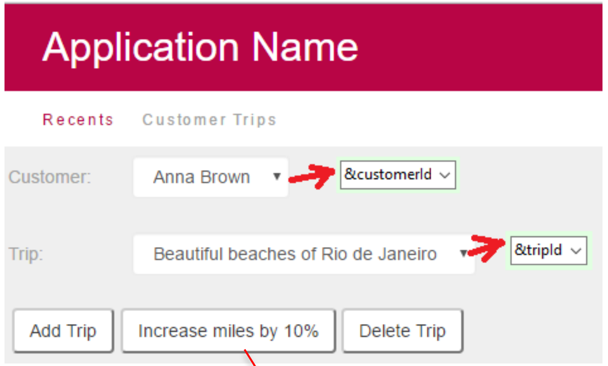
```
Event 'Add Trip'
    &customer.Load(&CustomerId)
    &customerTrip = new()
    &customerTrip.TripId = &TripId
    &customerTrip.CustomerTripMiles = 500
    &customer.Trip.Add(&customerTrip)
    &customer.Save()
    Commit
Endevent
```

Insert / Update in two-level transaction

Insert



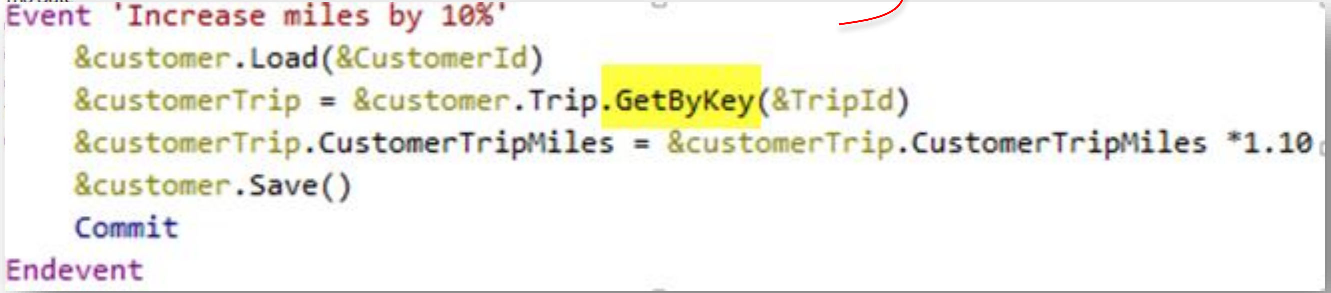
Name	Type	Descript...	Formula
Customer	Customer	Customer	
CustomerId	Numeric(4.0)	Custome...	
CustomerName	Character(20)	Custome...	
CustomerLastName	Character(20)	Custome...	
CustomerAddress	Address, GeneXus	Custome...	
CustomerPhone	Phone, GeneXus	Custome...	
CustomerEMail	Email, GeneXus	Custome...	
CustomerAddedDate	Date	Custome...	
CustomerMiles	Numeric(4.0)	Custome...	sum(CustomerTripMiles)
CustomerFreeTrips	Numeric(4.0)	Custome...	count(TripId, TripIsFree=True)
Trip	Trip	Trip	
TripId	Id	Trip Id	
TripDate	Date	Trip Date	
CountryId	Id		
CityId	Id		
CityName	Name		
TripIsFree	Numeric(4.0)		
CustomerTripMiles	Numeric(4.0)		



Application Name

Recents Customer Trips

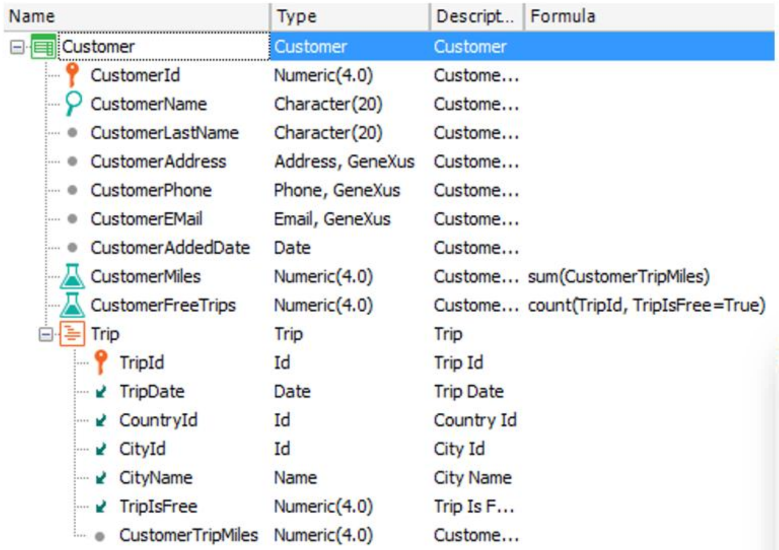
Customer:	Anna Brown	&customerId
Trip:	Beautiful beaches of Rio de Janeiro	&tripId



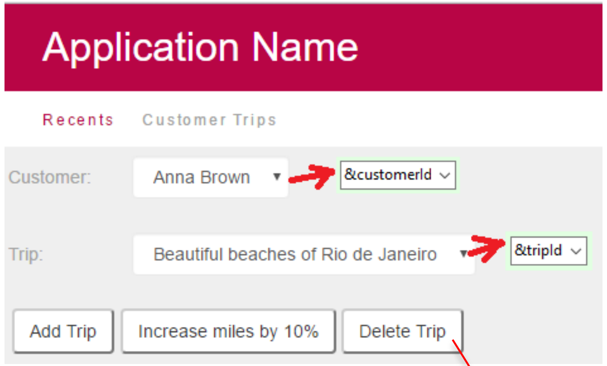
```
Event 'Increase miles by 10%'
    &customer.Load(&CustomerId)
    &customerTrip = &customer.Trip.GetByKey(&TripId)
    &customerTrip.CustomerTripMiles = &customerTrip.CustomerTripMiles *1.10
    &customer.Save()
    Commit
Endevent
```

Insert / Update in two-level transaction

Insert

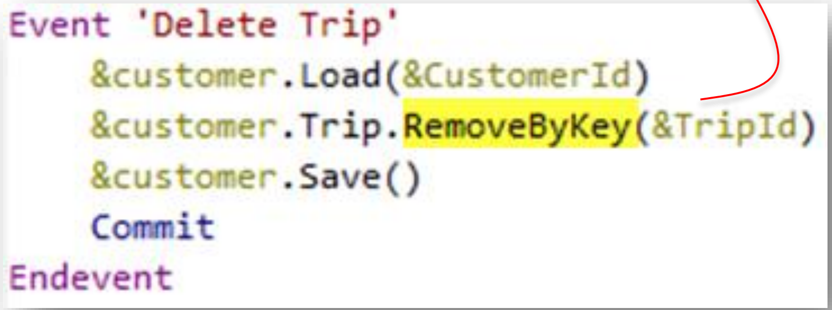


Name	Type	Descript...	Formula
Customer	Customer	Customer	
CustomerId	Numeric(4.0)	Custome...	
CustomerName	Character(20)	Custome...	
CustomerLastName	Character(20)	Custome...	
CustomerAddress	Address, GeneXus	Custome...	
CustomerPhone	Phone, GeneXus	Custome...	
CustomerEMail	Email, GeneXus	Custome...	
CustomerAddedDate	Date	Custome...	
CustomerMiles	Numeric(4.0)	Custome...	sum(CustomerTripMiles)
CustomerFreeTrips	Numeric(4.0)	Custome...	count(TripId, TripIsFree=True)
Trip	Trip	Trip	
TripId	Id	Trip Id	
TripDate	Date	Trip Date	
CountryId	Id	Country Id	
CityId	Id	City Id	
CityName	Name	City Name	
TripIsFree	Numeric(4.0)	Trip Is F...	
CustomerTripMiles	Numeric(4.0)	Custome...	



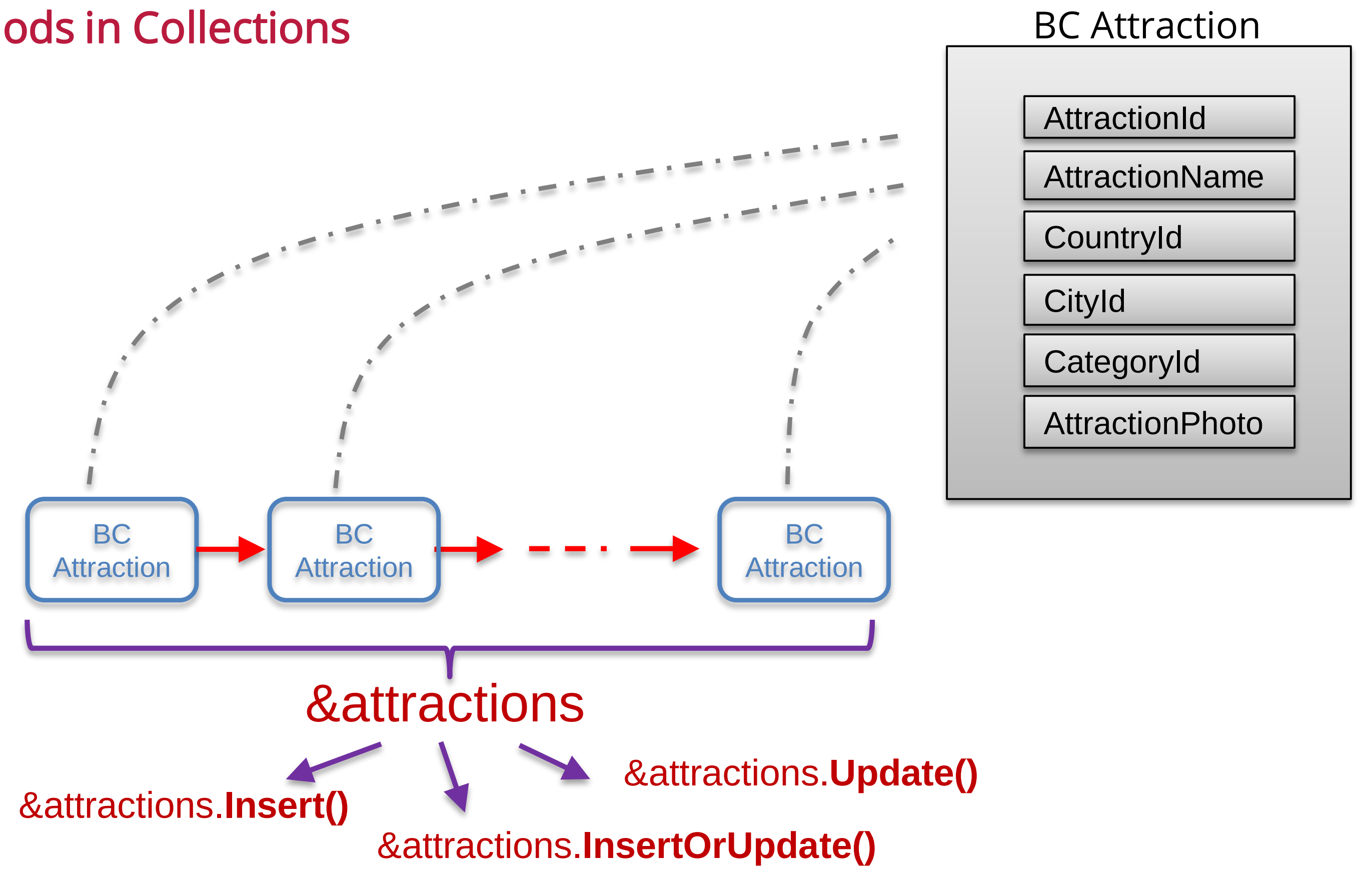
Customer: Anna Brown &customerId

Trip: Beautiful beaches of Rio de Janeiro &tripId



```
Event 'Delete Trip'
    &customer.Load(&CustomerId)
    &customer.Trip.RemoveByKey(&TripId)
    &customer.Save()
    Commit
Endevent
```

Methods in Collections



Insert / Update / InsertOrUpdate

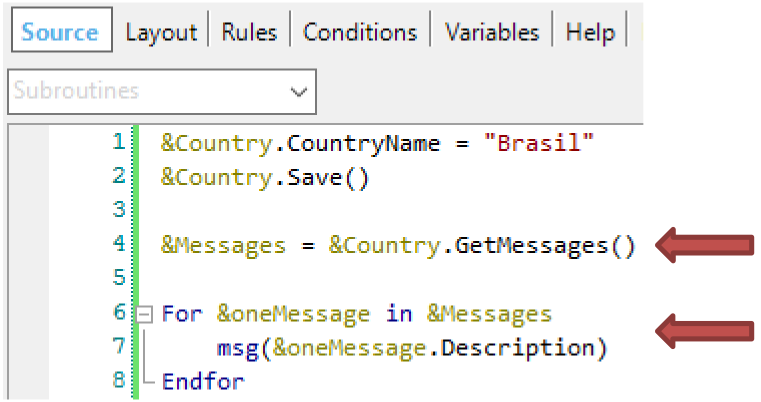
Using the methods *Insert*, *Update* and *InsertOrUpdate* is recommended because:

- When the *Load* and *Save* methods are used to make changes, the database is accessed twice, which reduces performance. With the *Update* or *InsertOrUpdate* methods, the database is accessed only once.

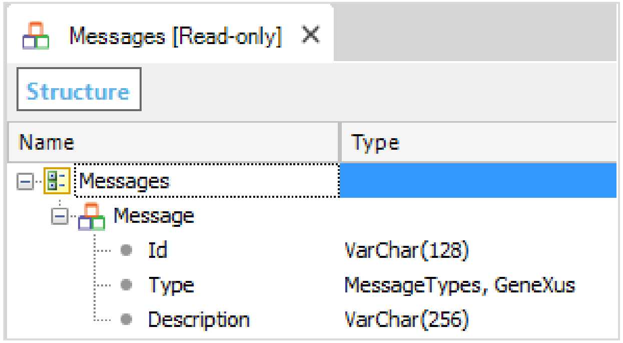
- The names of these new methods are self-explanatory about their purpose.

Error handling working with BC

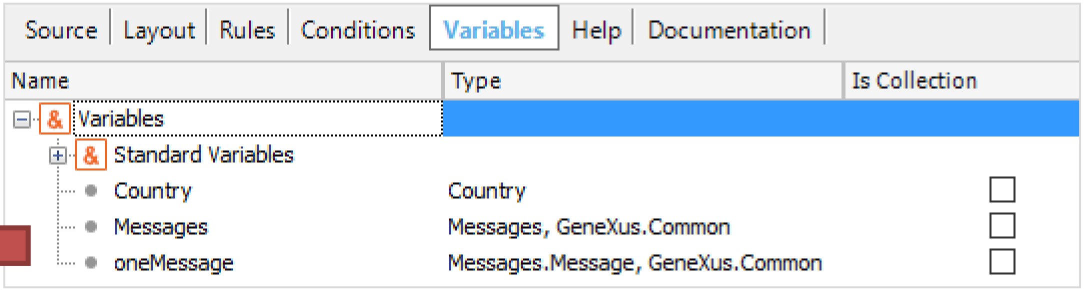
For each Business Component variable, a collection is loaded in memory with all the warning or error messages resulting from operations.



```
&Country.CountryName = "Brasil"
&Country.Save()

&Messages = &Country.GetMessages()

For &oneMessage in &Messages
    msg(&oneMessage.Description)
Endfor
```



Name	Type
Messages	
Message	
Id	VarChar(128)
Type	MessageTypes, GeneXus
Description	VarChar(256)



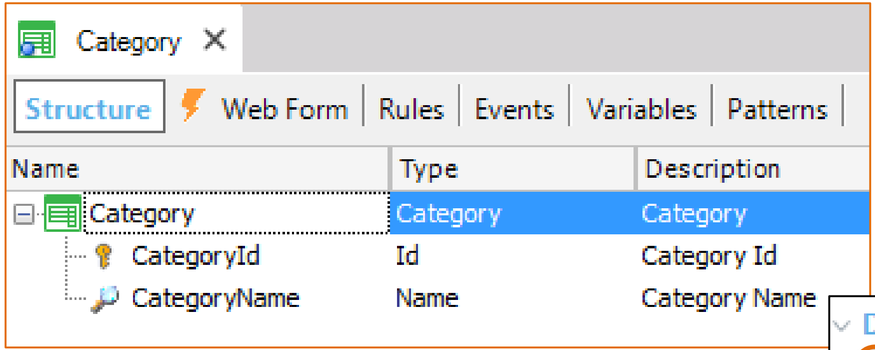
Name	Type	Is Collection
Variables		
Standard Variables		
Country	Country	☐
Messages	Messages, GeneXus.Common	☐
oneMessage	Messages.Message, GeneXus.Common	☐

Data Population

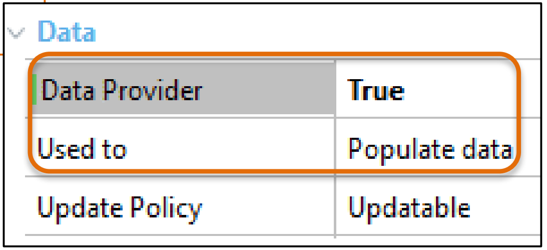
Transaction

Initializing data automatically

GeneXus makes it easy to define the data used to populate the physical tables that are created associated with transactions, so as to avoid resorting to other means to load data.

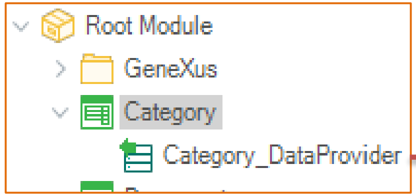


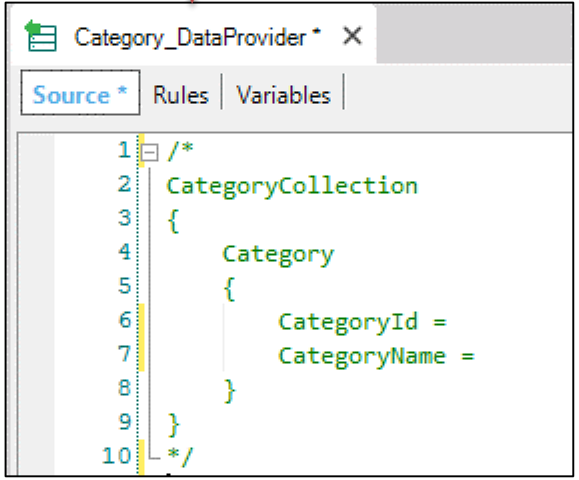
Name	Type	Description
Category	Category	Category
CategoryId	Id	Category Id
CategoryName	Name	Category Name



Data Provider	True
Used to	Populate data
Update Policy	Updatable

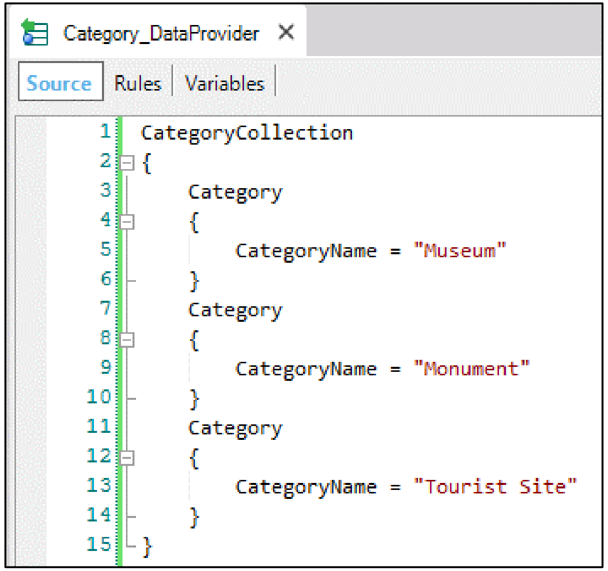
Initializing data





```
/*
CategoryCollection
{
    Category
    {
        CategoryId =
        CategoryName =
    }
}
*/
```

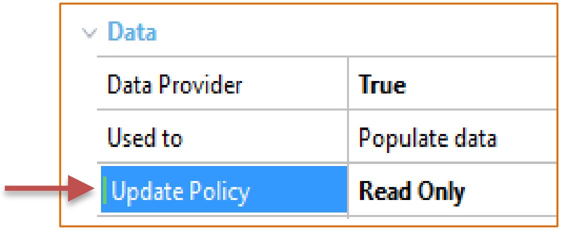
The CategoryId is not loaded
because it has been set as autonumbered



```
CategoryCollection
{
    Category
    {
        CategoryName = "Museum"
    }
    Category
    {
        CategoryName = "Monument"
    }
    Category
    {
        CategoryName = "Tourist Site"
    }
}
```

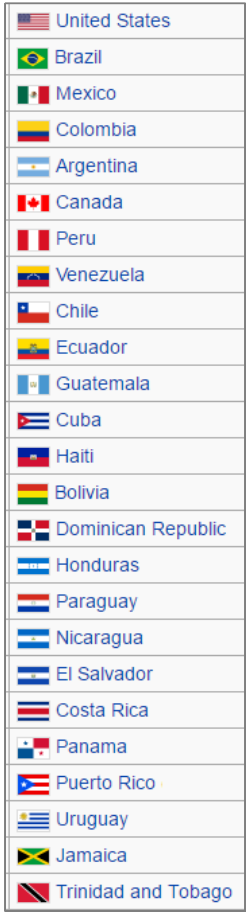
Initializing data: Read-only

```
Country
{
   CountryId*
   CountryName
}
```



Data

Data Provider	True
Used to	Populate data
Update Policy	Read Only



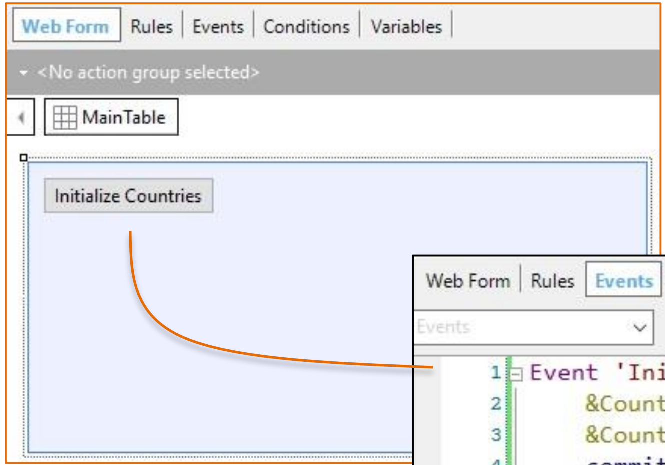
United States
Brazil
Mexico
Colombia
Argentina
Canada
Peru
Venezuela
Chile
Ecuador
Guatemala
Cuba
Haiti
Bolivia
Dominican Republic
Honduras
Paraguay
Nicaragua
El Salvador
Costa Rica
Panama
Puerto Rico
Uruguay
Jamaica
Trinidad and Tobago

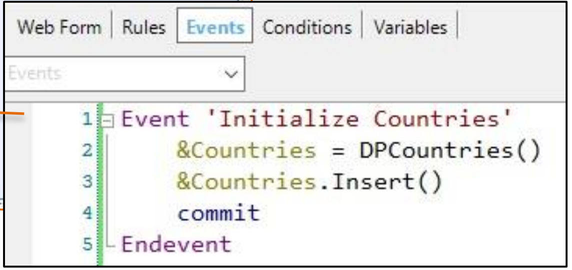
Data Population

Business Components and Data Providers

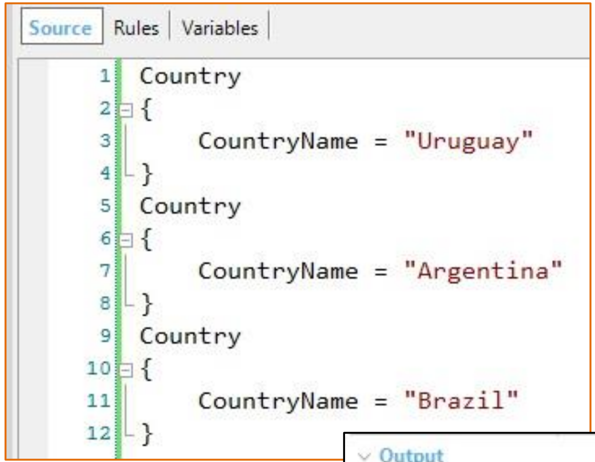
Example

Country
{
  CountryId* ← Autonumber = True
  CountryName
}

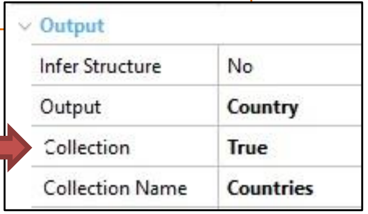




```
Event 'Initialize Countries'
    &Countries = DPCountries()
    &Countries.Insert()
    commit
Endevent
```

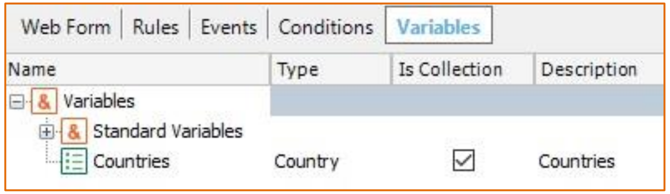


```
Country
{
    CountryName = "Uruguay"
}
Country
{
    CountryName = "Argentina"
}
Country
{
    CountryName = "Brazil"
}
```



Output

Infer Structure	No
Output	Country
Collection	True
Collection Name	Countries



Name	Type	Is Collection	Description
Variables			
Standard Variables			
Countries	Country	☑	Countries

Data Population with Procedures
New / For Each / Delete Commands

Notes

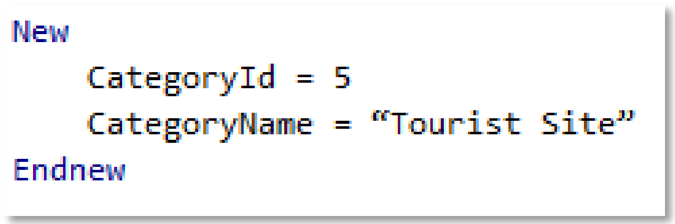
The commands New / For Each / Delete allow inserting, updating and deleting data from the database, but only can be used in Procedures

Even though the following commands allow inserting, updating and deleting data from the database, using a Business Component is recommended because they:

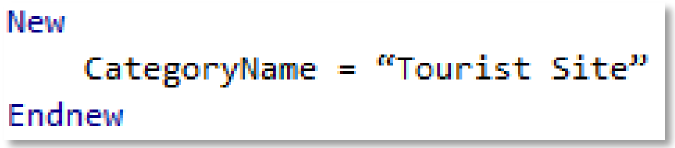
- Control referential integrity
- Trigger the rules declared in the transaction

Insertion – NEW Command

```
Category
{
  CategoryId*
  CategoryName
}
```



```
New
    CategoryId = 5
    CategoryName = "Tourist Site"
Endnew
```

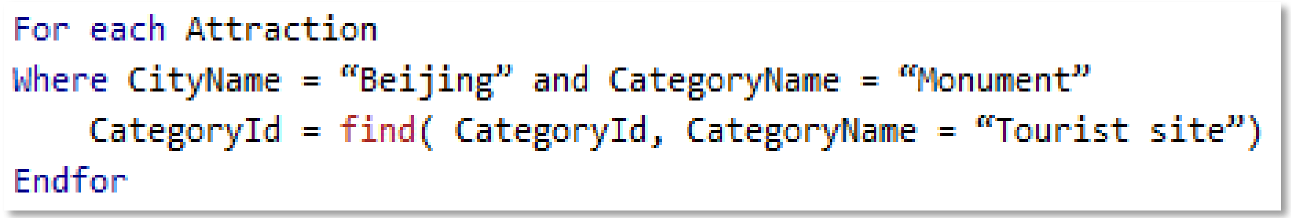


```
New
    CategoryName = "Tourist Site"
Endnew
```

If the attribute is autonumbered it doesn't have to be inserted

GeneXus™

Modification / FOR EACH Command



```
For each Attraction
Where CityName = "Beijing" and CategoryName = "Monument"
    CategoryId = find( CategoryId, CategoryName = "Tourist site")
Endfor
```

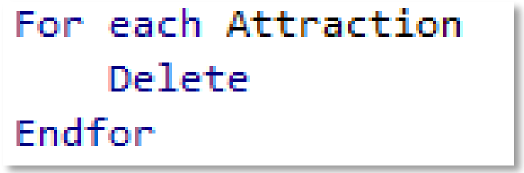
Category
```
{
  CategoryId*
  CategoryName
}
```

Country
```
{
  CountryId*
  CountryName
  City
  {
    CityId*
    CityName
  }
}
```

Attraction
```
{
  AttractionId*
  AttractionName
  CategoryId
  CategoryName
  CountryId
  CountryName
  CityId
  CitlyName
}
```

Deletion – DELETE Command

Attraction
{
  AttractionId*
  AttractionName
  CategoryId
  CategoryName
  CountryId
  CountryName
  CityId
  CitlyName
}

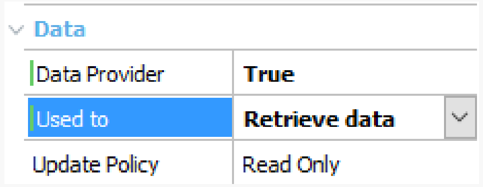


```
For each Attraction
    Delete
Endfor
```

Dynamic Transactions

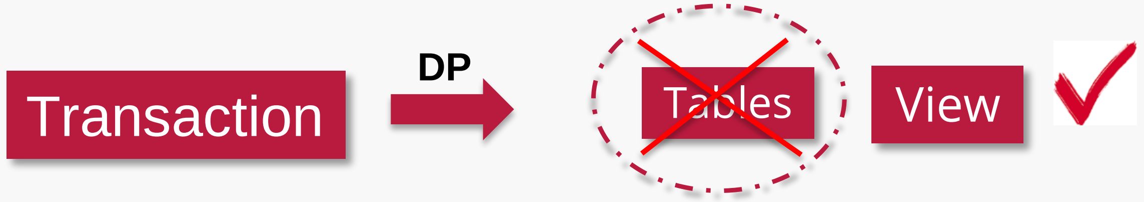
Dynamic Transactions

1. **Data Provider: True**
2. **Used to: Retrieve data**
3. **Update Policy:** **- Read Only**
   - Updatable



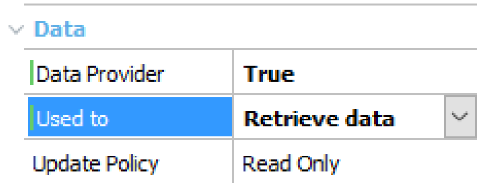
Data

Data Provider	True
Used to	Retrieve data
Update Policy	Read Only



Dynamic Transactions to retrieve data

- In the Data Provider, we need to indicate what data we want to retrieve
- A dynamic transaction can be referenced as Base Trn

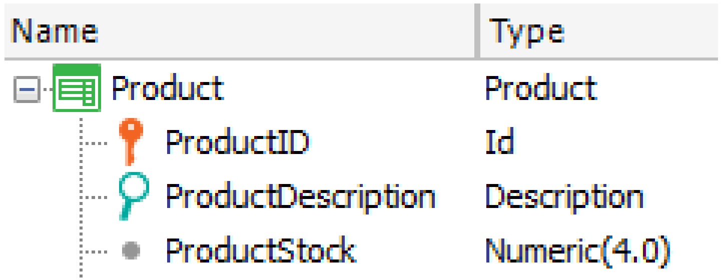


Data

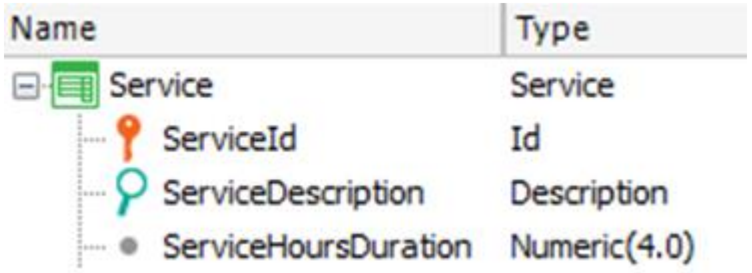
Data Provider	**True**
Used to	**Retrieve data**
Update Policy	Read Only

Example 1: Data join

A company sells products and provides services.
A list is to be issued containing everything that the company offers in alphabetical order.

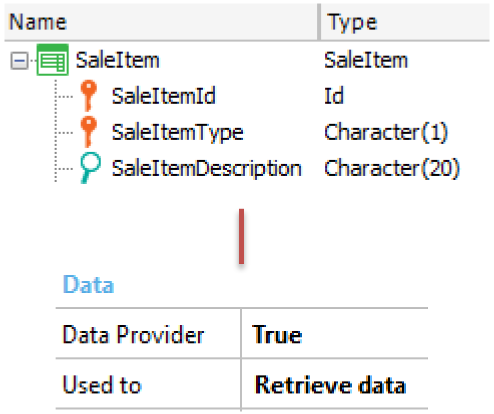


Name	Type
Product	Product
ProductID	Id
ProductDescription	Description
ProductStock	Numeric(4.0)



Name	Type
Service	Service
ServiceId	Id
ServiceDescription	Description
ServiceHoursDuration	Numeric(4.0)

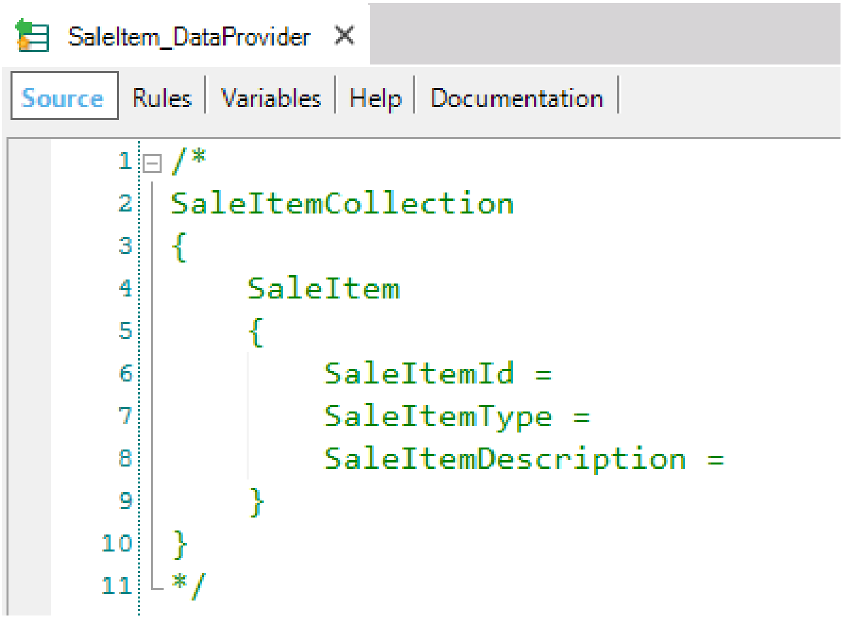
Example: DataProvider generated



Name	Type
SaleItem	SaleItem
SaleItemId	Id
SaleItemType	Character(1)
SaleItemDescription	Character(20)

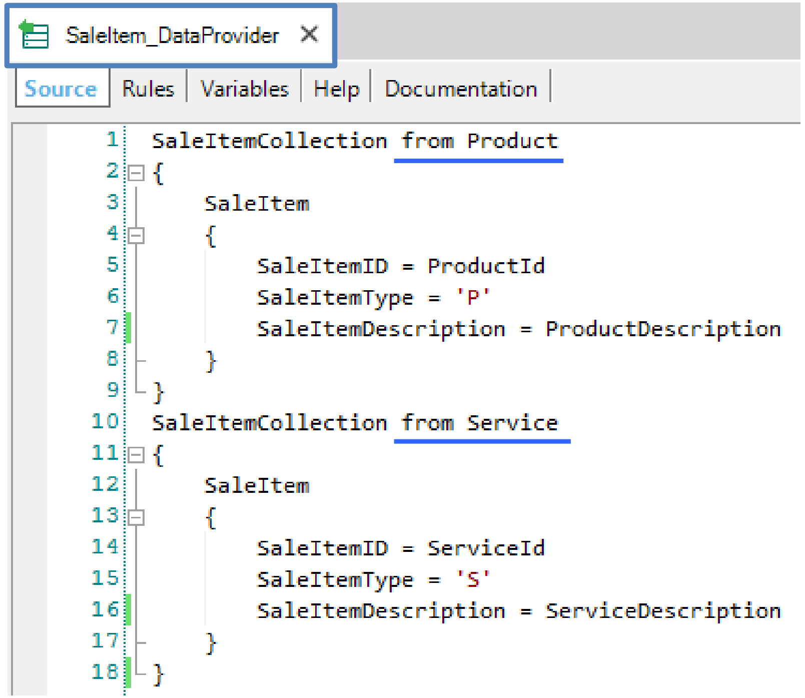
Data

Data Provider	**True**
Used to	**Retrieve data**



```
/*
SaleItemCollection
{
    SaleItem
    {
        SaleItemId =
        SaleItemType =
        SaleItemDescription =
    }
}
*/
```

Example: DataProvider



```
SaleItemCollection from Product
{
    SaleItem
    {
        SaleItemID = ProductId
        SaleItemType = 'P'
        SaleItemDescription = ProductDescription
    }
}
SaleItemCollection from Service
{
    SaleItem
    {
        SaleItemID = ServiceId
        SaleItemType = 'S'
        SaleItemDescription = ServiceDescription
    }
}
```

Example: SaleItem specification

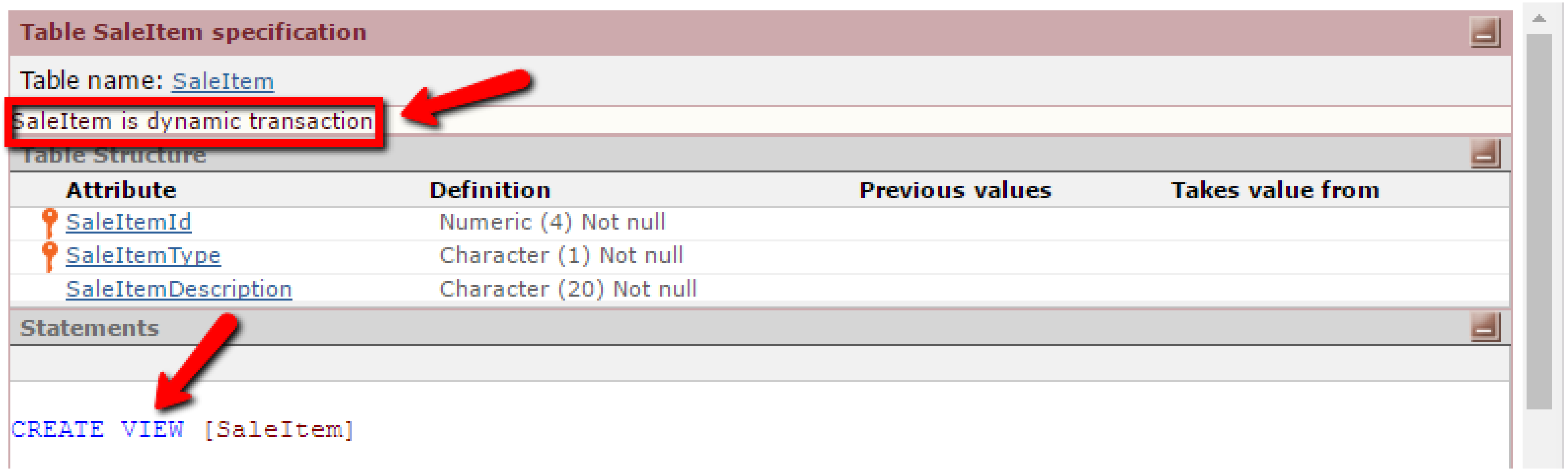


Table SaleItem specification

Table name: SaleItem

SaleItem is dynamic transaction

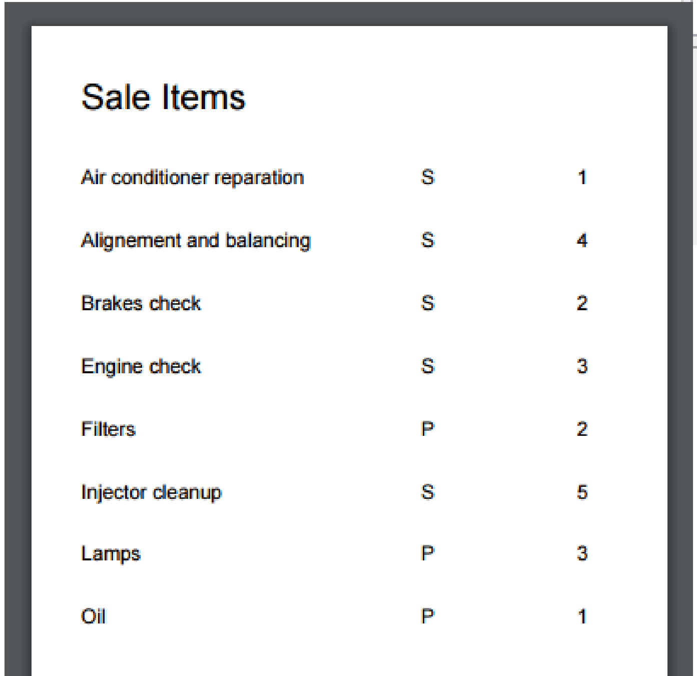
Table Structure

	Attribute	Definition	Previous values	Takes value from
🔑	SaleItemId	Numeric (4) Not null		
🔑	SaleItemType	Character (1) Not null		
	SaleItemDescription	Character (20) Not null		

Statements

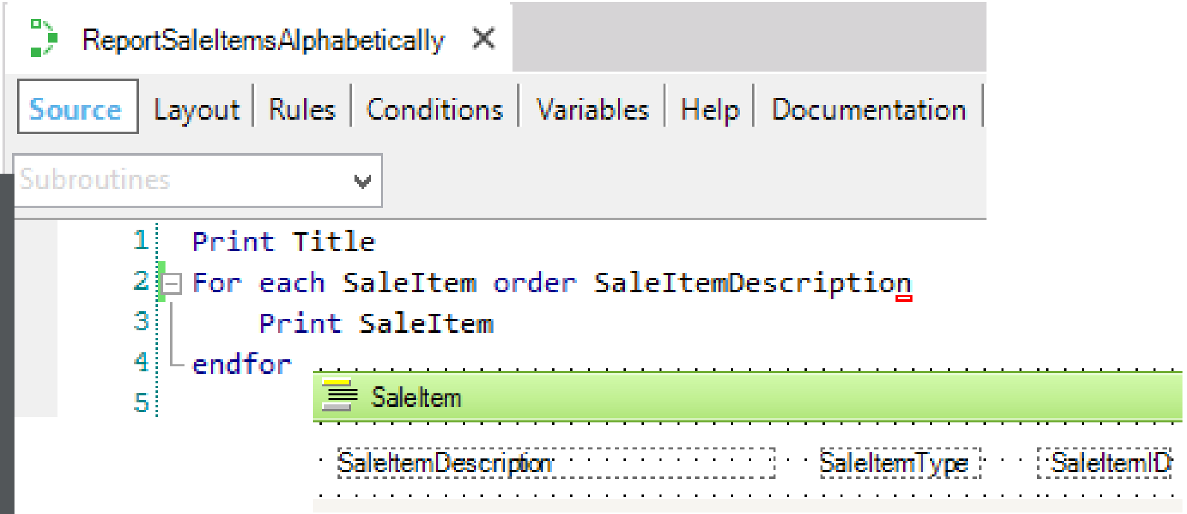
```
CREATE VIEW [SaleItem]
```

Example: PDF list



Sale Items

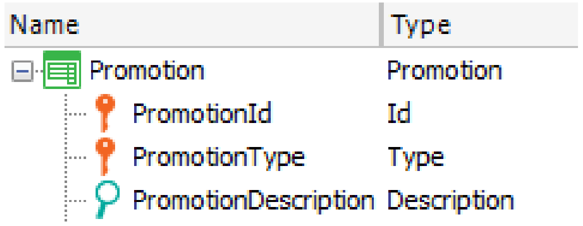
Air conditioner reparation	S	1
Alignement and balancing	S	4
Brakes check	S	2
Engine check	S	3
Filters	P	2
Injector cleanup	S	5
Lamps	P	3
Oil	P	1



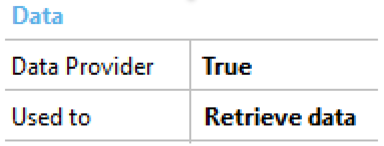
```
Print Title
For each SaleItem order SaleItemDescription
    Print SaleItem
endfor
```

Example 2: Modeling reality

- Products → stock > 1000
- Services → < 10 contrataciones

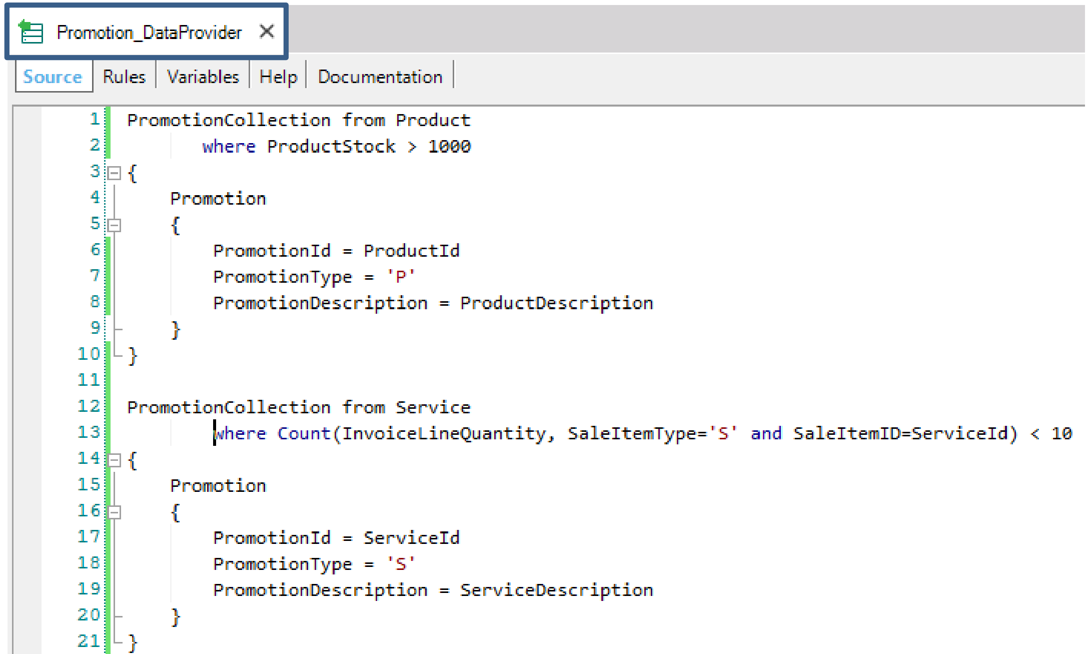


Name	Type
Promotion	Promotion
PromotionId	Id
PromotionType	Type
PromotionDescription	Description

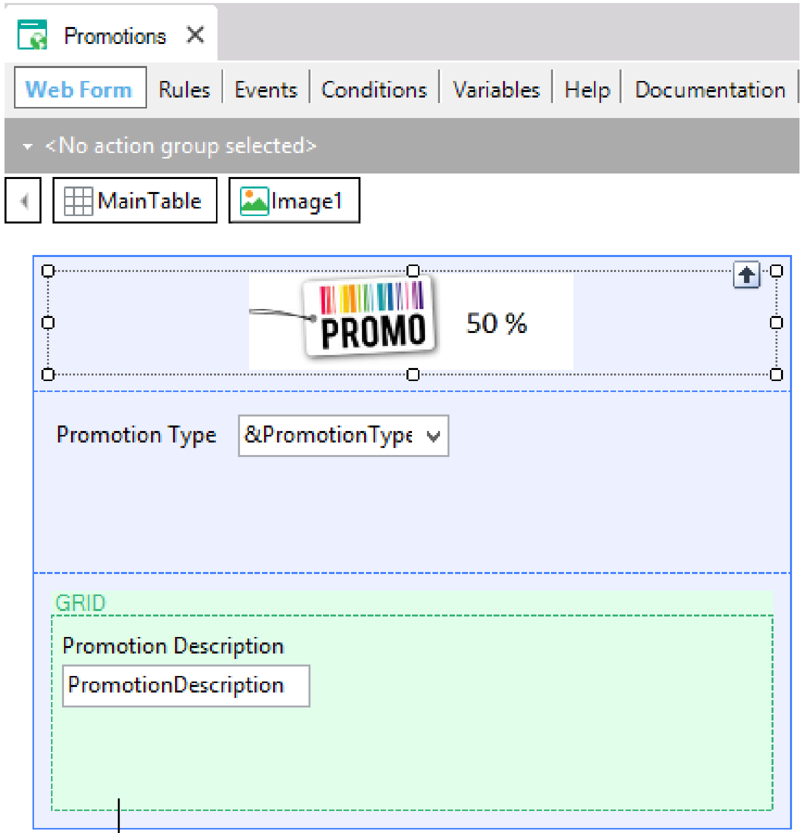


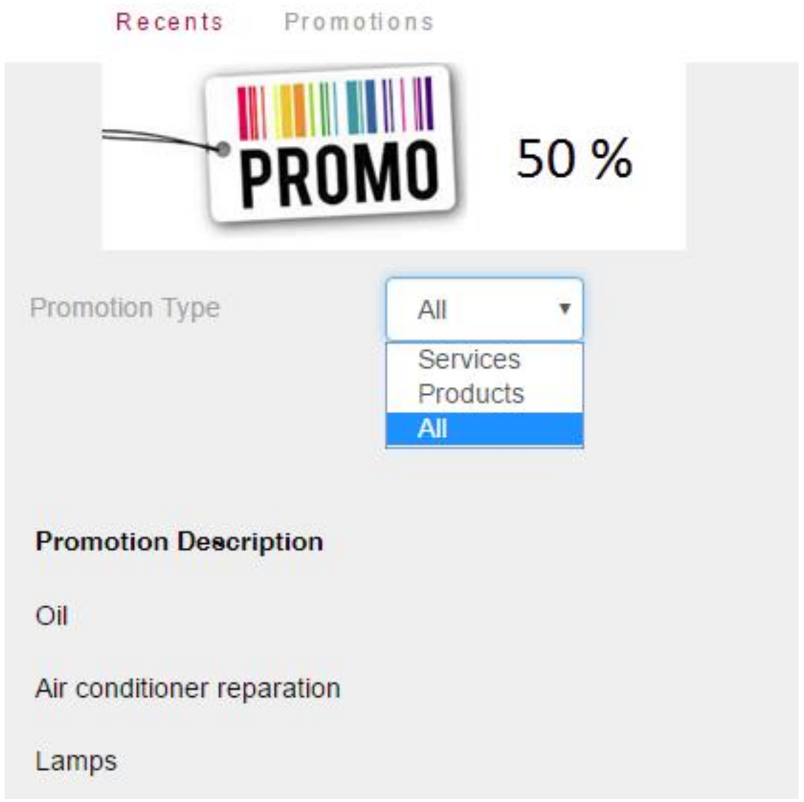
Data

Data Provider	True
Used to	Retrieve data



```
PromotionCollection from Product
        where ProductStock > 1000
{
    Promotion
    {
        PromotionId = ProductId
        PromotionType = 'P'
        PromotionDescription = ProductDescription
    }
}

PromotionCollection from Service
        where Count(InvoiceLineQuantity, SaleItemType='S' and SaleItemID=ServiceId) < 10
{
    Promotion
    {
        PromotionId = ServiceId
        PromotionType = 'S'
        PromotionDescription = ServiceDescription
    }
}
```

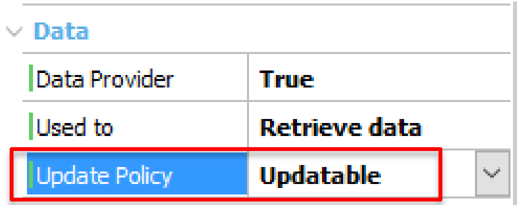




Conditions	**PromotionType=&PromotionType when &PromotionType <> 'A';**

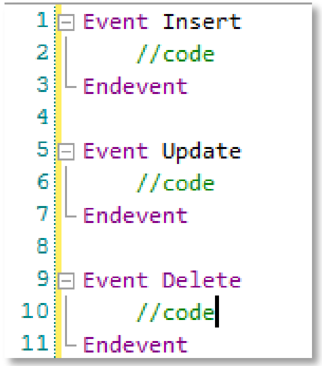
Dynamic Transactions to update data

How do we update data if we don’t have a table associated with the transaction?



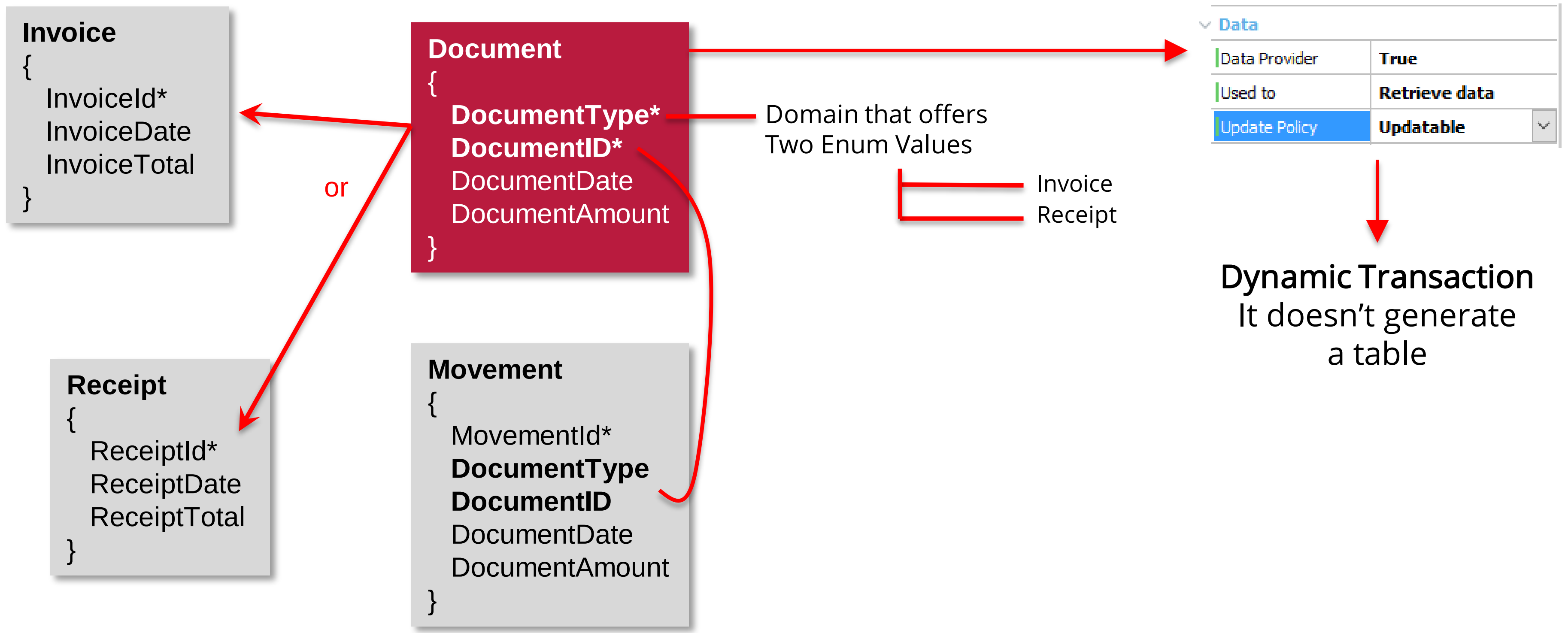
Data

Data Provider	True
Used to	Retrieve data
Update Policy	Updatable



```
Event Insert
      //code
Endevent

Event Update
      //code
Endevent

Event Delete
      //code
Endevent
```

The developer will have to program the events **Insert**, **Update** and **Delete**.



```
Invoice
{
   InvoiceId*
   InvoiceDate
   InvoiceTotal
}
```

or

```
Receipt
{
   ReceiptId*
   ReceiptDate
   ReceiptTotal
}
```

```
Document
{
   DocumentType*
   DocumentID*
   DocumentDate
   DocumentAmount
}
```

```
Movement
{
   MovementId*
   DocumentType
   DocumentID
   DocumentDate
   DocumentAmount
}
```

Domain that offers Two Enum Values

- Invoice
- Receipt

Data	
Data Provider	True
Used to	Retrieve data
Update Policy	Updatable

Dynamic Transaction
It doesn't generate a table

Example 3: Using Dynamic Transactions to update data

```
Document
{
   DocumentType*
   DocumentID*
   DocumentDate
   DocumentAmount
}
```

Events

```
Event Insert
  If DocumentType = "Invoice"
    &invoice = new()
    &invoice.InvoiceId = DocumentID
    &invoice.InvoiceDate = DocumentDate
    &invoce.InvoiceTotal = DocumentAmount
    &inovice.Insert()
  else
    &receipt = new()
    &receipt.ReceiptId = DocumentId
    &receipt.ReceiptDate = DocumentDate
    &receipt.ReceiptTotal = DocumentAmount
    &receipt.Insert()
  endif
endevent
```

&invoice → BC Invoice

&receipt → BC Receipt

Transactional Integrity

Concepts

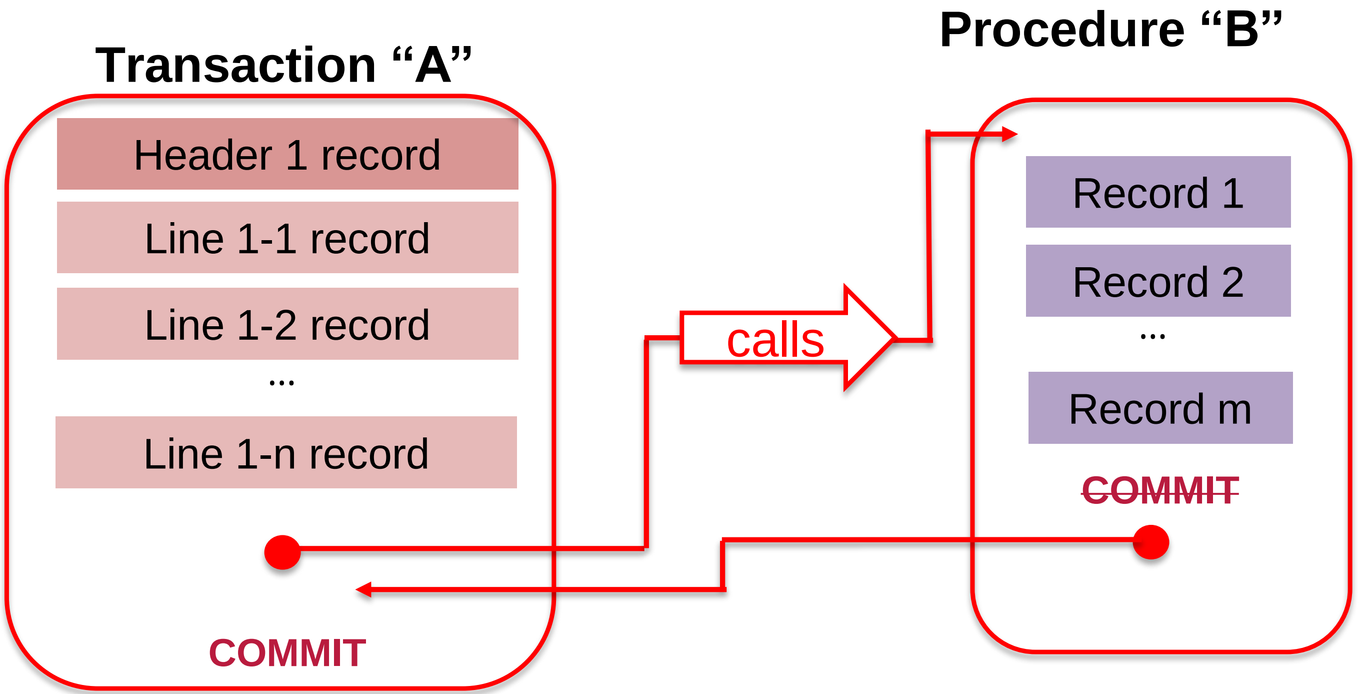
- A set of updates to the database has **transactional integrity** when in case of an "abnormal" termination the database remains in a **consistent state.**

- **Consistency** at this point is determined by Logical Work Units (LWU): operations on the database performed between two **Commit** operations.

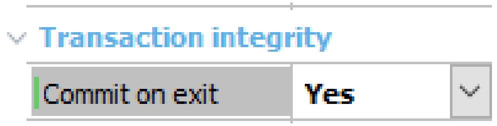
- **Transactions and Procedures** → At the end, GeneXus automatically writes the **Commit** command in the generated programs**.** The object can be disabled through the **Commit on Exit** ("Yes", "No") property of the object.

- **Business Component** → GeneXus doesn't write the Commit.

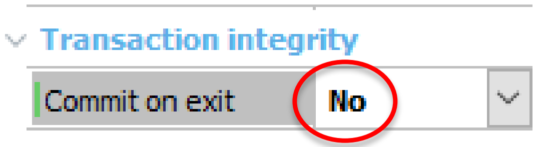
Customizing LUWs



B (parm$_1$, … , parm$_n$) **on BeforeComplete**;

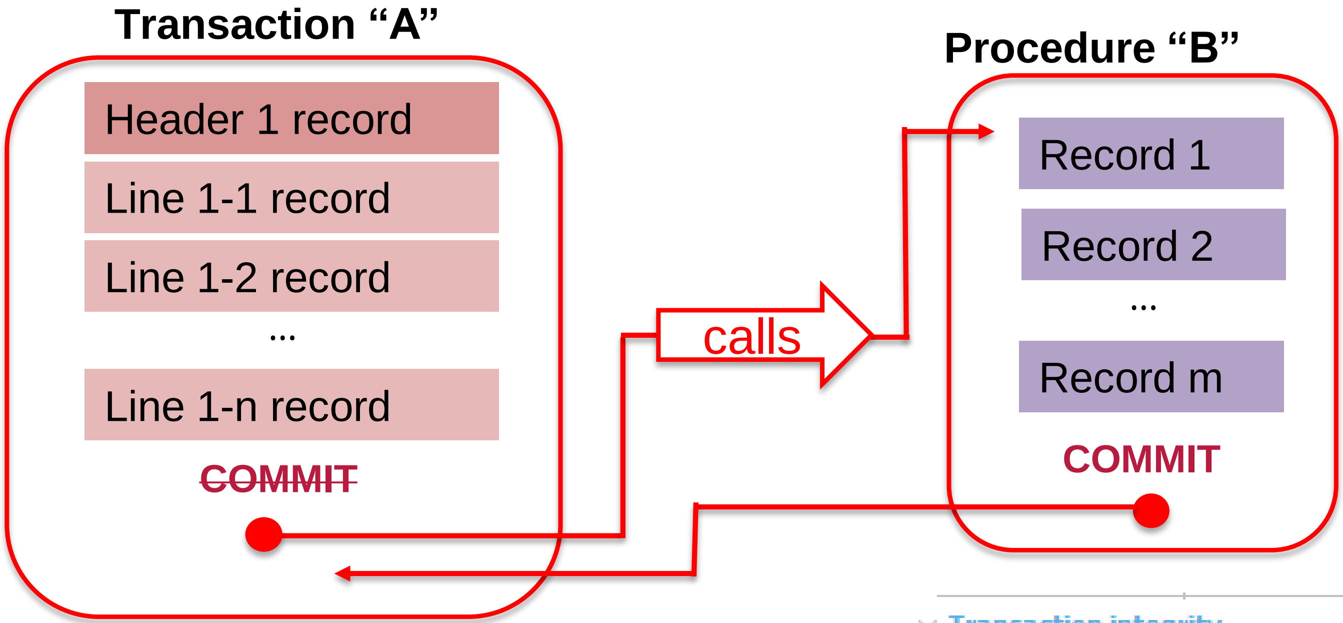

Transaction integrity

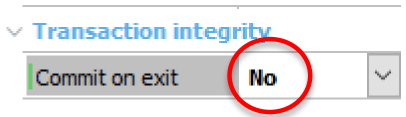
Commit on exit	Yes


Transaction integrity

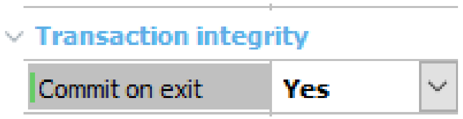
Commit on exit	[✓ No]

Customizing LUWs




Transaction integrity

Commit on exit	[✓ No]

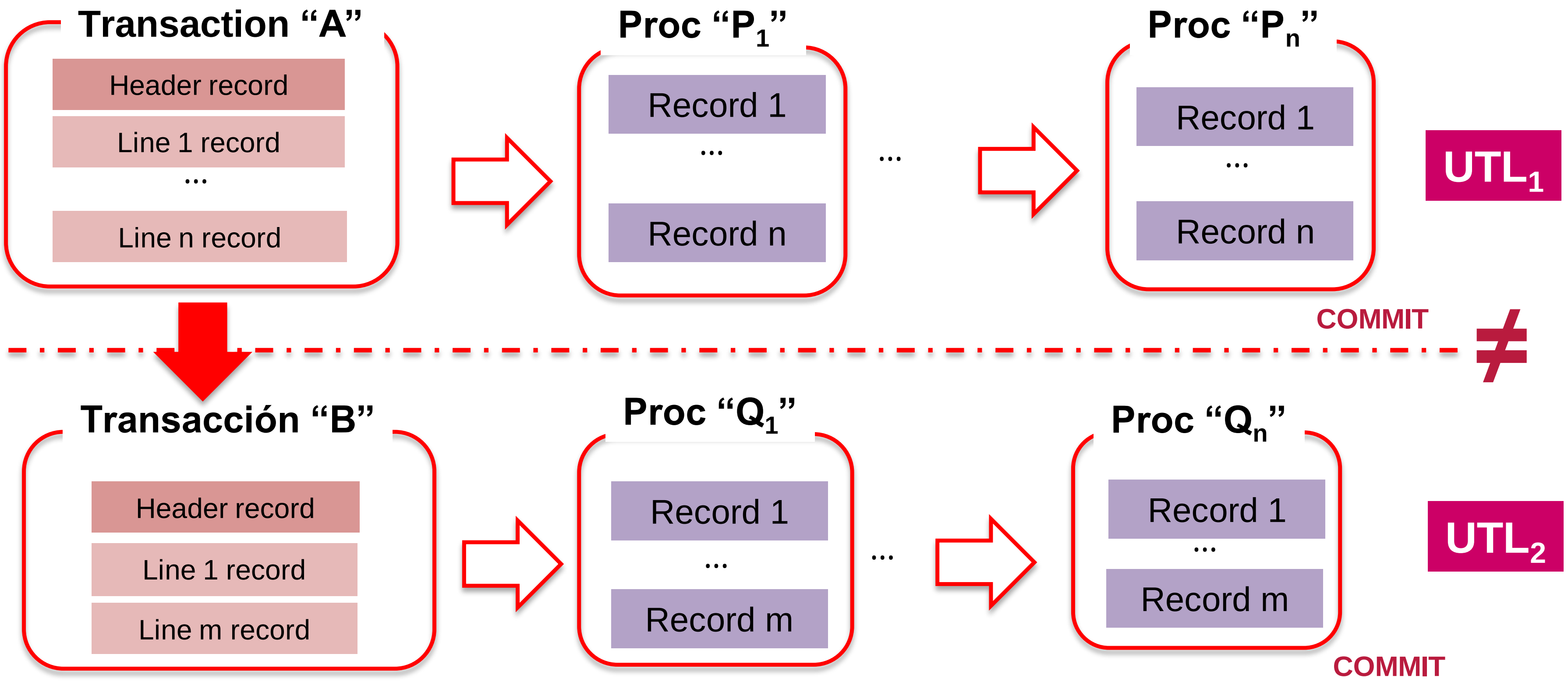

Transaction integrity

Commit on exit	Yes

B (parm$_1$, ... , parm$_n$) on AfterComplete;

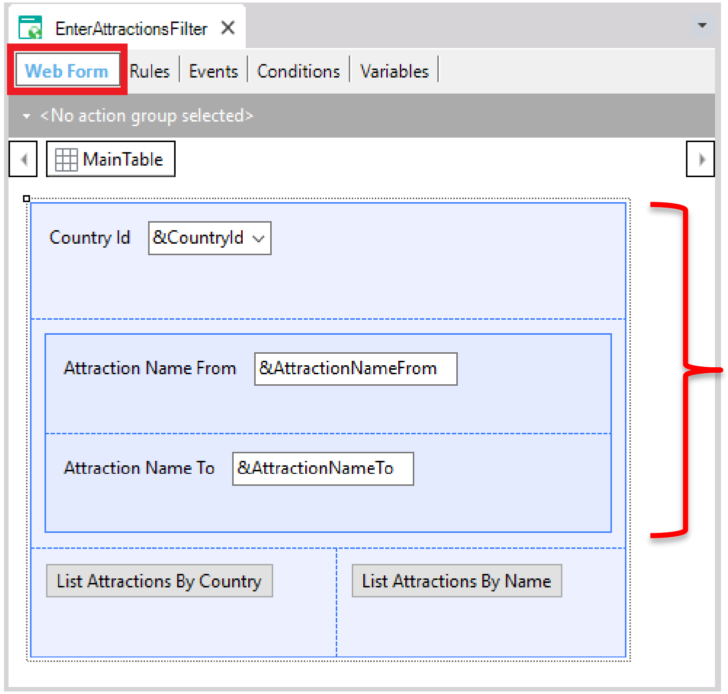
Customizing LUWs

Transaction may only commit its records and those of procedures in a chain of invocations: NOT the records of another transaction:



Web Panels

Web Panel without a grid, with variables in the form


EnterAttractionsFilter

Web Form | Rules | Events | Conditions | Variables

<No action group selected>

MainTable

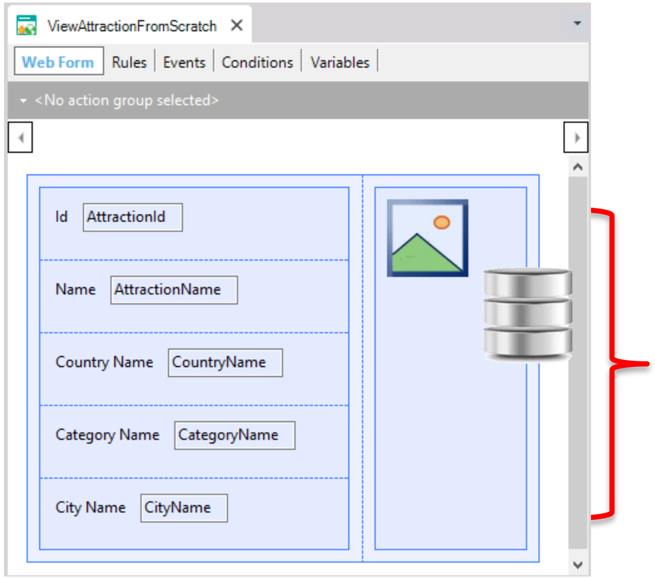
Country Id	&CountryId
Attraction Name From	&AttractionNameFrom
Attraction Name To	&AttractionNameTo

List Attractions By Country | List Attractions By Name

Variables: **input**
(not read-only)

Web Panel without a grid, with attributes in the form

Parm(in: **AttractionId**);

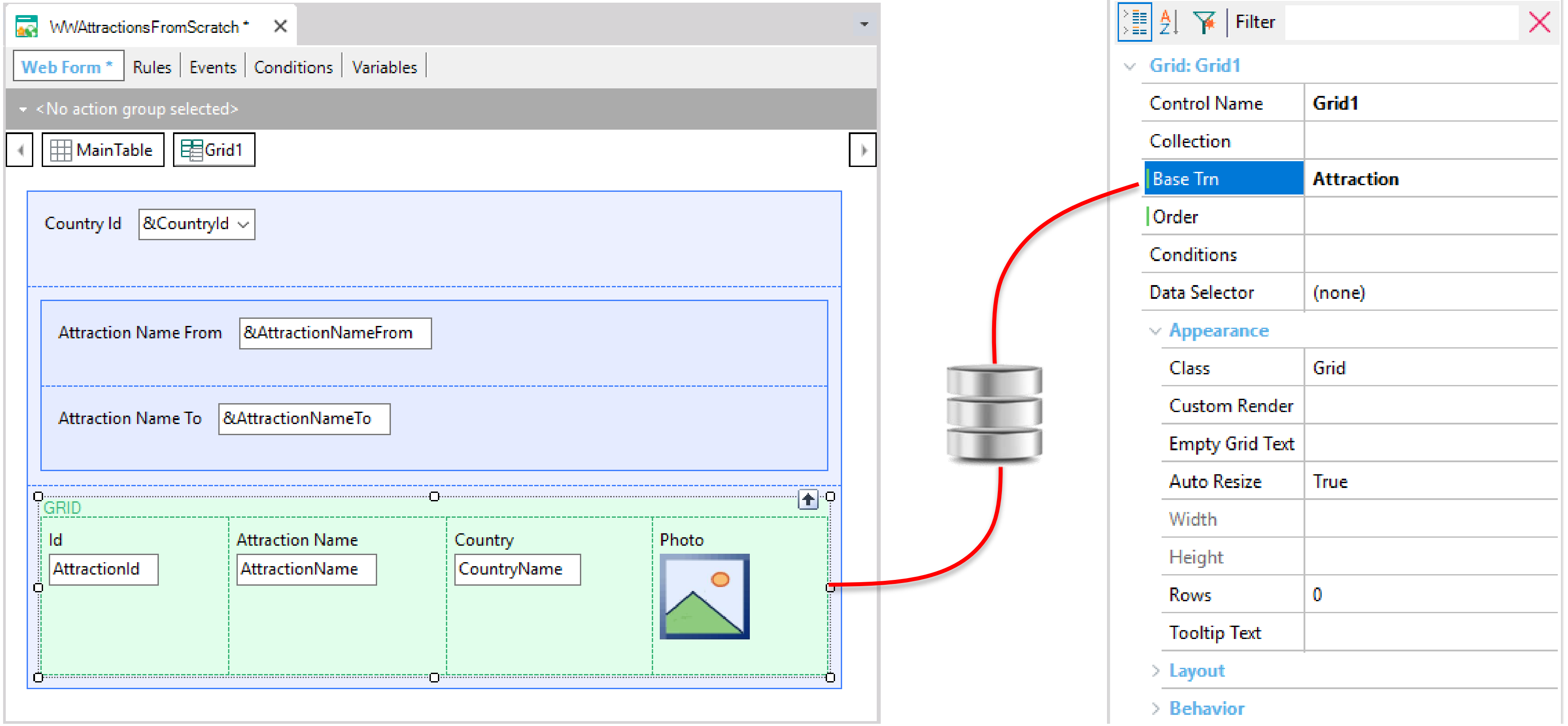


Id	AttractionId
Name	AttractionName
Country Name	CountryName
Category Name	CategoryName
City Name	CityName

Only **one** record is loaded

GeneXus™

Grid: WITH BASE TABLE



WWAttractionsFromScratch *

Web Form * | Rules | Events | Conditions | Variables

<No action group selected>

MainTable | Grid1

Country Id &CountryId

Attraction Name From &AttractionNameFrom

Attraction Name To &AttractionNameTo

GRID

Id	Attraction Name	Country	Photo
AttractionId	AttractionName	CountryName	

Properties

General | Class

Filter

Grid: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	
Conditions	
Data Selector	(none)

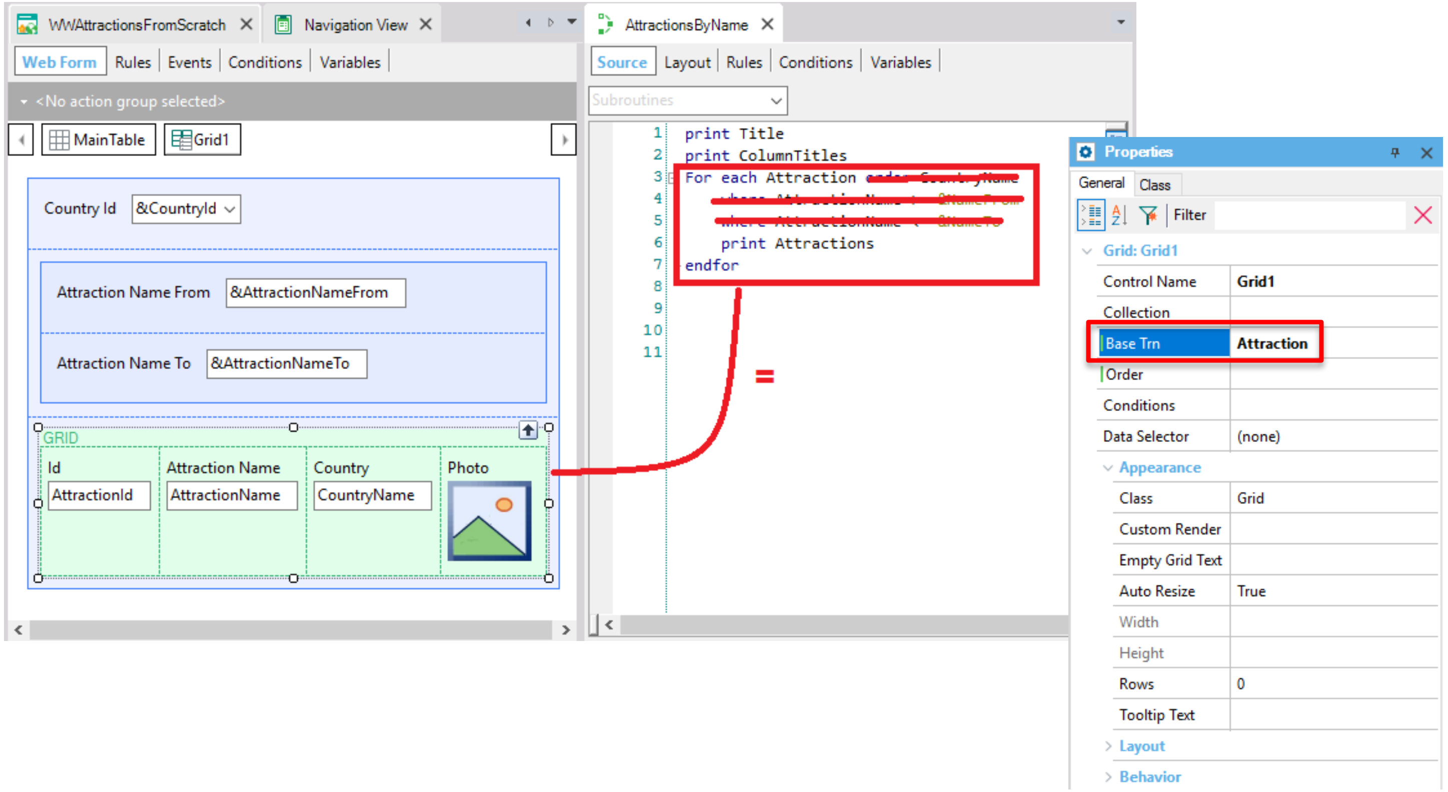
Appearance

Class	Grid
Custom Render	
Empty Grid Text	
Auto Resize	True
Width	
Height	
Rows	0
Tooltip Text	

Layout

Behavior

Base Transaction



WWAttractionsFromScratch | Navigation View | AttractionsByName

Web Form | Rules | Events | Conditions | Variables

<No action group selected>

MainTable | Grid1

Country Id: &CountryId

Attraction Name From: &AttractionNameFrom

Attraction Name To: &AttractionNameTo

GRID

Id	Attraction Name	Country	Photo
AttractionId	AttractionName	CountryName	

Source | Layout | Rules | Conditions | Variables

Subroutines

```
print Title
print ColumnTitles
For each Attraction ~~order CountryName~~
    ~~where AttractionName >= &NameFrom~~
    ~~where AttractionName <= &NameTo~~
    print Attractions
endfor
```

=

Properties

General | Class

Filter

Grid: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	
Conditions	
Data Selector	(none)

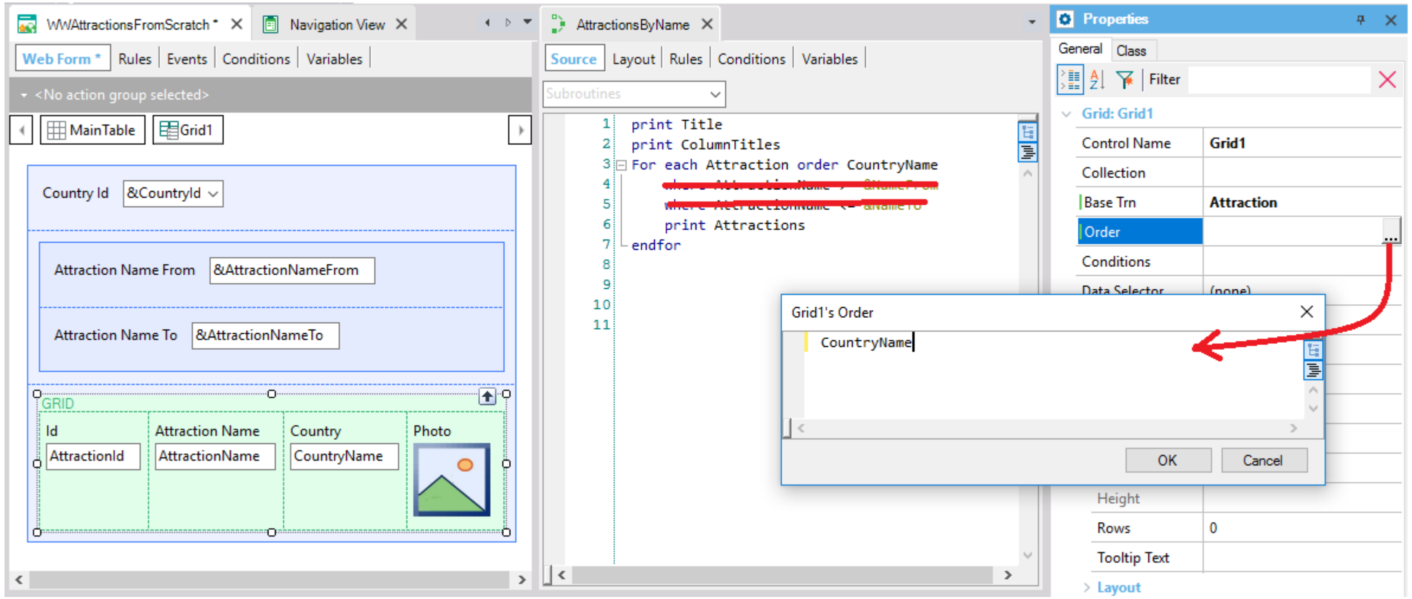
Appearance

Class	Grid
Custom Render	
Empty Grid Text	
Auto Resize	True
Width	
Height	
Rows	0
Tooltip Text	

Layout

Behavior

Order



```
print Title
print ColumnTitles
For each Attraction order CountryName
    ~~where AttractionName >= &NameFrom~~
    ~~where AttractionName <= &NameTo~~
    print Attractions
endfor
```

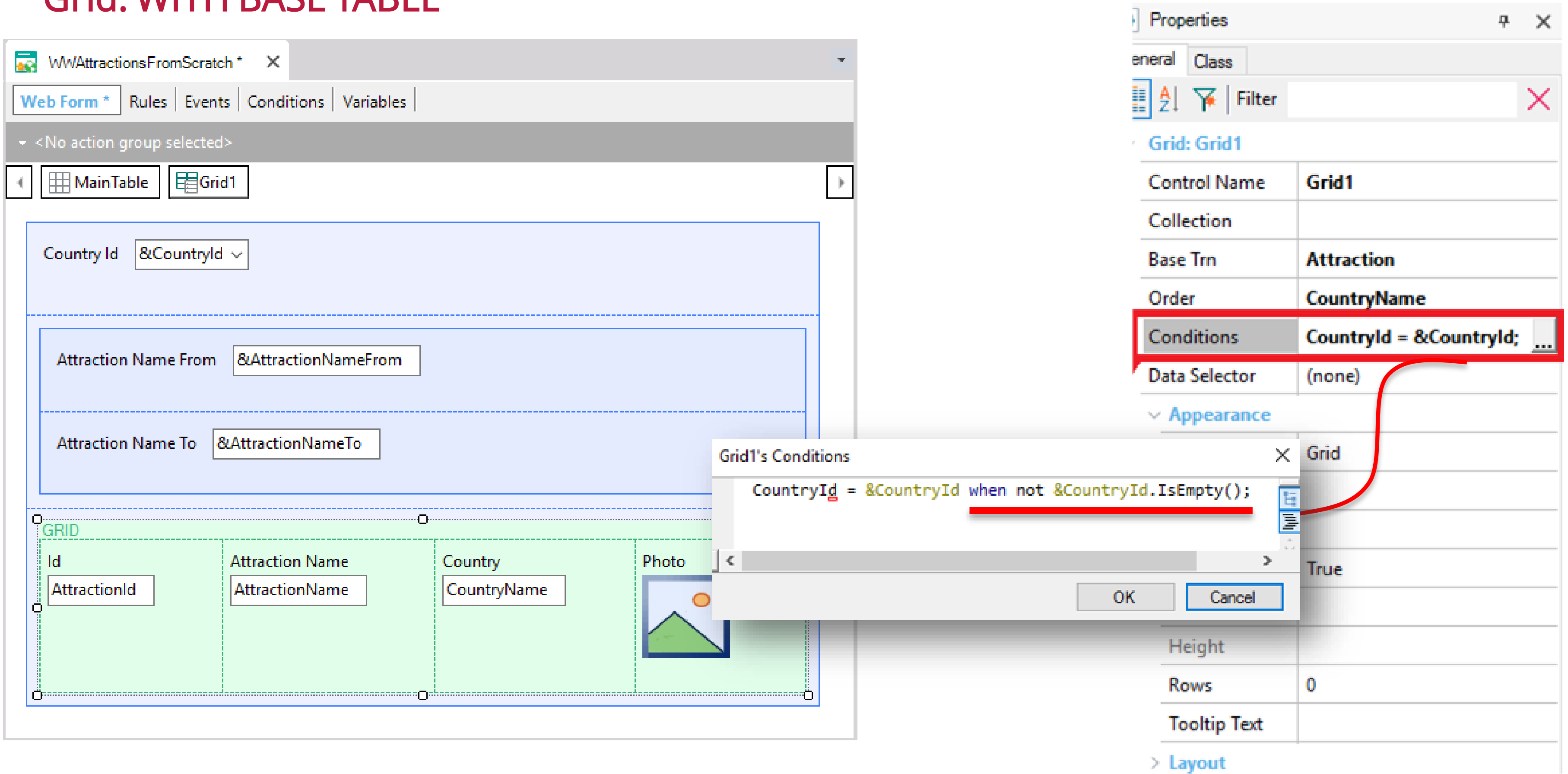
Control Name	Grid1
Collection	
Base Trn	Attraction
Order	
Conditions	
Data Selector	(none)
Height	
Rows	0
Tooltip Text	

Grid1's Order

```
CountryName
```

GeneXus™

Grid: WITH BASE TABLE



Country Id	&CountryId
Attraction Name From	&AttractionNameFrom
Attraction Name To	&AttractionNameTo

Id	Attraction Name	Country	Photo
AttractionId	AttractionName	CountryName	

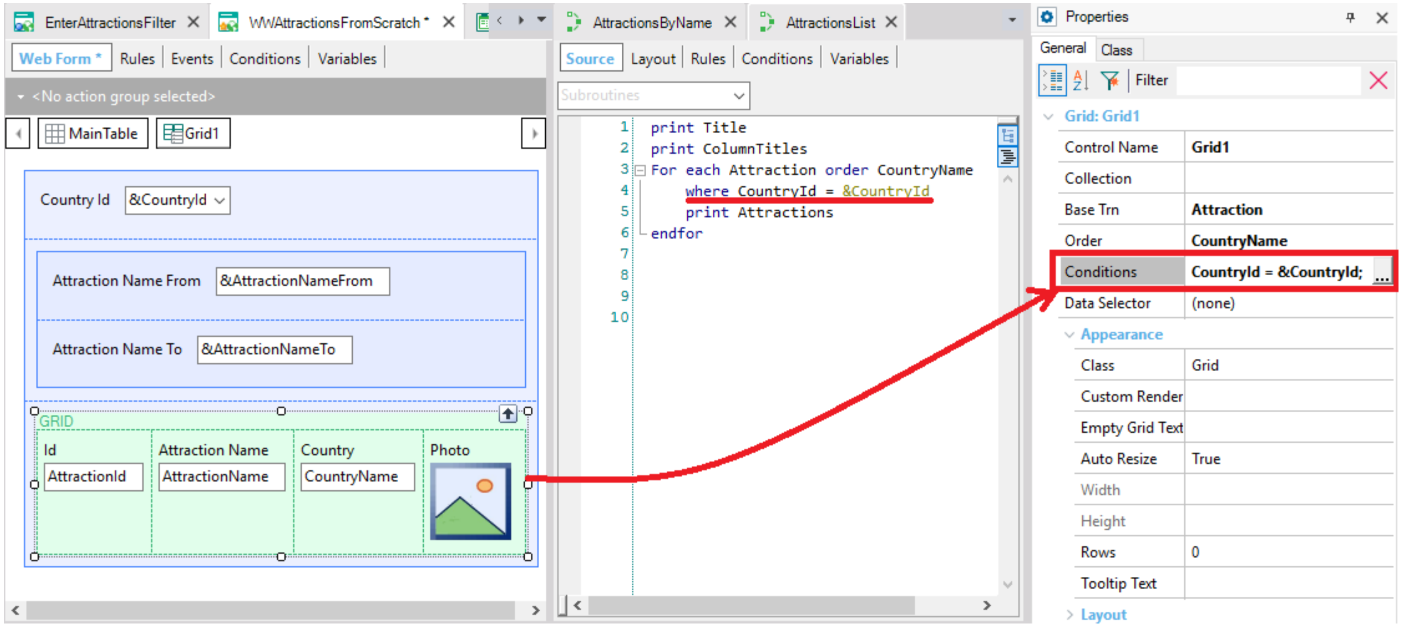
Grid: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	CountryName
Conditions	CountryId = &CountryId;
Data Selector	(none)
	Grid
	True
Height	
Rows	0
Tooltip Text	

Grid1's Conditions

```
CountryId = &CountryId when not &CountryId.IsEmpty();
```

Filter conditions



```
print Title
print ColumnTitles
For each Attraction order CountryName
    where CountryId = &CountryId
    print Attractions
endfor
```

Grid: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	CountryName
Conditions	CountryId = &CountryId;
Data Selector	(none)

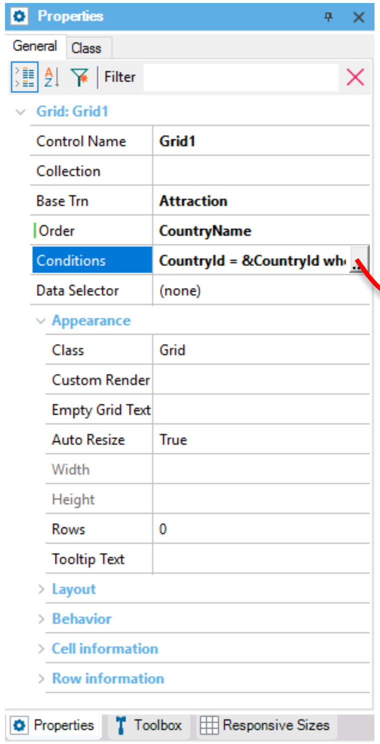
Appearance

Class	Grid
Custom Render	
Empty Grid Text	
Auto Resize	True
Width	
Height	
Rows	0
Tooltip Text	

Layout

GeneXus

Many conditions



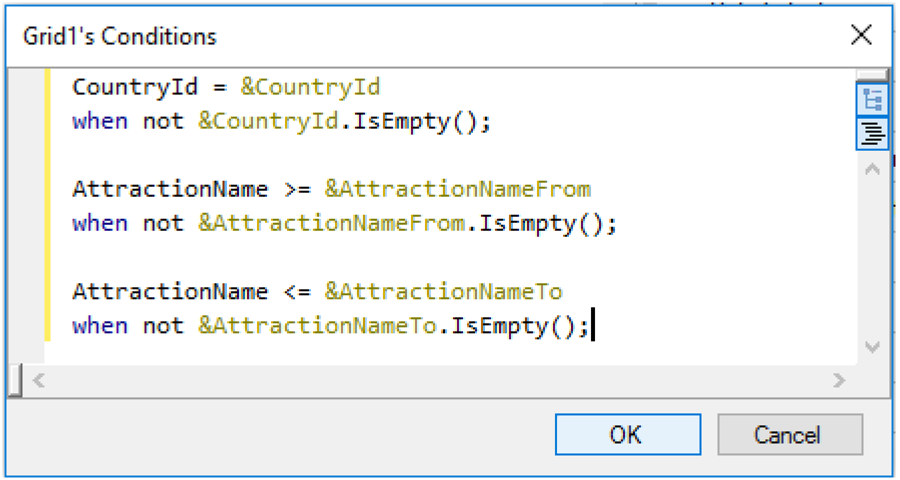
Grid: Grid1

Control Name	Grid1
Collection	
Base Trn	Attraction
Order	CountryName
Conditions	CountryId = &CountryId whe
Data Selector	(none)

Appearance

Class	Grid
Custom Render	
Empty Grid Text	
Auto Resize	True
Width	
Height	
Rows	0
Tooltip Text	

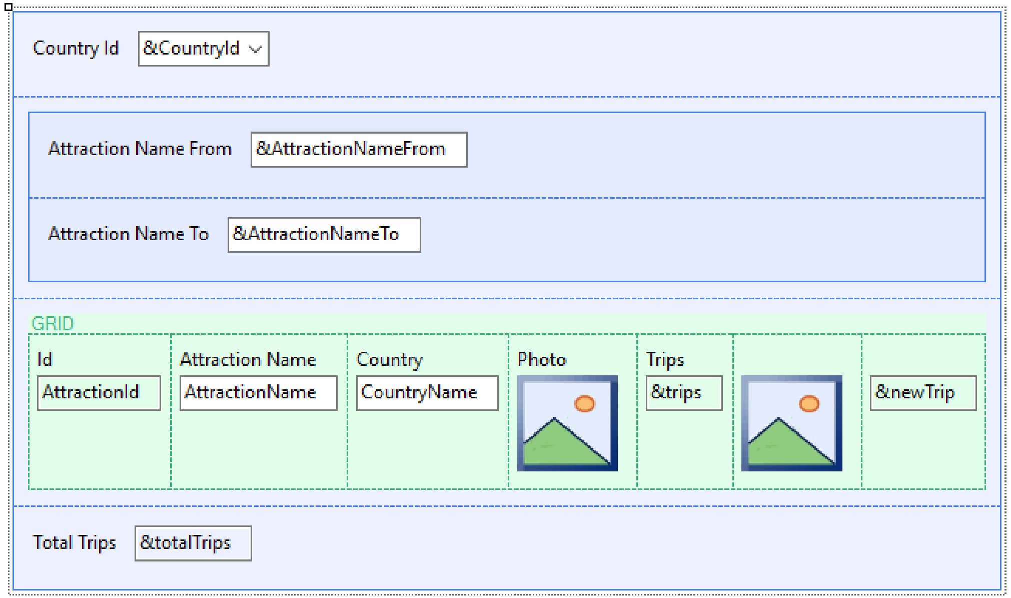
Layout
Behavior
Cell information
Row information



```
CountryId = &CountryId
when not &CountryId.IsEmpty();

AttractionName >= &AttractionNameFrom
when not &AttractionNameFrom.IsEmpty();

AttractionName <= &AttractionNameTo
when not &AttractionNameTo.IsEmpty();
```

GeneXus

Events


Country Id &CountryId

Attraction Name From &AttractionNameFrom

Attraction Name To &AttractionNameTo

GRID

Id	Attraction Name	Country	Photo	Trips		
AttractionId	AttractionName	CountryName		&trips		&newTrip

Total Trips &totalTrips

First time

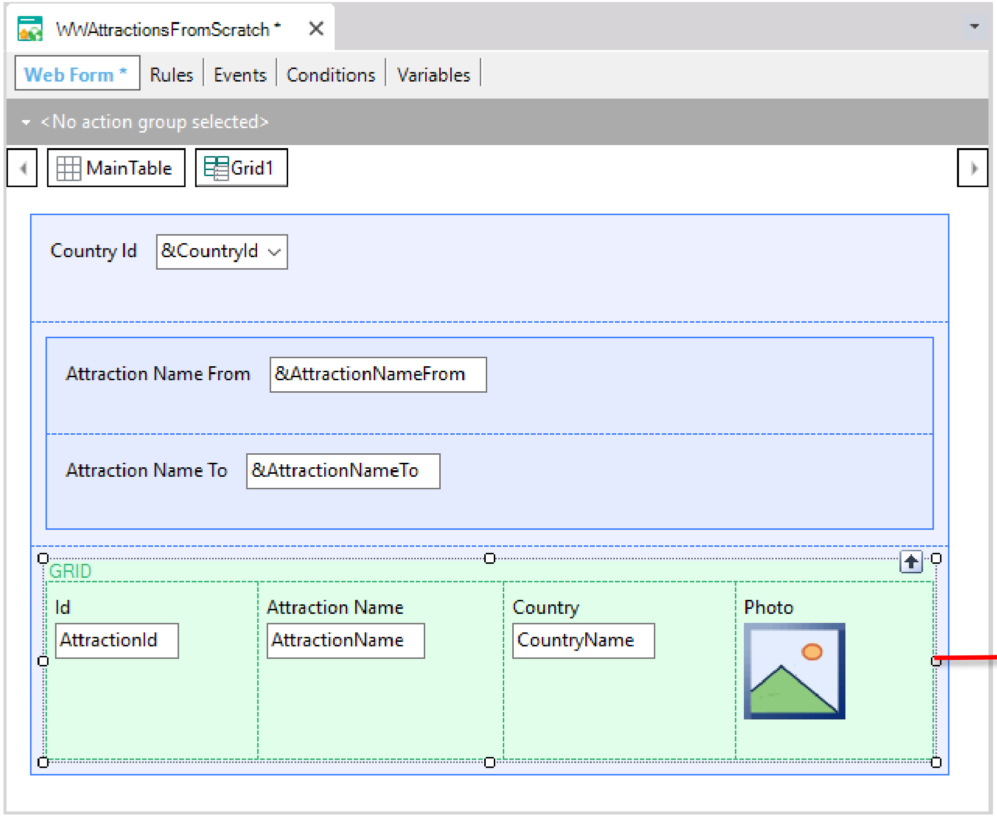
Start

Refresh

Load

User / Control Event

Load event in Web Panel WITH base table


WWAttractionsFromScratch *

Web Form * | Rules | Events | Conditions | Variables

<No action group selected>

MainTable | Grid1

Country Id: &CountryId

Attraction Name From: &AttractionNameFrom

Attraction Name To: &AttractionNameTo

GRID

Id	Attraction Name	Country	Photo
AttractionId	AttractionName	CountryName	

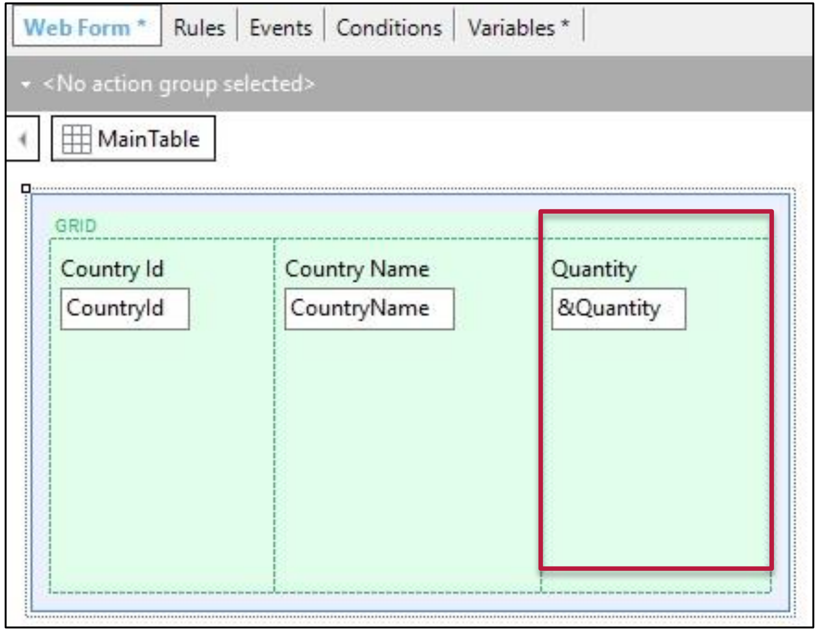
LOAD Event

“N times, as many as records existing in the table run through.”

Another example

```
Country
{
  CountryId*
  CountryName
}
```

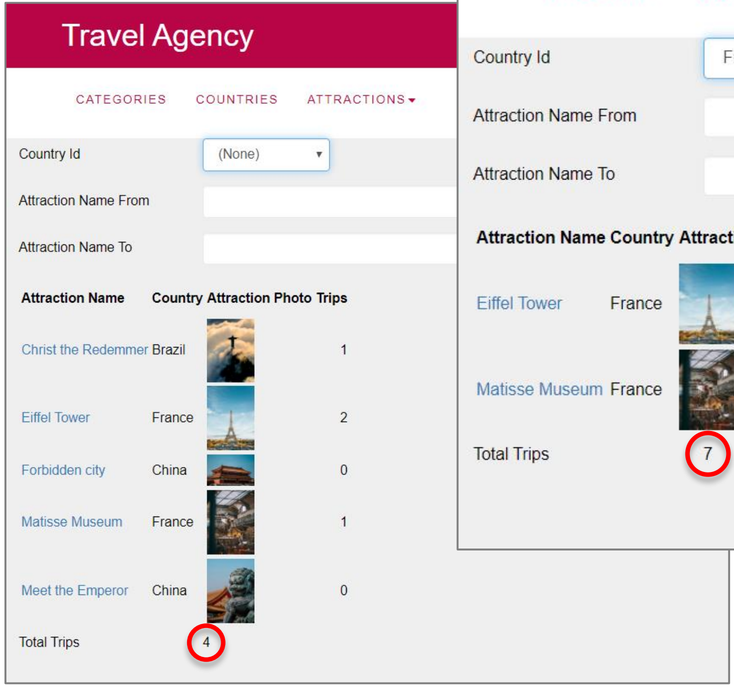
```
Attraction
{
  AttractionId*
  AttractionName
  CountryId
  CountryName
}
```



Country Id	Country Name	Quantity
CountryId	CountryName	&Quantity

```
Event Load
    &Quantity = Count(AttractionName)
endevent
```

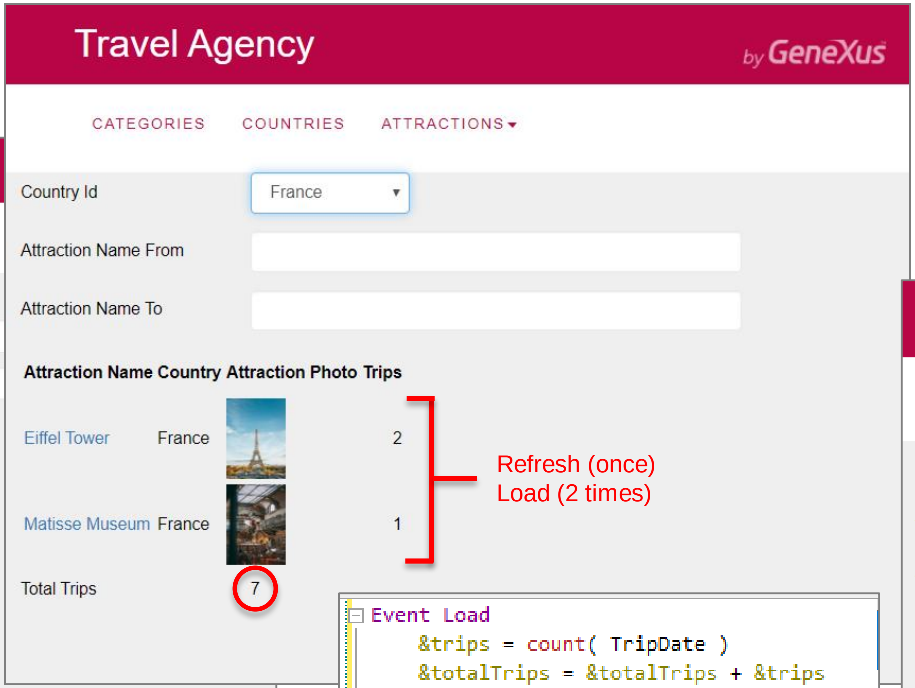
Refresh event


Travel Agency

Country Id	(None)
Attraction Name From	
Attraction Name To	

Attraction Name	Country	Attraction Photo	Trips
Christ the Redemmer	Brazil		1
Eiffel Tower	France		2
Forbidden city	China		0
Matisse Museum	France		1
Meet the Emperor	China		0

Total Trips 4

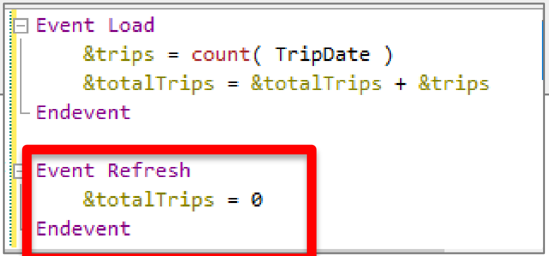

Travel Agency by GeneXus

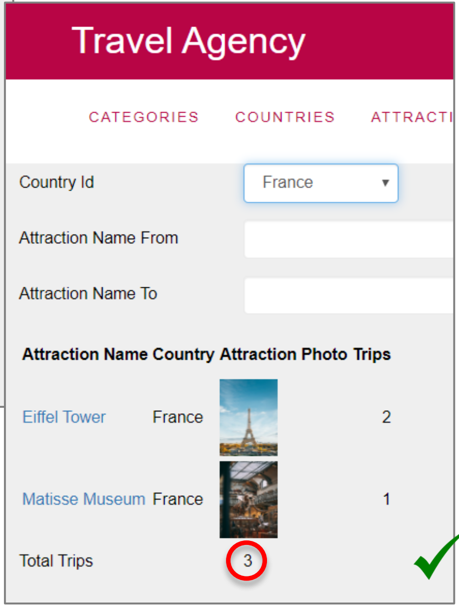
Country Id	France
Attraction Name From	
Attraction Name To	

Attraction Name	Country	Attraction Photo	Trips
Eiffel Tower	France		2
Matisse Museum	France		1

Total Trips 7

Refresh (once)
Load (2 times)


```
Event Load
    &trips = count( TripDate )
    &totalTrips = &totalTrips + &trips
Endevent

Event Refresh
    &totalTrips = 0
Endevent
```

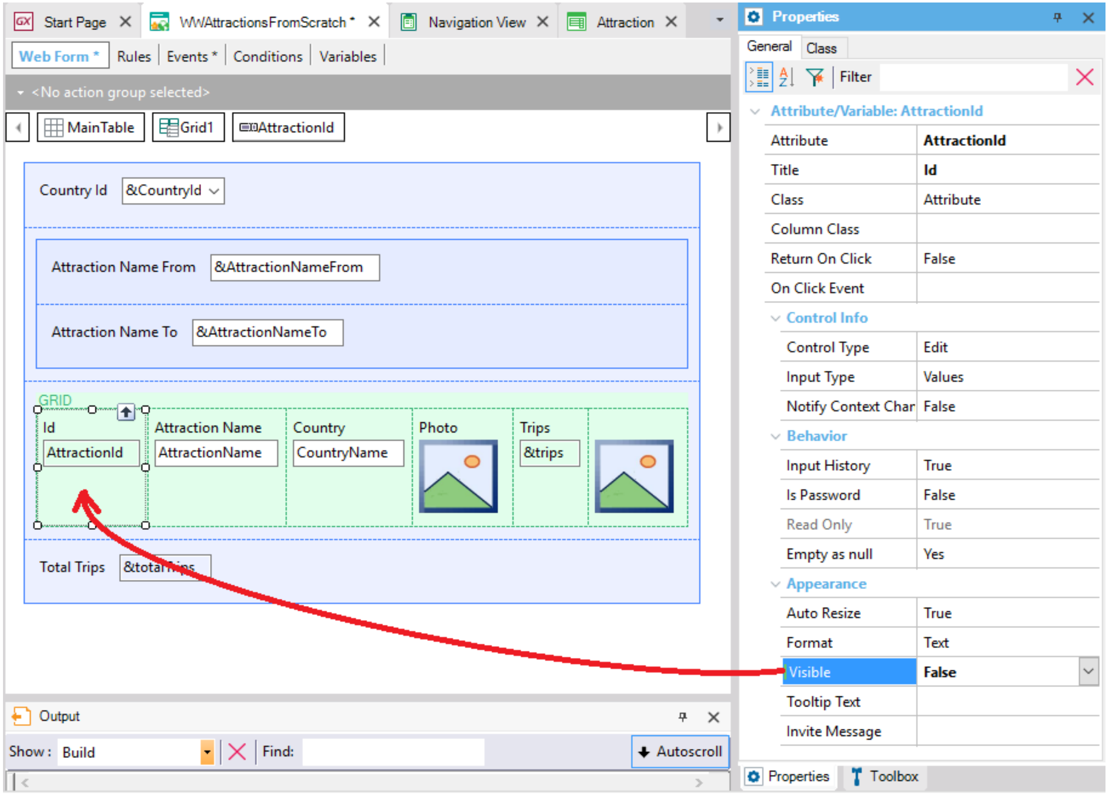

Travel Agency

Country Id	France
Attraction Name From	
Attraction Name To	

Attraction Name	Country	Attraction Photo	Trips
Eiffel Tower	France		2
Matisse Museum	France		1

Total Trips 3 ✓

Attributes in the Grid

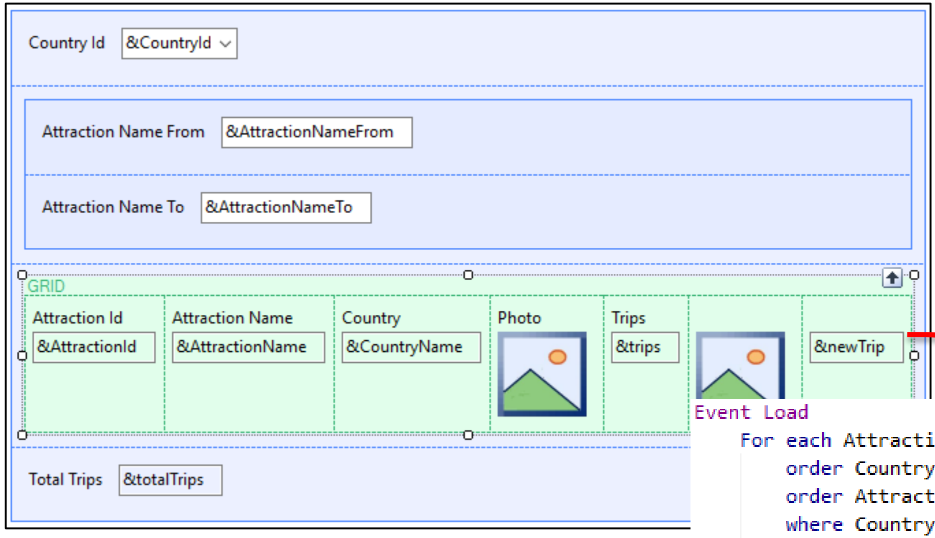


Attribute/Variable: AttractionId

Attribute	AttractionId
Title	Id
Class	Attribute
Column Class	
Return On Click	False
On Click Event	
Control Info	
Control Type	Edit
Input Type	Values
Notify Context Char	False
Behavior	
Input History	True
Is Password	False
Read Only	True
Empty as null	Yes
Appearance	
Auto Resize	True
Format	Text
Visible	False
Tooltip Text	
Invite Message	

Web Panels without Base Table

Web Panels WITHOUT BASE TABLE



Country Id &CountryId

Attraction Name From &AttractionNameFrom

Attraction Name To &AttractionNameTo

Attraction Id	Attraction Name	Country	Photo	Trips		
&AttractionId	&AttractionName	&CountryName		&trips		&newTrip

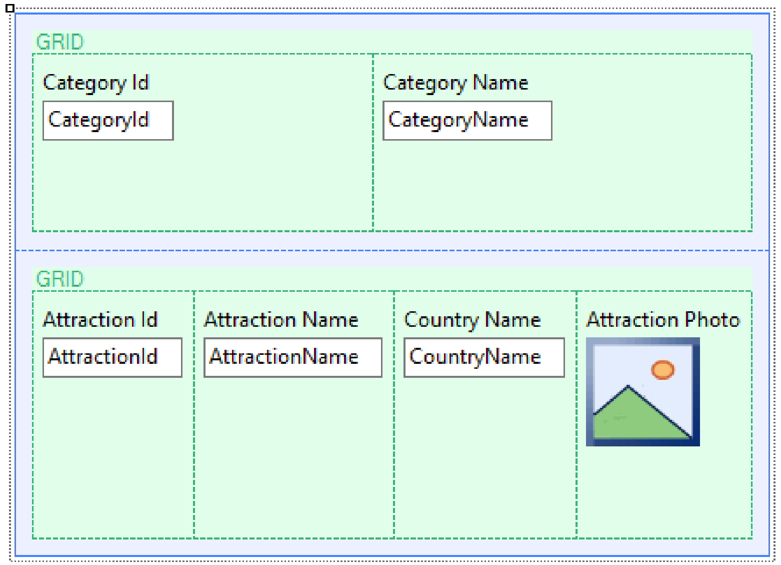
Total Trips &totalTrips

LOAD Event
"Once"

```
Event Load
    For each Attraction
        order CountryId, AttractionName when not &CountryId.IsEmpty()
        order AttractionName
        where CountryId = &CountryId when not &CountryId.IsEmpty()
        where AttractionName >= &AttractionNameFrom when not &AttractionNameFrom.IsEmpty()
        where AttractionName <= &AttractionNameTo when not &AttractionNameTo.IsEmpty()
        &AttractionId = AttractionId
        &AttractionName = AttractionName
        &CountryName = CountryName
        &AttractionPhoto = AttractionPhoto
        &trips = count( TripDate )
        Load
        &totalTrips = &totalTrips + &trips
    endfor
Endevent
```

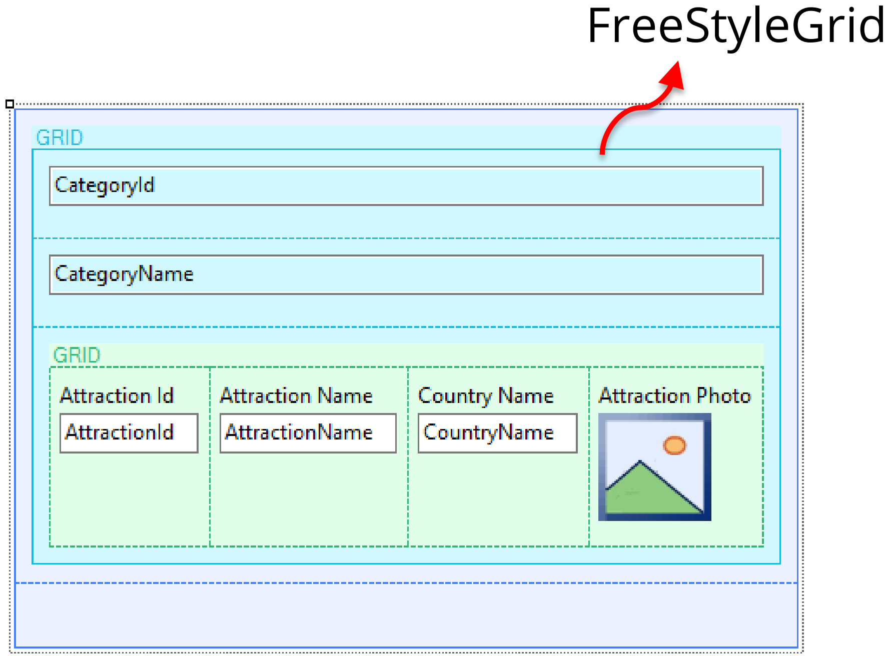
Web Panels
Multiple Grids

Multiple grids



PARALLEL

Independent navigation



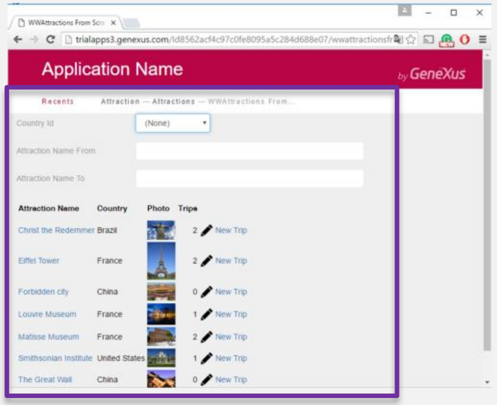
NESTED

Navigation of related tables

Types of Web Panels

Types of Web Panels

Web page (default)



WWAttractions From Scr...

trialapps3.genexus.com/Id8562acf4c97c0fe8095a5c284d688e07/wwattractionsfr...

Application Name *by GeneXus*

Recents — Attraction — Attractions — WWAttractions From...

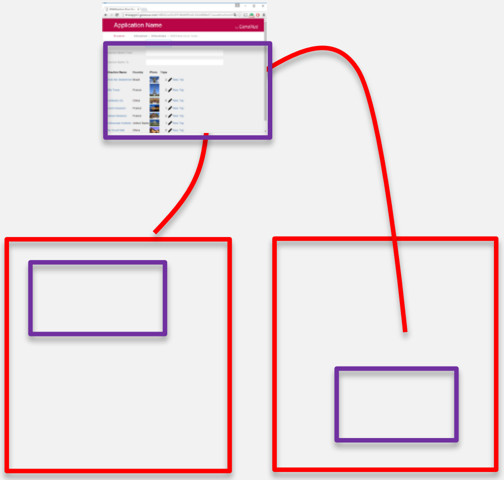
Country Id: (None)

Attraction Name From:

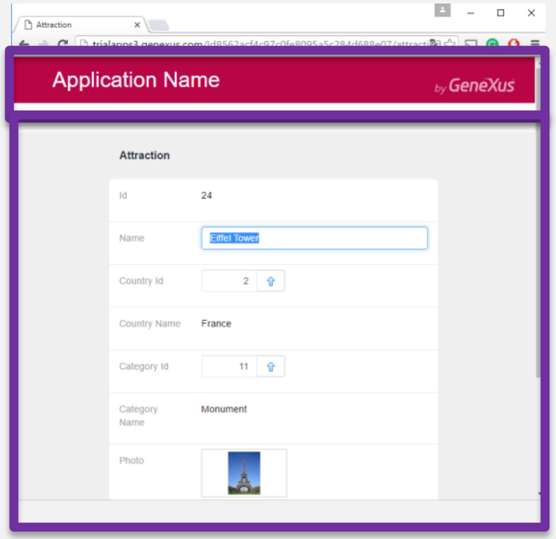
Attraction Name To:

Attraction Name	Country	Photo	Trips	
Christ the Redemmer	Brazil		2	New Trip
Eiffel Tower	France		2	New Trip
Forbidden city	China		0	New Trip
Louvre Museum	France		1	New Trip
Matisse Museum	France		2	New Trip
Smithsonian Institute	United States		1	New Trip
The Great Wall	China		0	New Trip

Component



Master page



Attraction

trialapps3.genexus.com/Id8562acf4c97c0fe8095a5c284d688e07/attracti...

Application Name *by GeneXus*

Attraction

Id	24
Name	Eiffel Tower
Country Id	2
Country Name	France
Category Id	11
Category Name	Monument
Photo	

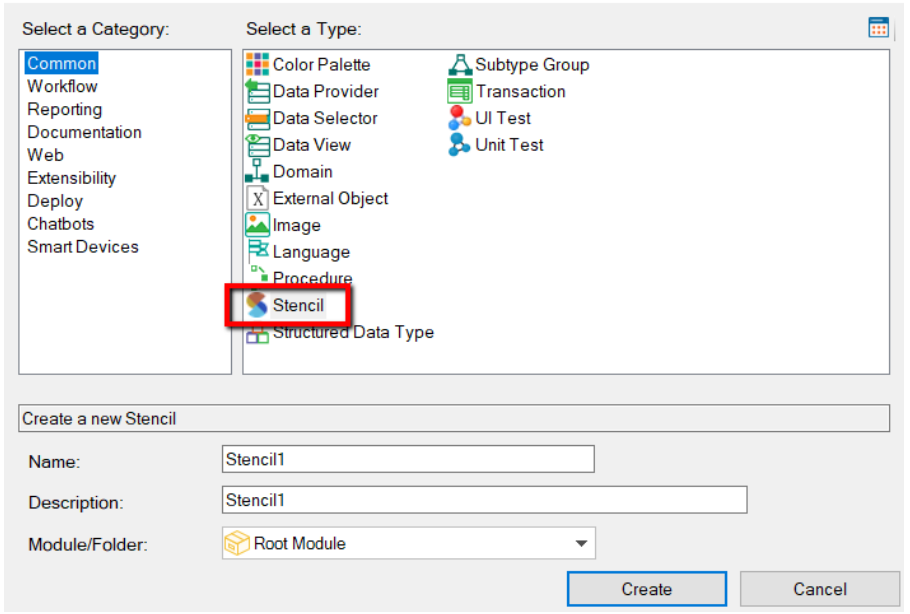
Design Systems

Design Systems

Elements that give consistency and coherence to the UX

Master Page	e.g. Header / footer	
Theme	Classes	**RWD**
Responsive sizes	Responsive tables	**RWD**
Controls	User control objects	
Base styles	CSS libraries	
Stencils	Design components	
Patterns	Design / behaviour	

Stencils


Select a Category:

- Common
- Workflow
- Reporting
- Documentation
- Web
- Extensibility
- Deploy
- Chatbots
- Smart Devices

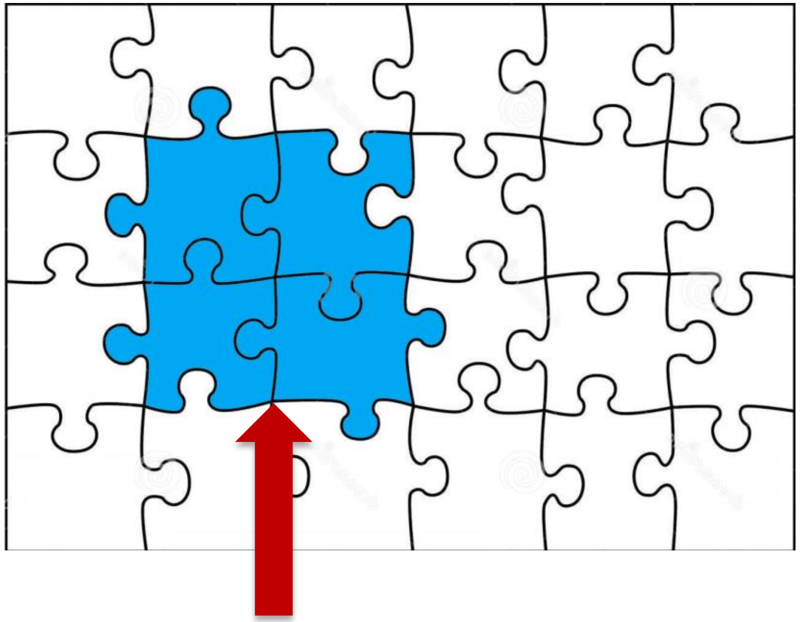
Select a Type:

- Color Palette
- Data Provider
- Data Selector
- Data View
- Domain
- External Object
- Image
- Language
- Procedure
- Stencil
- Structured Data Type
- Subtype Group
- Transaction
- UI Test
- Unit Test

Create a new Stencil

Name:	Stencil1
Description:	Stencil1
Module/Folder:	Root Module

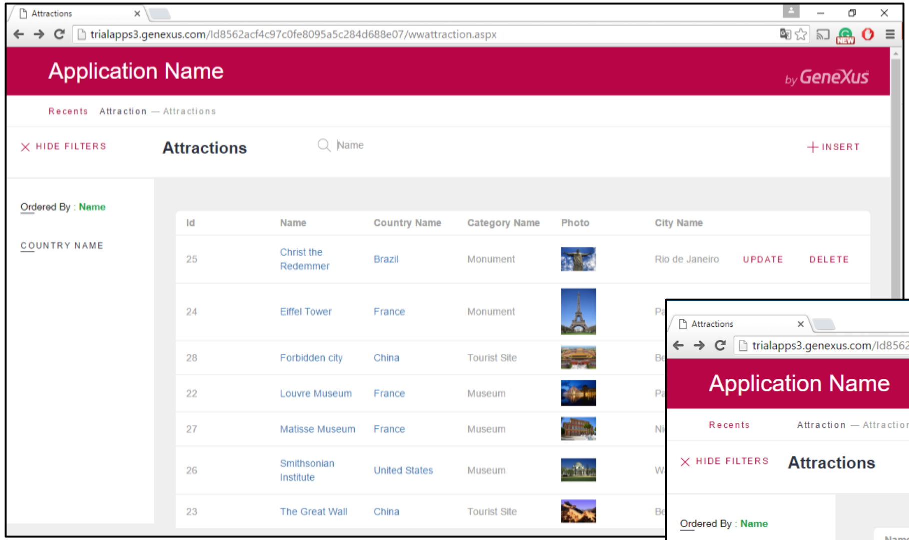
Create Cancel


Design component

Object that allows repeating the design of the same portion of the screen (a set of controls), in many screens

Responsive Design

Responsive Web design



Attractions

trialapps3.genexus.com/Id8562acf4c97c0fe8095a5c284d688e07/wwattraction.aspx

Application Name by GeneXus

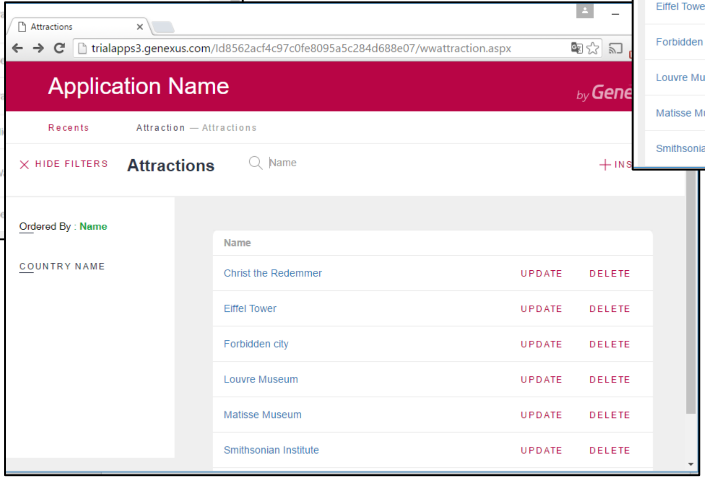
Recents Attraction — Attractions

HIDE FILTERS Attractions Name INSERT

Ordered By : Name

COUNTRY NAME

Id	Name	Country Name	Category Name	Photo	City Name		
25	Christ the Redemmer	Brazil	Monument		Rio de Janeiro	UPDATE	DELETE
24	Eiffel Tower	France	Monument		Pa		
28	Forbidden city	China	Tourist Site		Be		
22	Louvre Museum	France	Museum		Pa		
27	Matisse Museum	France	Museum		Ni		
26	Smithsonian Institute	United States	Museum		W		
23	The Great Wall	China	Tourist Site		Be		



Attractions

trialapps3.genexus.com/Id8562acf4c97c0fe8095a5c284d688e07/wwattraction.aspx

Application Name by Gene

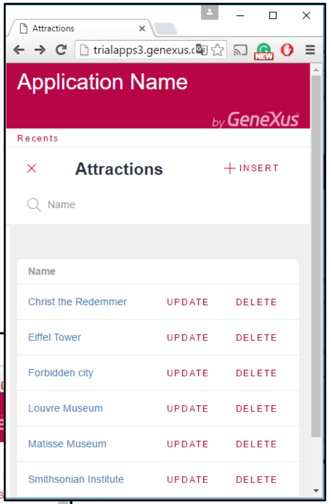
Recents Attraction — Attractions

HIDE FILTERS Attractions Name INS

Ordered By : Name

COUNTRY NAME

Name		
Christ the Redemmer	UPDATE	DELETE
Eiffel Tower	UPDATE	DELETE
Forbidden city	UPDATE	DELETE
Louvre Museum	UPDATE	DELETE
Matisse Museum	UPDATE	DELETE
Smithsonian Institute	UPDATE	DELETE



Attractions

trialapps3.genexus.c

Application Name by GeneXus

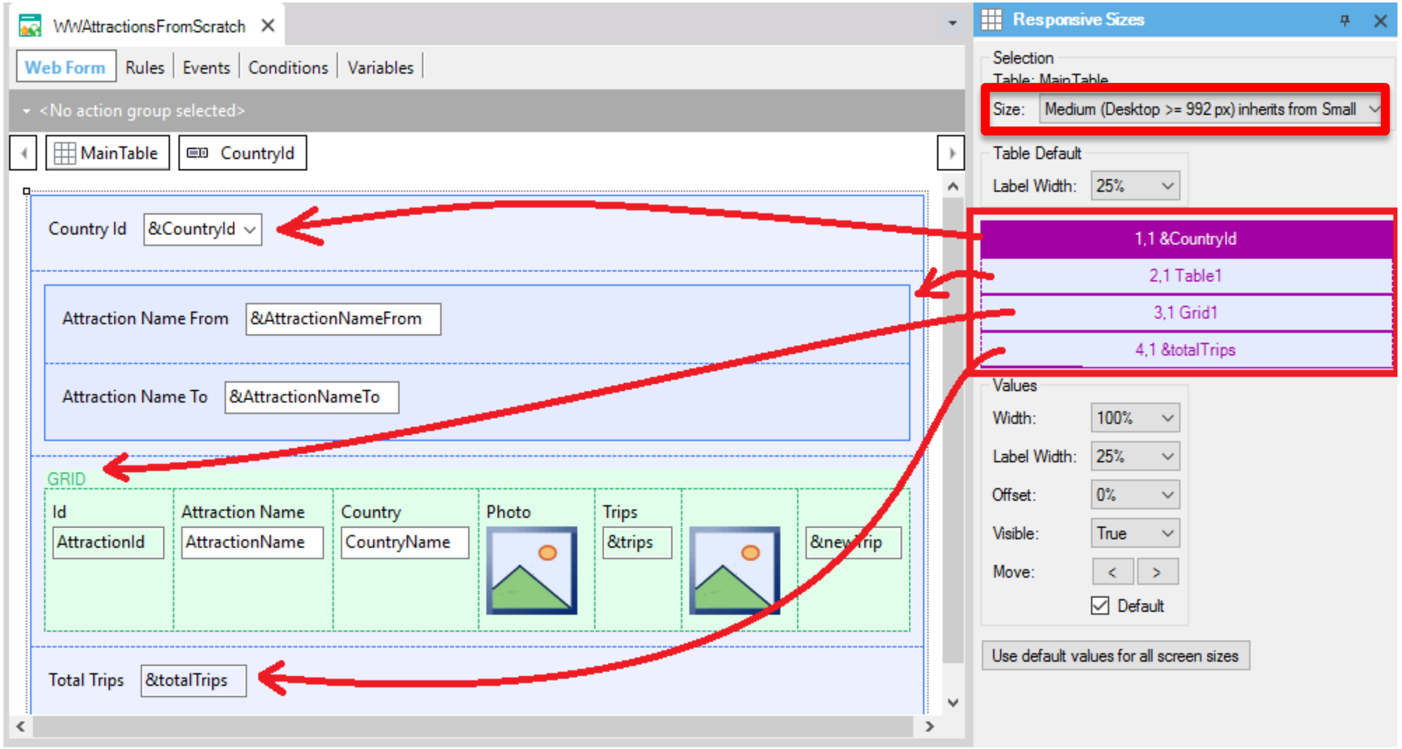
Recents

Attractions INSERT

Name

Name		
Christ the Redemmer	UPDATE	DELETE
Eiffel Tower	UPDATE	DELETE
Forbidden city	UPDATE	DELETE
Louvre Museum	UPDATE	DELETE
Matisse Museum	UPDATE	DELETE
Smithsonian Institute	UPDATE	DELETE

Responsive Web design



WWAttractionsFromScratch

Web Form | Rules | Events | Conditions | Variables

<No action group selected>

MainTable | CountryId

Country Id &CountryId

Attraction Name From &AttractionNameFrom

Attraction Name To &AttractionNameTo

GRID

Id	Attraction Name	Country	Photo	Trips		
AttractionId	AttractionName	CountryName		&trips		&newTrip

Total Trips &totalTrips

Responsive Sizes

Selection

Table: MainTable

Size: Medium (Desktop >= 992 px) inherits from Small

Table Default

Label Width: 25%

1,1 &CountryId

2,1 Table1

3,1 Grid1

4,1 &totalTrips

Values

Width: 100%

Label Width: 25%

Offset: 0%

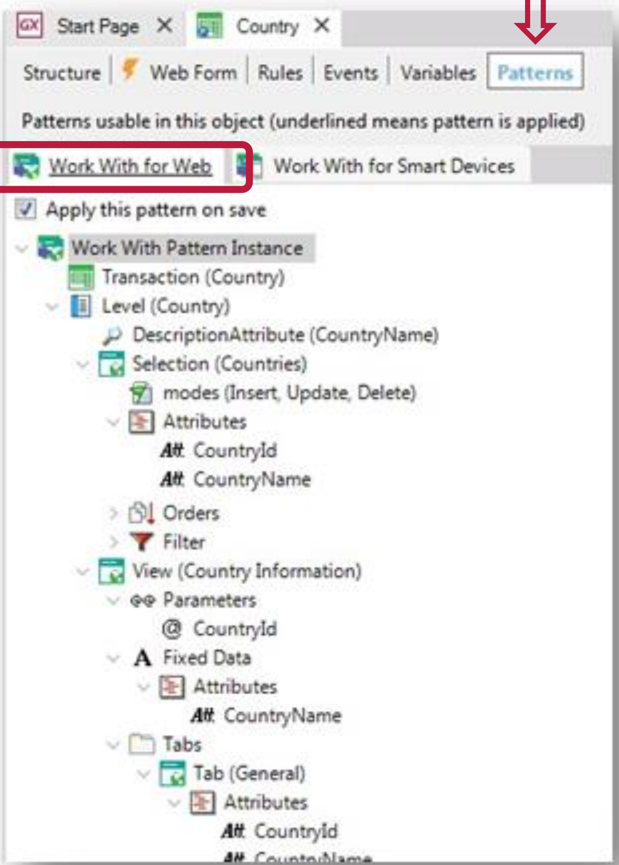
Visible: True

Move: < >

☑ Default

Use default values for all screen sizes

Patterns



Start Page | Country

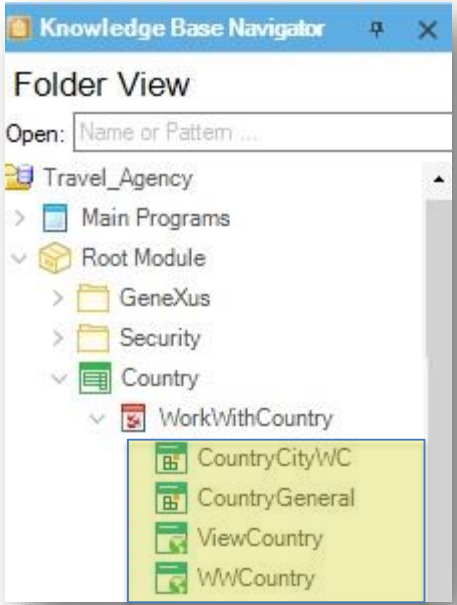
Structure | Web Form | Rules | Events | Variables | Patterns

Patterns usable in this object (underlined means pattern is applied)

Work With for Web | Work With for Smart Devices

☑ Apply this pattern on save

```
Work With Pattern Instance
  Transaction (Country)
  Level (Country)
    DescriptionAttribute (CountryName)
    Selection (Countries)
      modes (Insert, Update, Delete)
      Attributes
        CountryId
        CountryName
      Orders
      Filter
    View (Country Information)
      Parameters
        CountryId
      Fixed Data
        Attributes
          CountryName
      Tabs
        Tab (General)
          Attributes
            CountryId
```



Knowledge Base Navigator

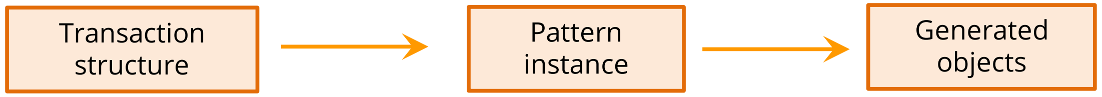
Folder View

Open: Name or Pattern ...

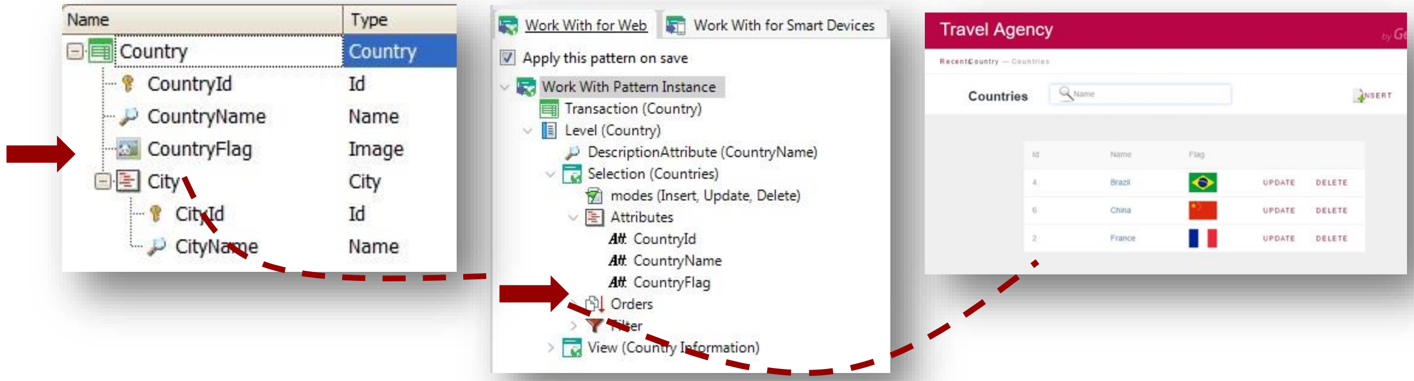
```
Travel_Agency
  Main Programs
  Root Module
    GeneXus
    Security
    Country
      WorkWithCountry
        CountryCityWC
        CountryGeneral
        ViewCountry
        WWCountry
```

Automatically generated by GeneXus

Dynamics



Example: New attribute CountryFlag



Name	Type
Country	Country
CountryId	Id
CountryName	Name
CountryFlag	Image
City	City
CityId	Id
CityName	Name

Work With for Web | Work With for Smart Devices

☑ Apply this pattern on save

```
Work With Pattern Instance
  Transaction (Country)
  Level (Country)
    DescriptionAttribute (CountryName)
    Selection (Countries)
      modes (Insert, Update, Delete)
      Attributes
        Att CountryId
        Att CountryName
        Att CountryFlag
      Orders
      Filter
    View (Country Information)
```

Travel Agency

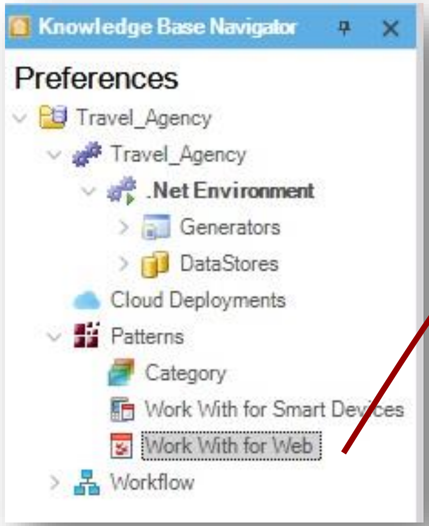
RecentCountry — Countries

Countries | Name | INSERT

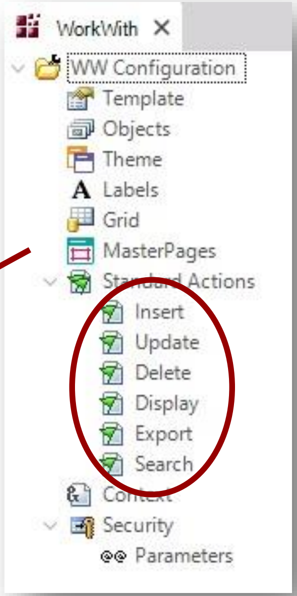
Id	Name	Flag		
4	Brazil		UPDATE	DELETE
6	China		UPDATE	DELETE
2	France		UPDATE	DELETE

General Settings

How are all instances initialized?

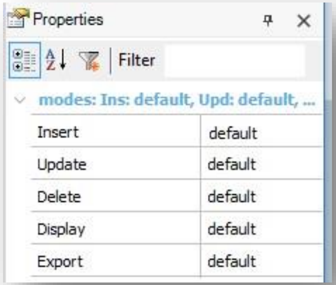

Knowledge Base Navigator

Preferences
- Travel_Agency
  - Travel_Agency
    - .Net Environment
      - Generators
      - DataStores
    - Cloud Deployments
  - Patterns
    - Category
    - Work With for Smart Devices
    - Work With for Web
  - Workflow


WorkWith

- WW Configuration
  - Template
  - Objects
  - Theme
  - Labels
  - Grid
  - MasterPages
  - Standard Actions
    - Insert
    - Update
    - Delete
    - Display
    - Export
    - Search
  - Context
  - Security
    - Parameters

In Country...

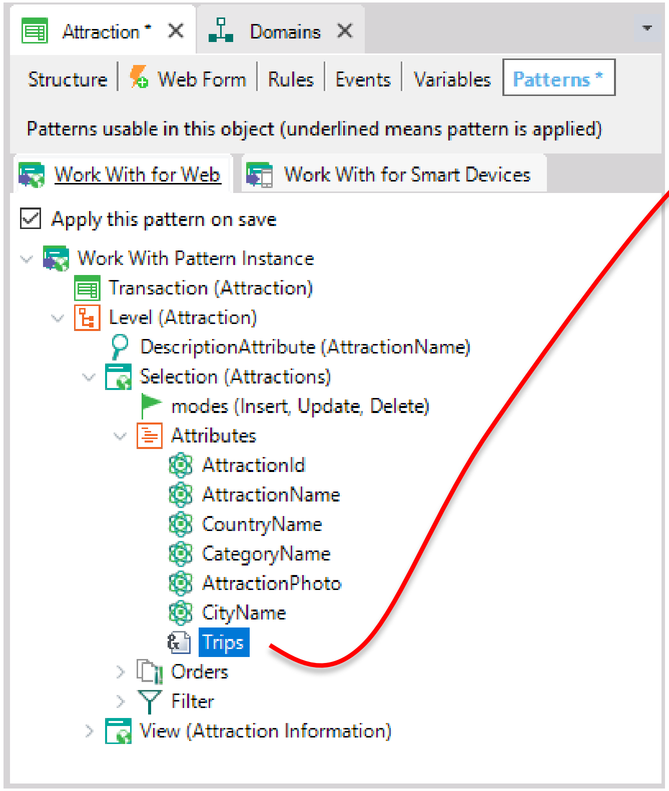

Properties

Filter

modes: Ins: default, Upd: default, ...

Insert	default
Update	default
Delete	default
Display	default
Export	default

Adding variables or attributes



Attraction * | Domains

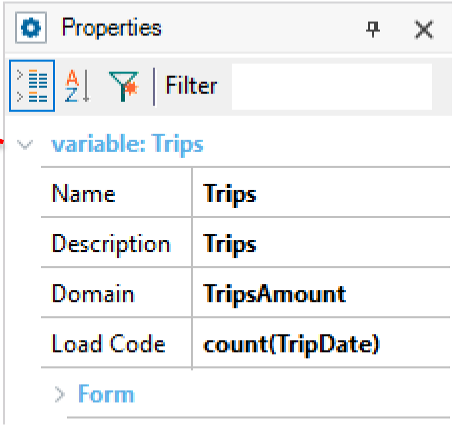
Structure | Web Form | Rules | Events | Variables | Patterns *

Patterns usable in this object (underlined means pattern is applied)

Work With for Web | Work With for Smart Devices

☑ Apply this pattern on save

```
Work With Pattern Instance
    Transaction (Attraction)
    Level (Attraction)
        DescriptionAttribute (AttractionName)
        Selection (Attractions)
            modes (Insert, Update, Delete)
            Attributes
                AttractionId
                AttractionName
                CountryName
                CategoryName
                AttractionPhoto
                CityName
                Trips
            Orders
            Filter
        View (Attraction Information)
```



Properties

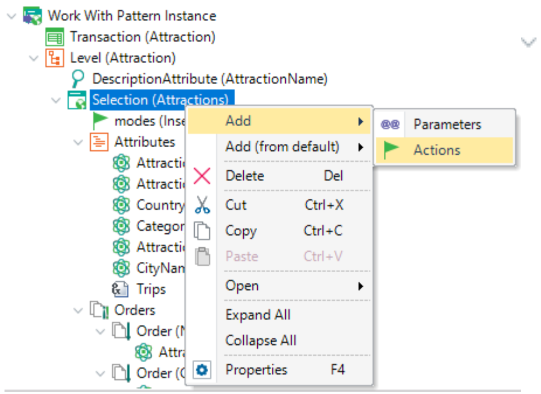
Filter

variable: Trips

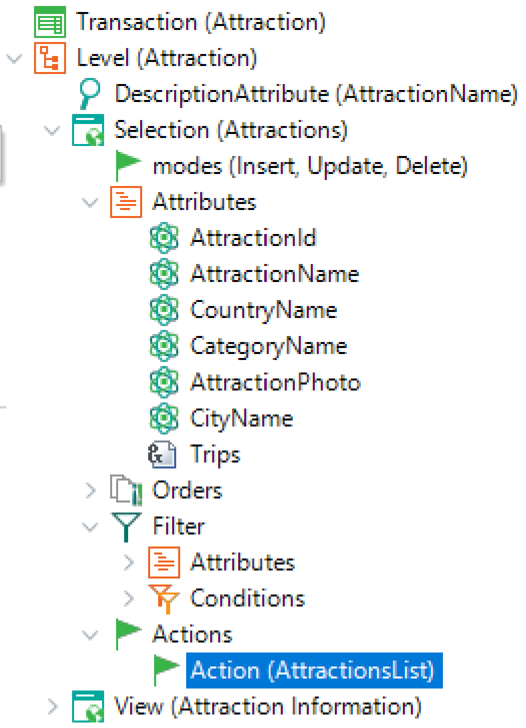
Name	Trips
Description	Trips
Domain	TripsAmount
Load Code	count(TripDate)

Form

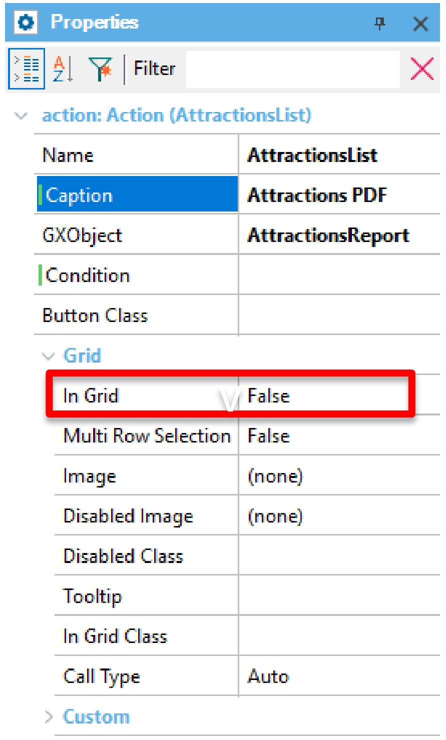
Adding actions



Work With Pattern Instance
Transaction (Attraction)
Level (Attraction)
DescriptionAttribute (AttractionName)
Selection (Attractions)
modes (Inse
Attributes
Attracti
Attracti
Country
Categor
Attracti
CityNan
Trips
Orders
Order (I
Attr
Order (



Transaction (Attraction)
Level (Attraction)
DescriptionAttribute (AttractionName)
Selection (Attractions)
modes (Insert, Update, Delete)
Attributes
AttractionId
AttractionName
CountryName
CategoryName
AttractionPhoto
CityName
Trips
Orders
Filter
Attributes
Conditions
Actions
Action (AttractionsList)
View (Attraction Information)



action: Action (AttractionsList)

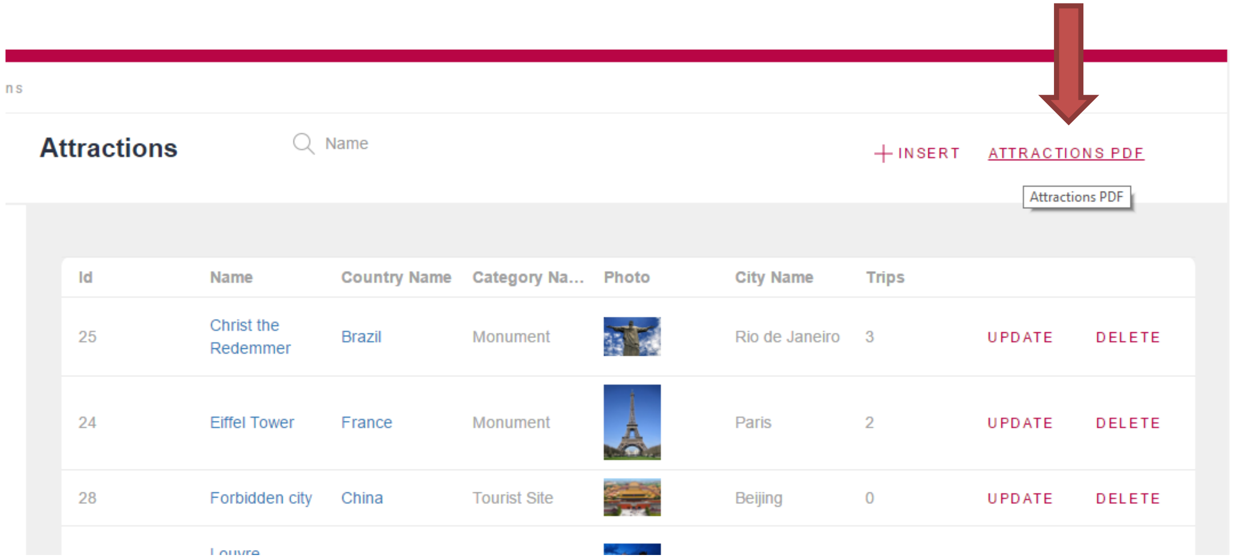
Name	AttractionsList
Caption	Attractions PDF
GXObject	AttractionsReport
Condition	
Button Class	

Grid

In Grid	False
Multi Row Selection	False
Image	(none)
Disabled Image	(none)
Disabled Class	
Tooltip	
In Grid Class	
Call Type	Auto

Custom

Adding actions



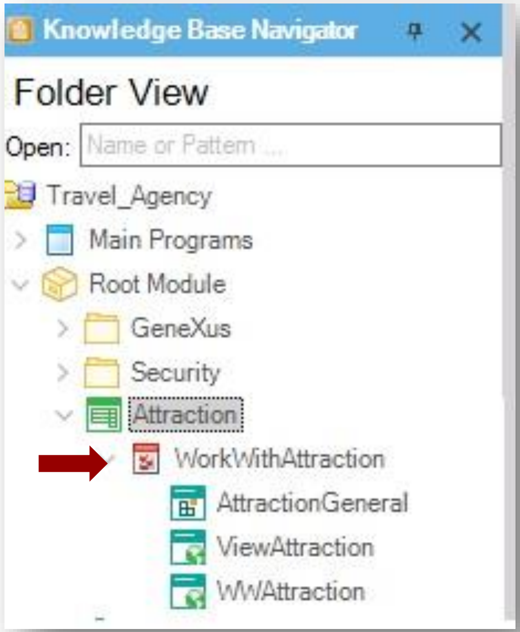
ns

Attractions

Id	Name	Country Name	Category Na...	Photo	City Name	Trips		
25	Christ the Redemmer	Brazil	Monument		Rio de Janeiro	3	UPDATE	DELETE
24	Eiffel Tower	France	Monument		Paris	2	UPDATE	DELETE
28	Forbidden city	China	Tourist Site		Beijing	0	UPDATE	DELETE

Deleting the pattern

Delete


Knowledge Base Navigator

Folder View

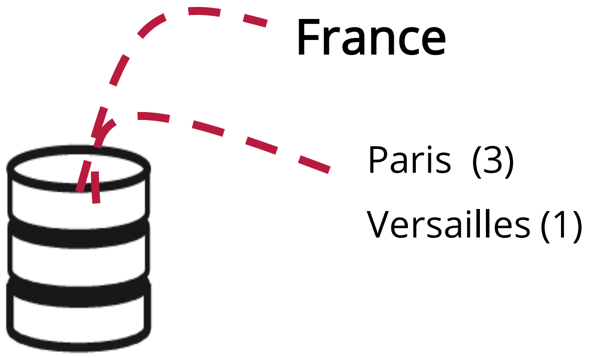
Open: Name or Pattern ....

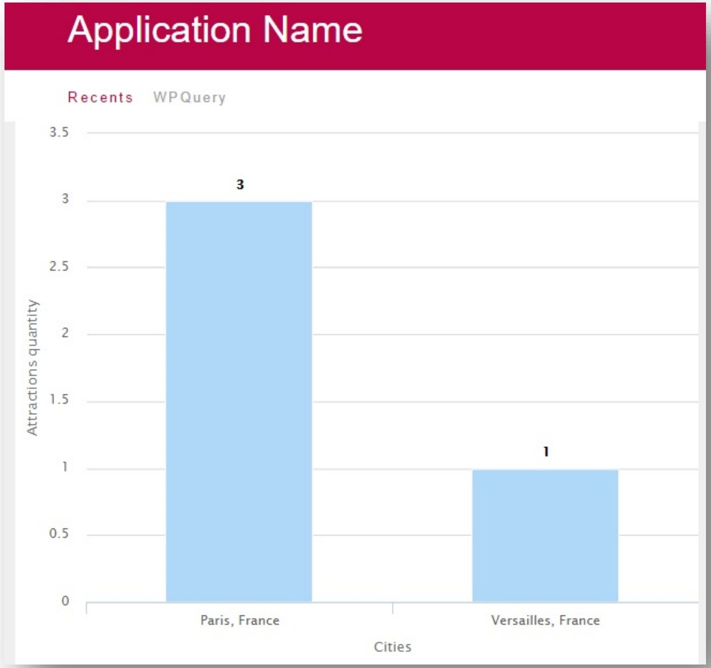
```
Travel_Agency
  Main Programs
  Root Module
    GeneXus
    Security
    Attraction
      WorkWithAttraction
        AttractionGeneral
        ViewAttraction
        WWAttraction
```

Dynamic Queries

Query Object

View the number of attractions of France, by city

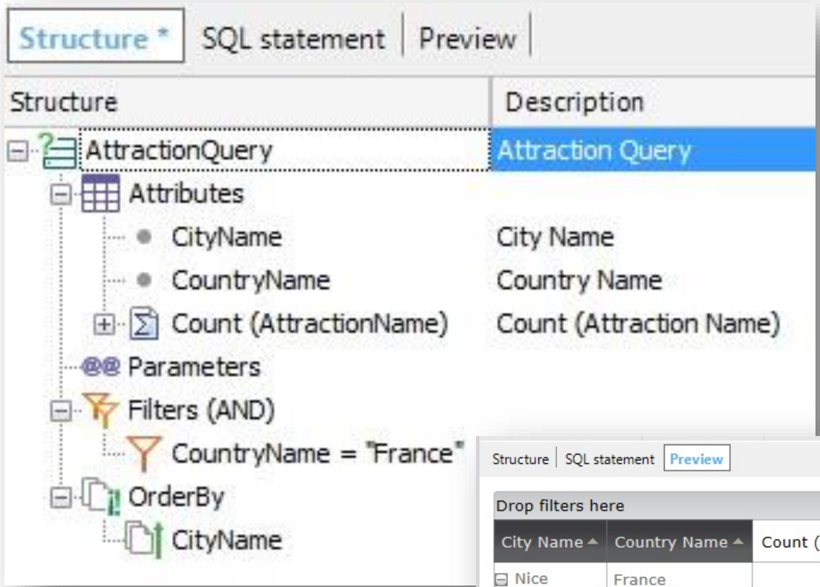




View the number of attractions of France, by city

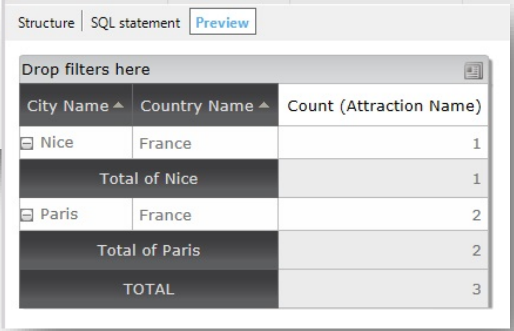
```
Country
{
  CountryId*
  CountryName
  City
  {
     CityId*
     CityName
   }
}
```

```
Attraction
{
  AttractionId*
  AttractionName
  CountryId
  CountryName
  CityId
  CityName
}
```



Structure * | SQL statement | Preview

Structure	Description
AttractionQuery	Attraction Query
Attributes	
CityName	City Name
CountryName	Country Name
Count (AttractionName)	Count (Attraction Name)
Parameters	
Filters (AND)	
CountryName = "France"	
OrderBy	
CityName	



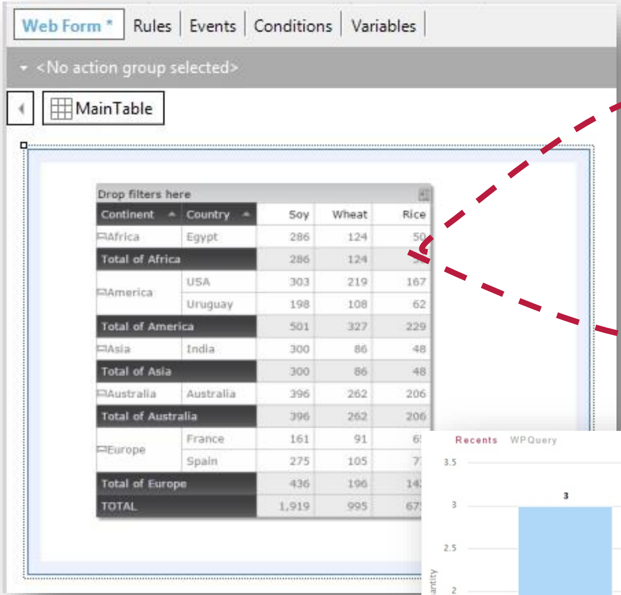
Structure | SQL statement | Preview

Drop filters here

City Name	Country Name	Count (Attraction Name)
Nice	France	1
Total of Nice		1
Paris	France	2
Total of Paris		2
TOTAL		3

Preview

Query viewer



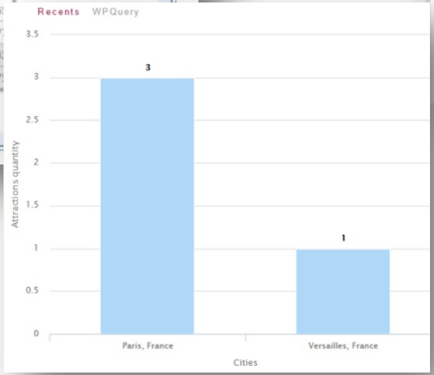
Web Form * | Rules | Events | Conditions | Variables

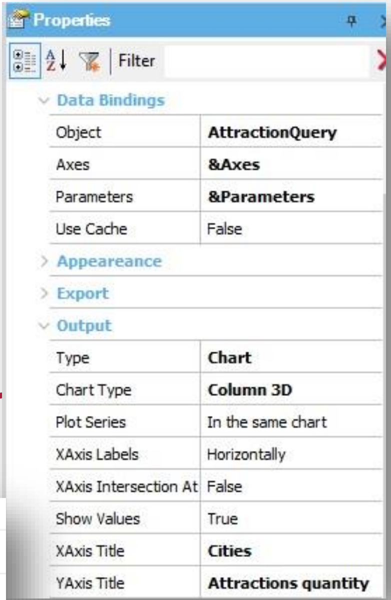
<No action group selected>

MainTable

Drop filters here

Continent	Country	Soy	Wheat	Rice
Africa	Egypt	286	124	50
Total of Africa		286	124	5
America	USA	303	219	167
	Uruguay	198	108	62
Total of America		501	327	229
Asia	India	300	86	48
Total of Asia		300	86	48
Australia	Australia	396	262	206
Total of Australia		396	262	206
Europe	France	161	91	6
	Spain	275	105	7
Total of Europe		436	196	14
TOTAL		1,919	995	67





Properties

Filter

Data Bindings

Object	AttractionQuery
Axes	&Axes
Parameters	&Parameters
Use Cache	False

Appearance

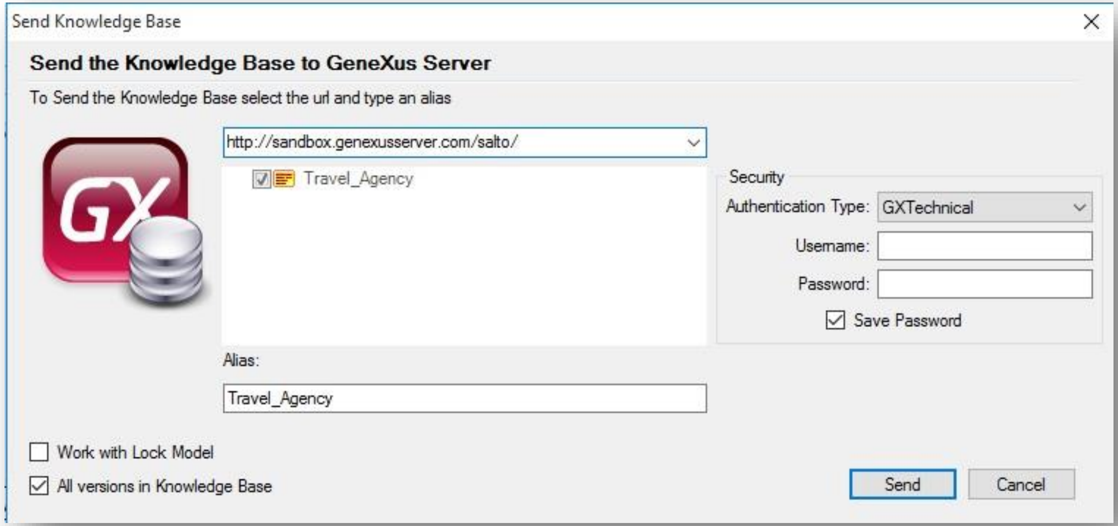
Export

Output

Type	Chart
Chart Type	Column 3D
Plot Series	In the same chart
XAxis Labels	Horizontally
XAxis Intersection At	False
Show Values	True
XAxis Title	Cities
YAxis Title	Attractions quantity

GeneXus Server

Send Knowledge Base to GeneXus Server



Send Knowledge Base

Send the Knowledge Base to GeneXus Server

To Send the Knowledge Base select the url and type an alias

http://sandbox.genexusserver.com/salto/

☑ Travel_Agency

Alias: Travel_Agency

Security

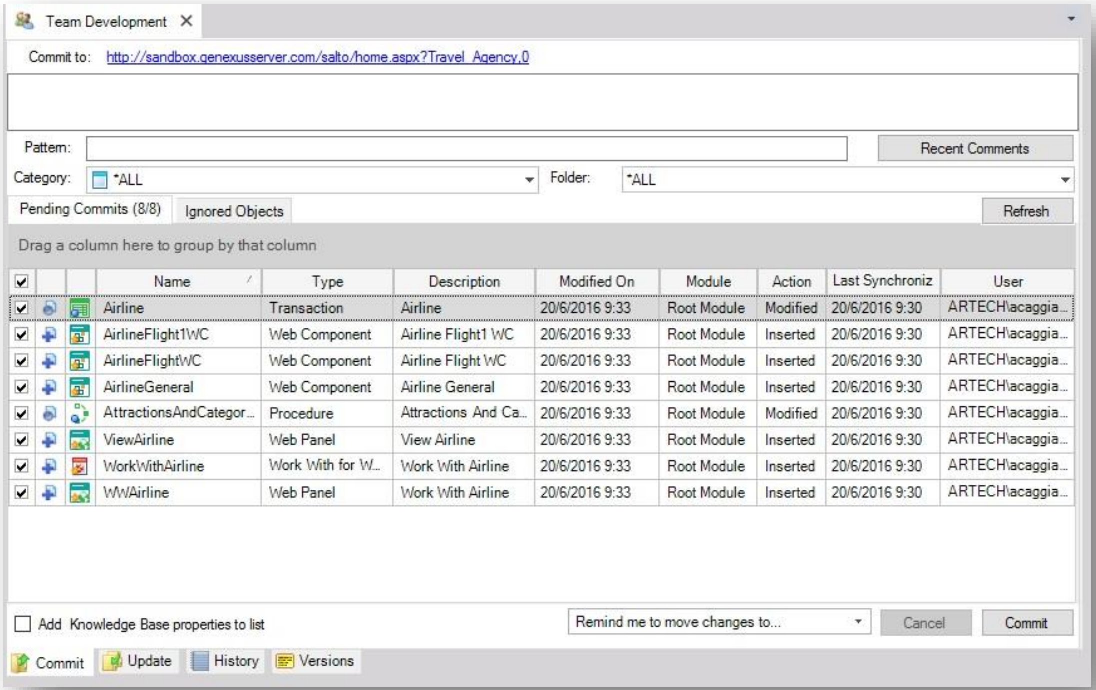
Authentication Type:	GXTechnical
Username:	
Password:	

☑ Save Password

☐ Work with Lock Model

☑ All versions in Knowledge Base

Team Development


Team Development ✕

Commit to: http://sandbox.genexusserver.com/salto/home.aspx?Travel_Agency,0

Pattern: | Recent Comments

Category: *ALL | Folder: *ALL

Pending Commits (8/8) | Ignored Objects | Refresh

Drag a column here to group by that column

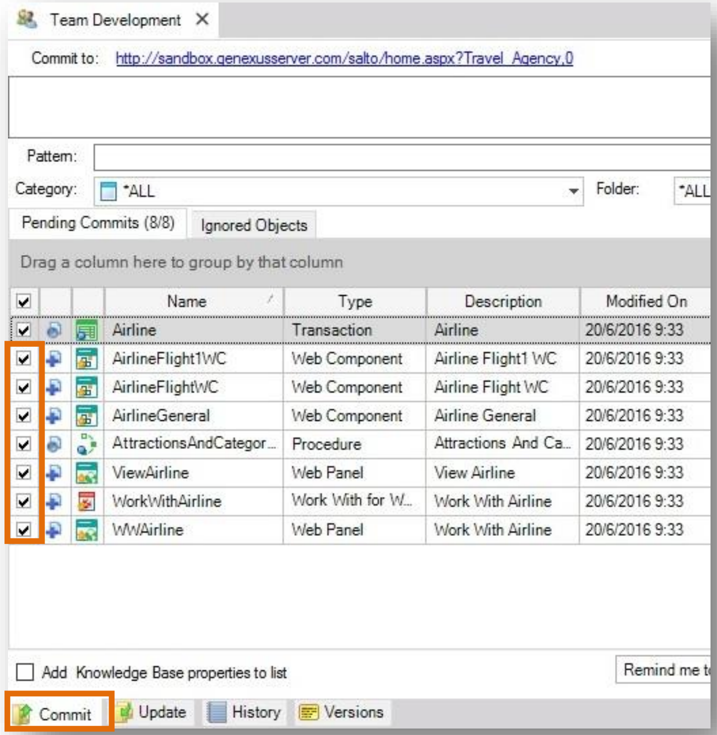
✓			Name	Type	Description	Modified On	Module	Action	Last Synchroniz	User
✓			Airline	Transaction	Airline	20/6/2016 9:33	Root Module	Modified	20/6/2016 9:30	ARTECH\acaggia...
✓			AirlineFlight1WC	Web Component	Airline Flight1 WC	20/6/2016 9:33	Root Module	Inserted	20/6/2016 9:30	ARTECH\acaggia...
✓			AirlineFlightWC	Web Component	Airline Flight WC	20/6/2016 9:33	Root Module	Inserted	20/6/2016 9:30	ARTECH\acaggia...
✓			AirlineGeneral	Web Component	Airline General	20/6/2016 9:33	Root Module	Inserted	20/6/2016 9:30	ARTECH\acaggia...
✓			AttractionsAndCategor...	Procedure	Attractions And Ca...	20/6/2016 9:33	Root Module	Modified	20/6/2016 9:30	ARTECH\acaggia...
✓			ViewAirline	Web Panel	View Airline	20/6/2016 9:33	Root Module	Inserted	20/6/2016 9:30	ARTECH\acaggia...
✓			WorkWithAirline	Work With for W...	Work With Airline	20/6/2016 9:33	Root Module	Inserted	20/6/2016 9:30	ARTECH\acaggia...
✓			WWAirline	Web Panel	Work With Airline	20/6/2016 9:33	Root Module	Inserted	20/6/2016 9:30	ARTECH\acaggia...

☐ Add Knowledge Base properties to list | Remind me to move changes to... | Cancel | Commit

Commit

Knowledge Manager / Team Development

Partial commit



Team Development

Commit to: http://sandbox.genexusserver.com/salto/home.aspx?Travel_Agency,0

Pattern:

Category: *ALL Folder: *ALL

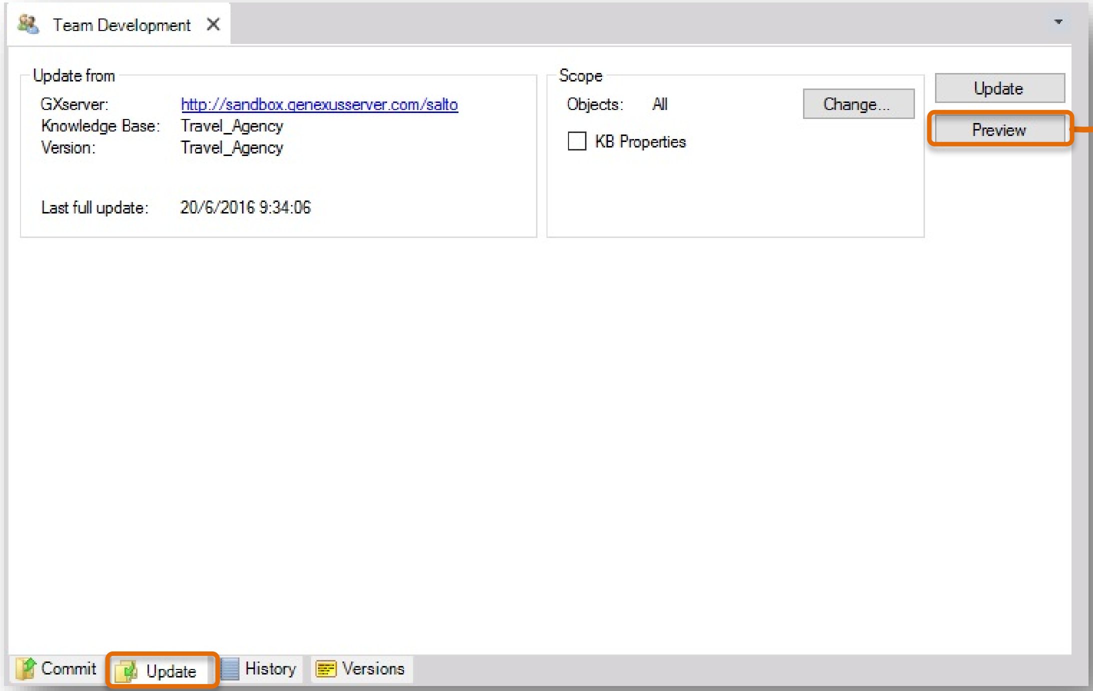
Pending Commits (8/8) Ignored Objects

Drag a column here to group by that column

✓			Name	Type	Description	Modified On
✓			Airline	Transaction	Airline	20/6/2016 9:33
✓			AirlineFlight1WC	Web Component	Airline Flight1 WC	20/6/2016 9:33
✓			AirlineFlightWC	Web Component	Airline Flight WC	20/6/2016 9:33
✓			AirlineGeneral	Web Component	Airline General	20/6/2016 9:33
✓			AttractionsAndCategor...	Procedure	Attractions And Ca...	20/6/2016 9:33
✓			ViewAirline	Web Panel	View Airline	20/6/2016 9:33
✓			WorkWithAirline	Work With for W...	Work With Airline	20/6/2016 9:33
✓			WWAirline	Web Panel	Work With Airline	20/6/2016 9:33

☐ Add Knowledge Base properties to list Remind me t

Update



Update from

GXserver: http://sandbox.genexusserver.com/salto
Knowledge Base: Travel_Agency
Version: Travel_Agency

Last full update: 20/6/2016 9:34:06

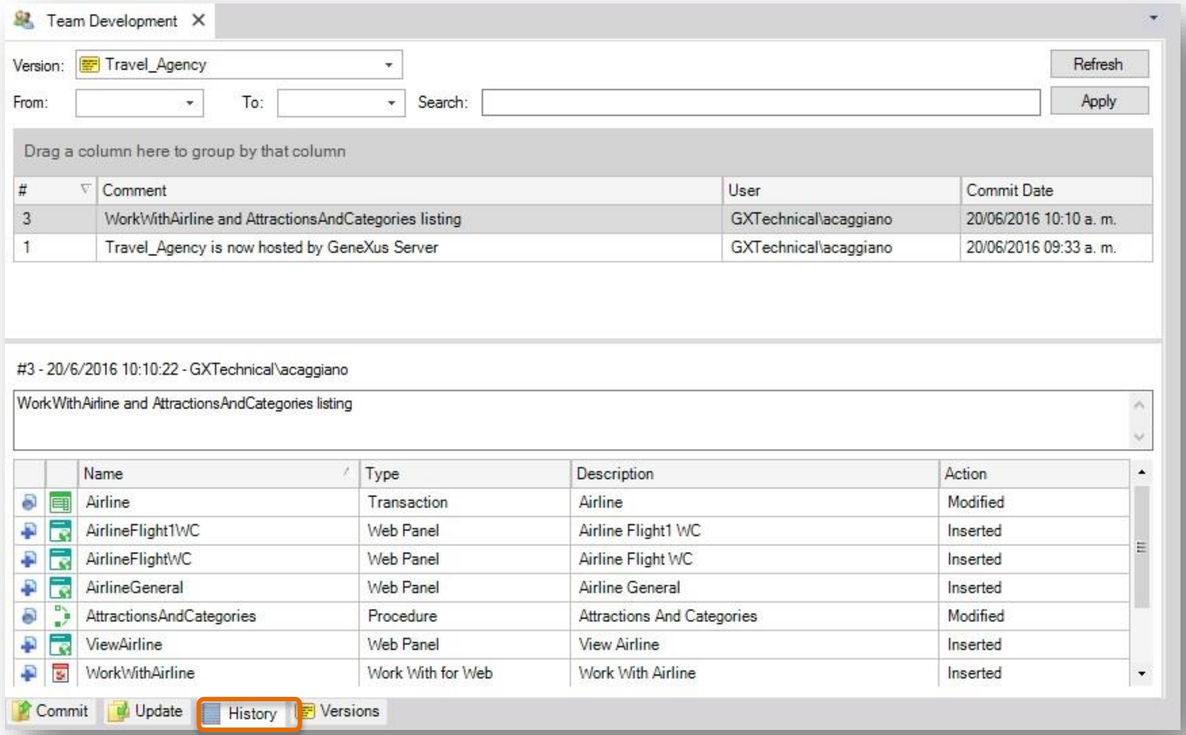
Scope

Objects: All

☐ KB Properties

Partial selection

History



Version: Travel_Agency

From: To: Search:

Drag a column here to group by that column

#	Comment	User	Commit Date
3	WorkWithAirline and AttractionsAndCategories listing	GXTechnical\acaggiano	20/06/2016 10:10 a. m.
1	Travel_Agency is now hosted by GeneXus Server	GXTechnical\acaggiano	20/06/2016 09:33 a. m.

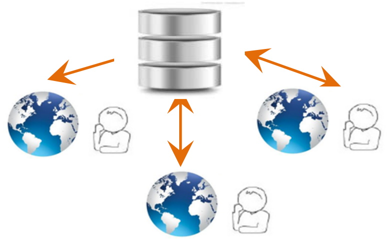
#3 - 20/6/2016 10:10:22 - GXTechnical\acaggiano

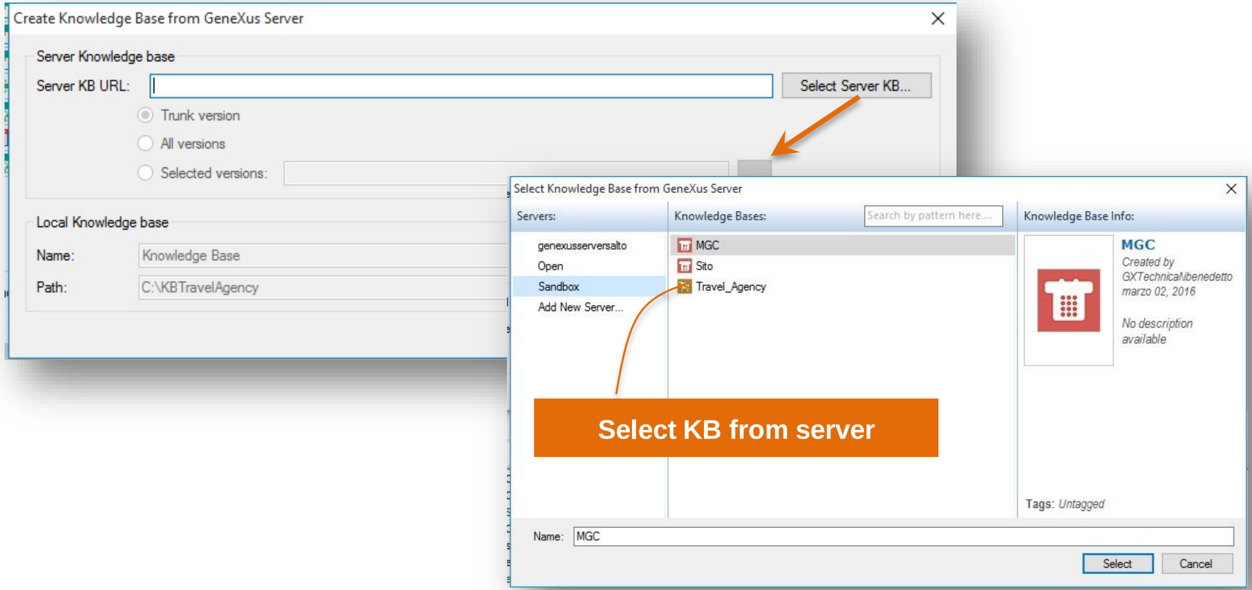
WorkWithAirline and AttractionsAndCategories listing

Name	Type	Description	Action
Airline	Transaction	Airline	Modified
AirlineFlight1WC	Web Panel	Airline Flight1 WC	Inserted
AirlineFlightWC	Web Panel	Airline Flight WC	Inserted
AirlineGeneral	Web Panel	Airline General	Inserted
AttractionsAndCategories	Procedure	Attractions And Categories	Modified
ViewAirline	Web Panel	View Airline	Inserted
WorkWithAirline	Work With for Web	Work With Airline	Inserted

Create KB from GeneXus Server

File / New / Knowledge Base from Server





Create Knowledge Base from GeneXus Server

Server Knowledge base

Server KB URL: | Select Server KB...

Trunk version

All versions

Selected versions:

Local Knowledge base

Name: Knowledge Base

Path: C:\KBTravelAgency

Select Knowledge Base from GeneXus Server

Servers: genexusserversalto, Open, Sandbox, Add New Server...

Knowledge Bases: Search by pattern here....

MGC

Sito

Travel_Agency

Knowledge Base Info:

MGC

Created by GXTechnicalVbenedetto marzo 02, 2016

No description available

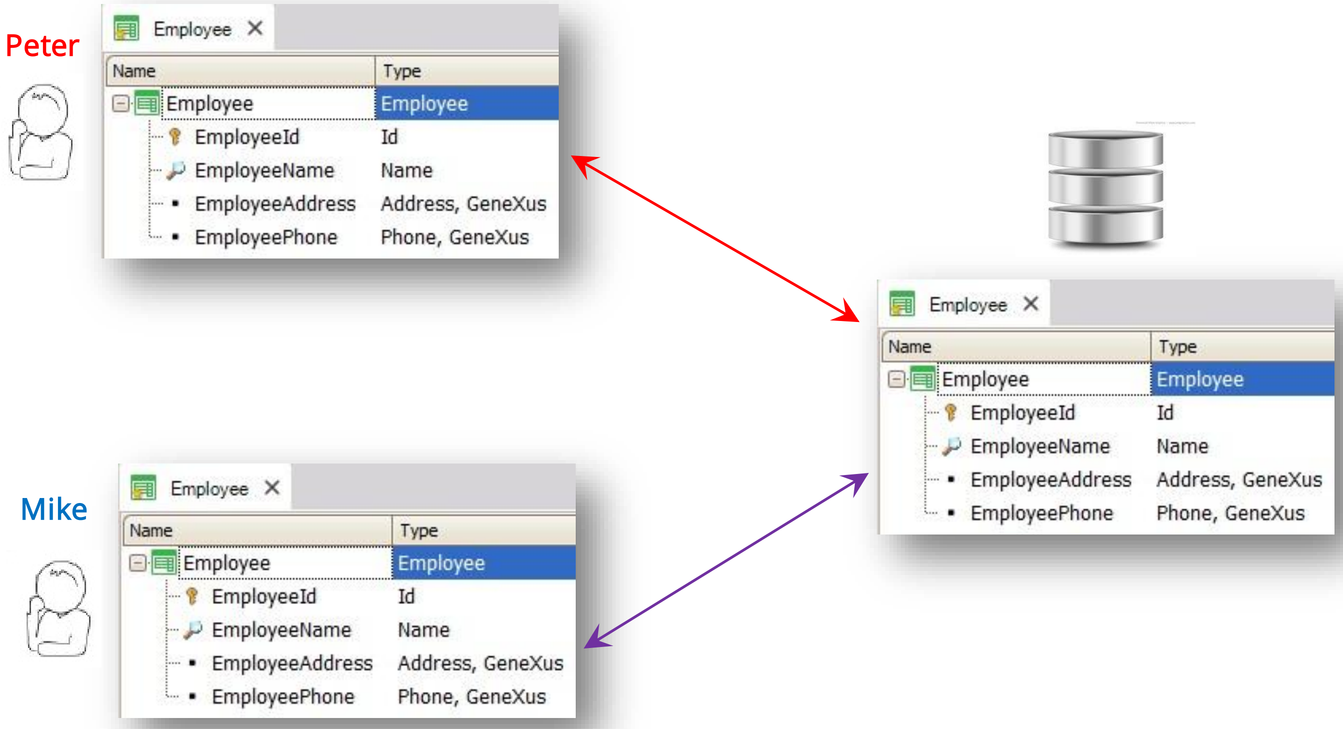
Tags: Untagged

Name: MGC

Select | Cancel

Select KB from server

Example



Employee ✕

Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus

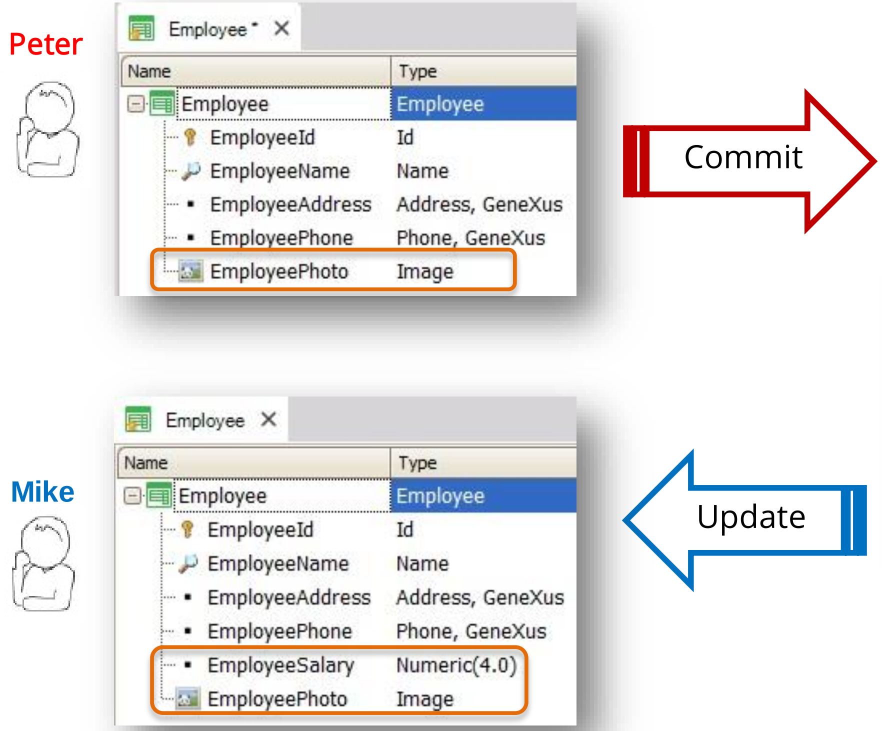
Employee ✕

Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus

Employee ✕

Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus

Example

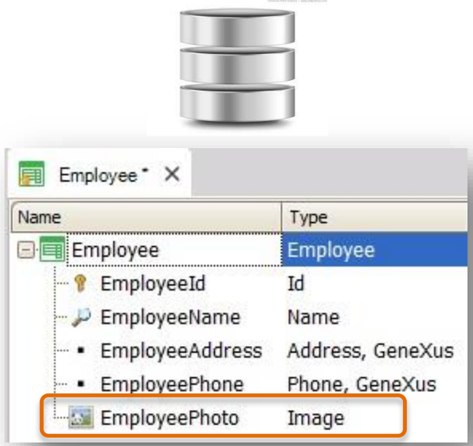


Employee *

Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus
EmployeePhoto	Image

Employee

Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus
EmployeeSalary	Numeric(4.0)
EmployeePhoto	Image

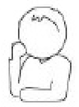


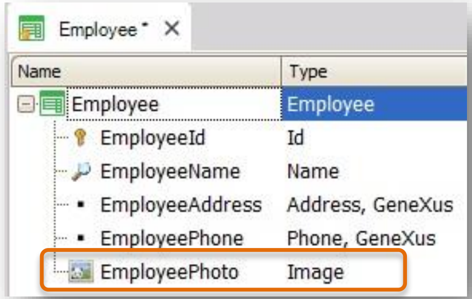
Employee *

Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus
EmployeePhoto	Image

Example

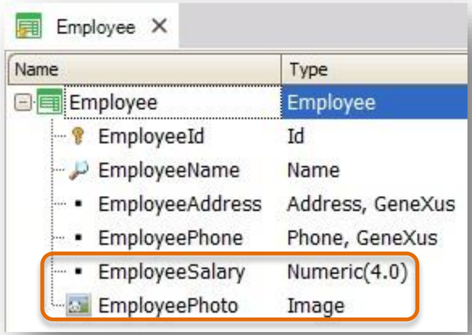
Peter





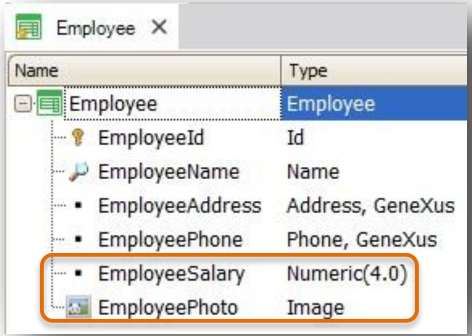
Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus
EmployeePhoto	Image

Mike



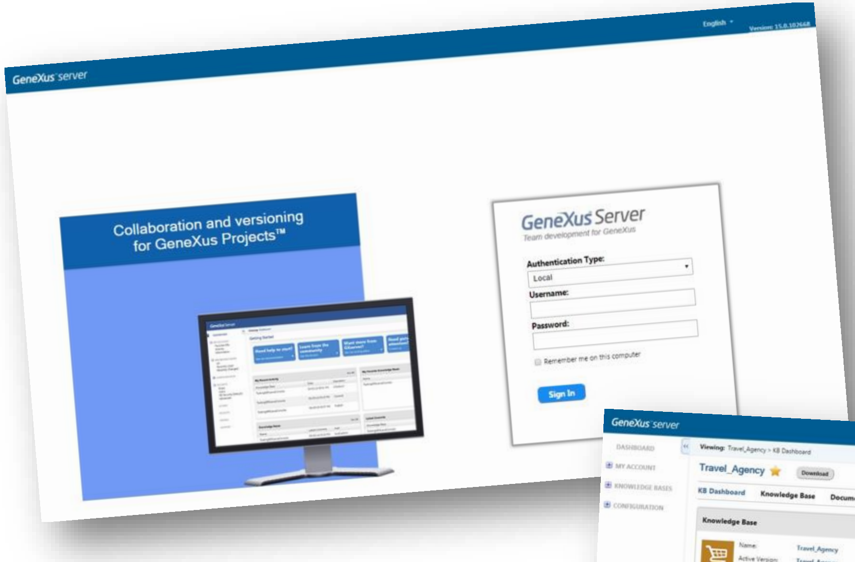
Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus
EmployeeSalary	Numeric(4.0)
EmployeePhoto	Image

Commit



Name	Type
Employee	Employee
EmployeeId	Id
EmployeeName	Name
EmployeeAddress	Address, GeneXus
EmployeePhone	Phone, GeneXus
EmployeeSalary	Numeric(4.0)
EmployeePhoto	Image

Web console

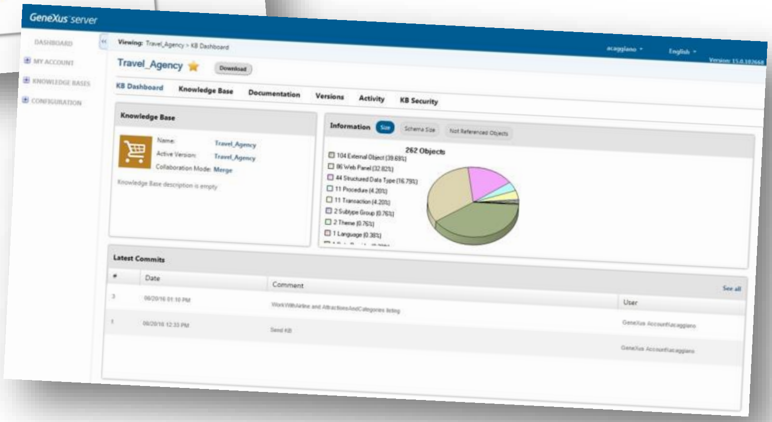

GeneXus server
English
Version: 15.0.102668
Collaboration and versioning for GeneXus Projects™
GeneXus Server
Team development for GeneXus
Sign In

Authentication Type: Local

Username:

Password:

Remember me on this computer



Viewing: Travel_Agency > KB Dashboard

Travel_Agency

Knowledge Base

Name: Travel_Agency
Active Version: Travel_Agency
Collaboration Mode: Merge

Knowledge Base description is empty

Information

262 Objects

- 104 External Object (39.69%)
- 86 Web Panel (32.82%)
- 44 Structured Data Type (16.79%)
- 11 Procedure (4.20%)
- 11 Transaction (4.20%)
- 2 Subtype Group (0.76%)
- 2 Theme (0.76%)
- 1 Language (0.38%)

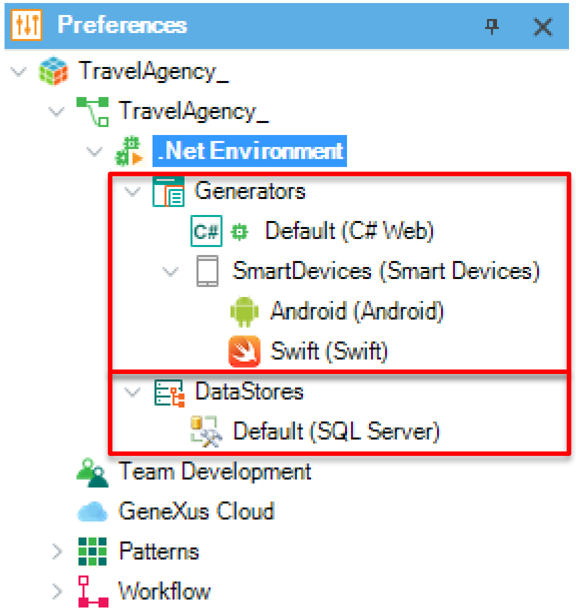
Latest Commits

#	Date	Comment	User
3	06/20/16 01:10 PM	WorkWithAirline and AttractionsAndCategories listing	GeneXus Account\acaggiano
1	06/20/16 12:33 PM	Send KB	GeneXus Account\acaggiano

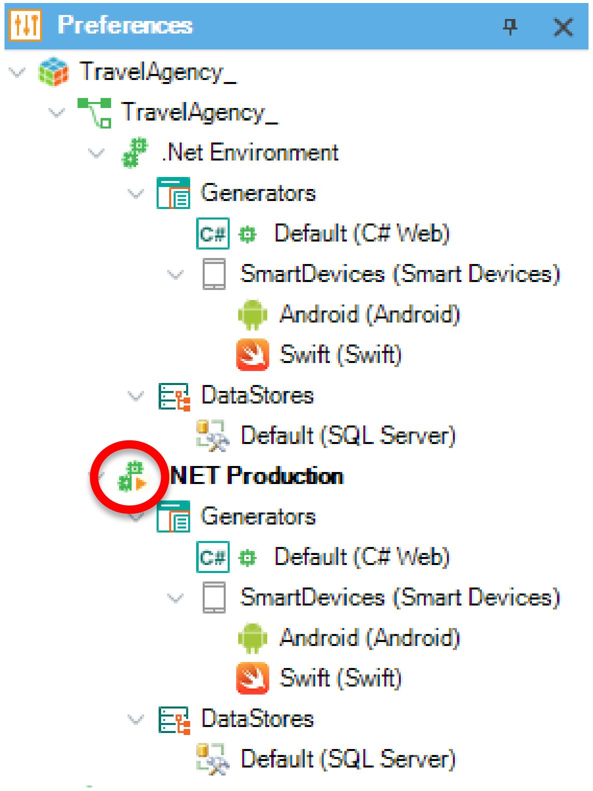
See all

Execution environments

Generators and databases


Preferences
- TravelAgency_
  - TravelAgency_
    - .Net Environment
      - Generators
        - Default (C# Web)
        - SmartDevices (Smart Devices)
          - Android (Android)
          - Swift (Swift)
      - DataStores
        - Default (SQL Server)
  - Team Development
  - GeneXus Cloud
  - Patterns
  - Workflow

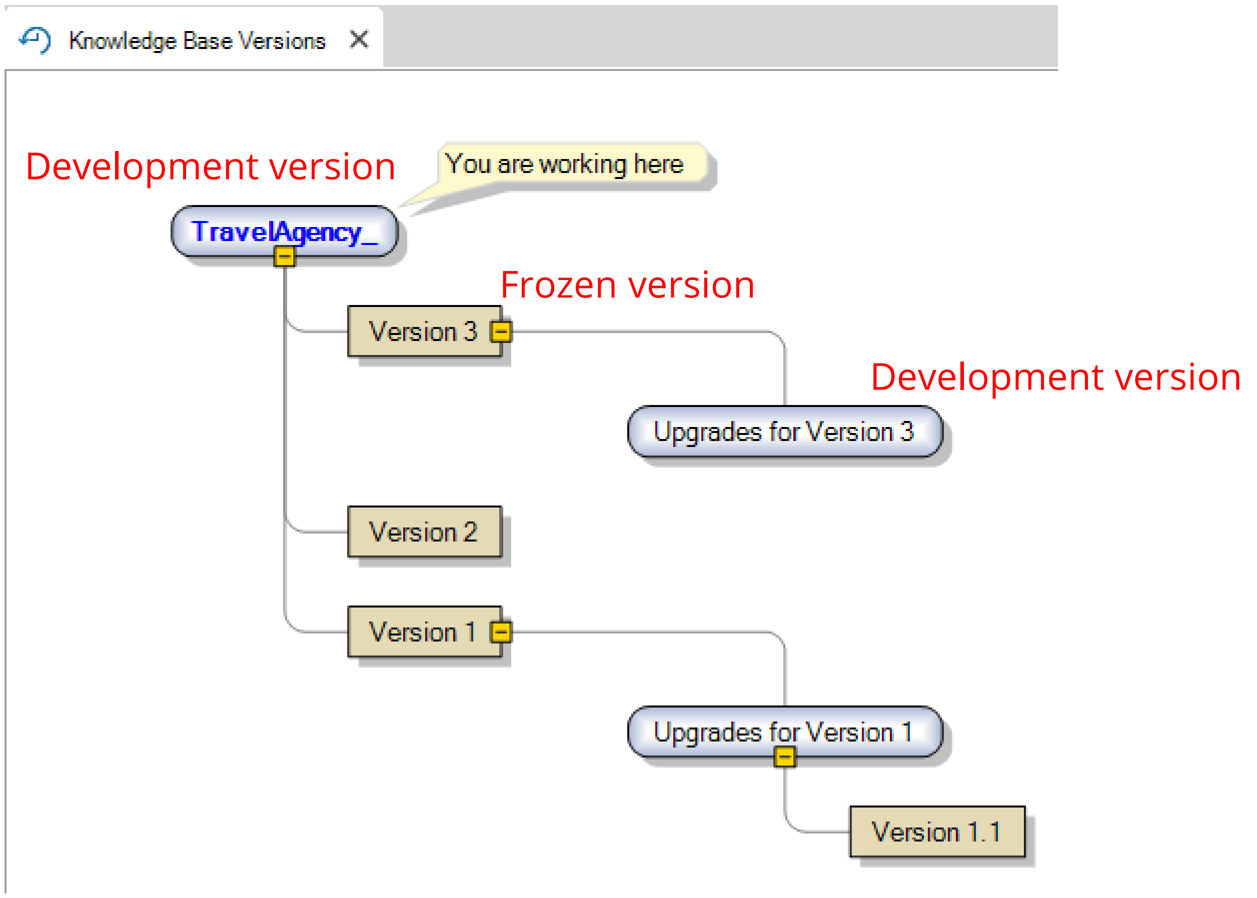
Working with more than one environment


Preferences

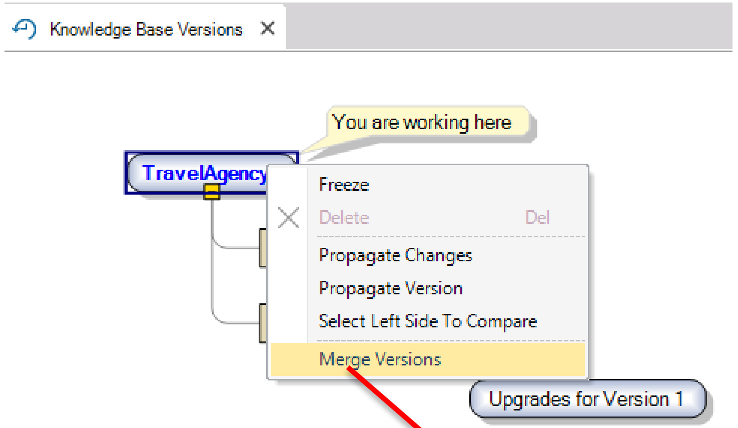
TravelAgency_
- TravelAgency_
  - .Net Environment
    - Generators
      - Default (C# Web)
      - SmartDevices (Smart Devices)
        - Android (Android)
        - Swift (Swift)
    - DataStores
      - Default (SQL Server)
  - .NET Production
    - Generators
      - Default (C# Web)
      - SmartDevices (Smart Devices)
        - Android (Android)
        - Swift (Swift)
    - DataStores
      - Default (SQL Server)

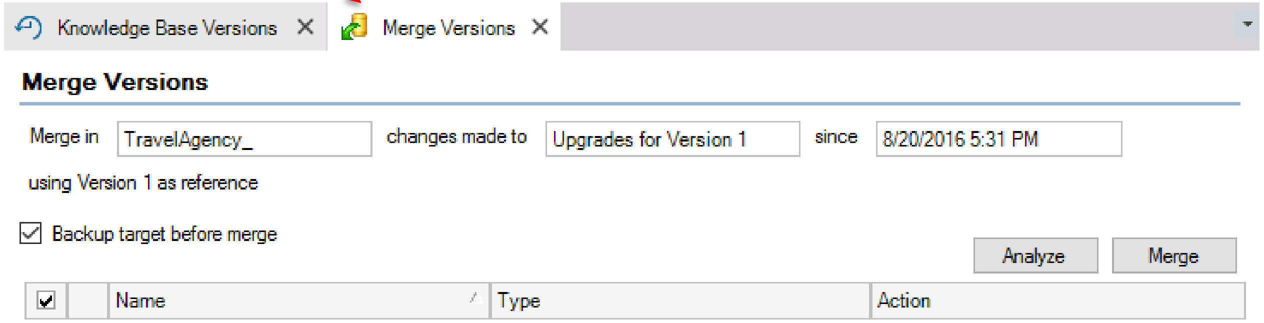
Knowledge Base versioning

Versions tree



To synchronize (merge) two development versions:





Merge Versions

Merge in TravelAgency_ changes made to Upgrades for Version 1 since 8/20/2016 5:31 PM

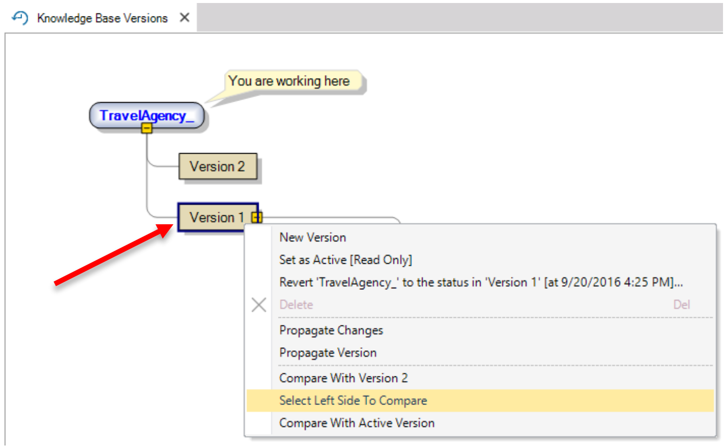
using Version 1 as reference

☑ Backup target before merge

☑		Name	Type	Action

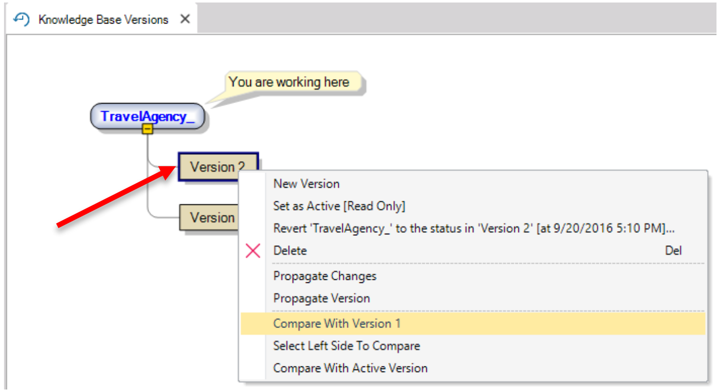
To compare two versions:

1)



New Version
Set as Active [Read Only]
Revert 'TravelAgency_' to the status in 'Version 1' [at 9/20/2016 4:25 PM]...
Delete Del
Propagate Changes
Propagate Version
Compare With Version 2
Select Left Side To Compare
Compare With Active Version

2)



New Version
Set as Active [Read Only]
Revert 'TravelAgency_' to the status in 'Version 2' [at 9/20/2016 5:10 PM]...
Delete Del
Propagate Changes
Propagate Version
Compare With Version 1
Select Left Side To Compare
Compare With Active Version

Security with GAM

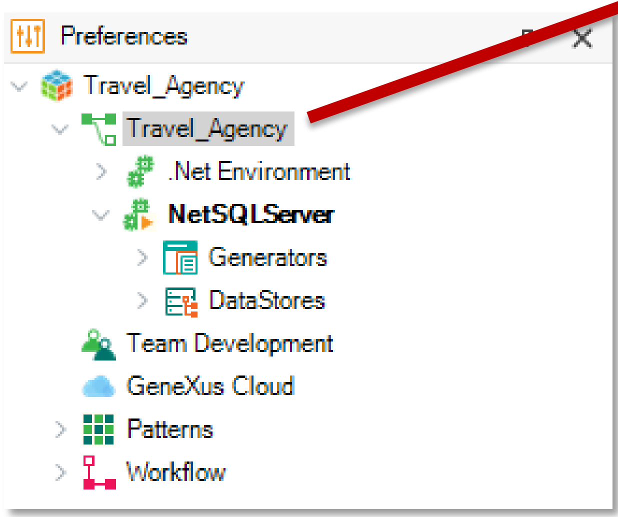
AUTHENTICATION

AUTHORIZATION

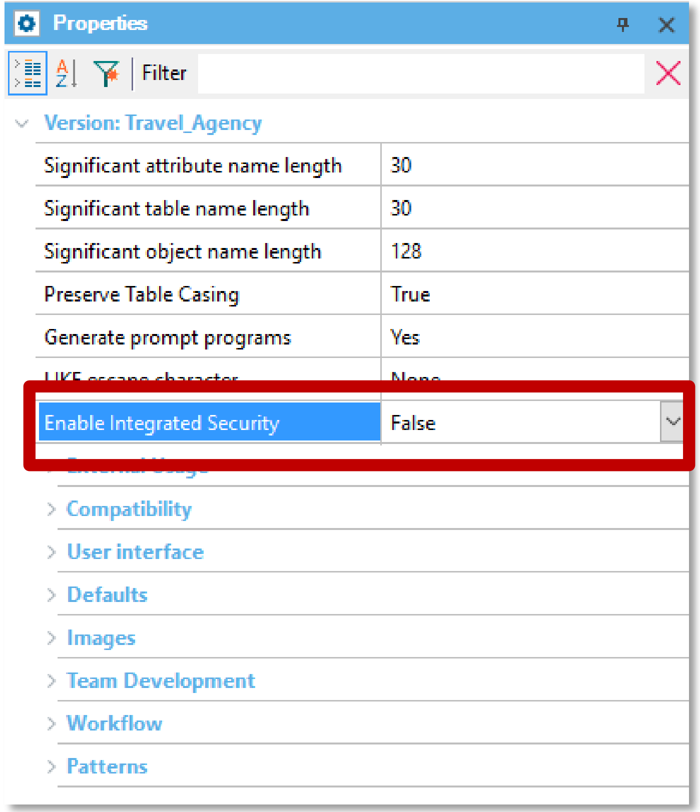
GeneXus™
ACCESS MANAGER

GeneXus™

Enabling the GeneXus Access Manager


Preferences

- Travel_Agency
  - Travel_Agency
    - .Net Environment
    - NetSQLServer
      - Generators
      - DataStores
    - Team Development
    - GeneXus Cloud
    - Patterns
    - Workflow


Properties

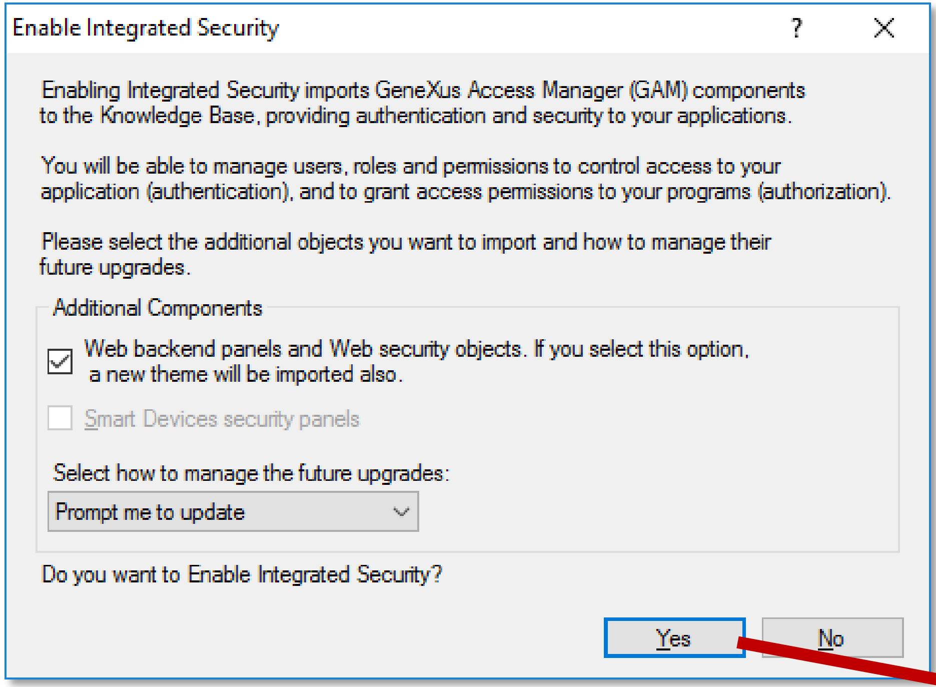
Filter

Version: Travel_Agency

Significant attribute name length	30
Significant table name length	30
Significant object name length	128
Preserve Table Casing	True
Generate prompt programs	Yes
LIKE escape character	None
Enable Integrated Security	False

- External Usage
- Compatibility
- User interface
- Defaults
- Images
- Team Development
- Workflow
- Patterns

Importing objects from GAM


Enable Integrated Security

Enabling Integrated Security imports GeneXus Access Manager (GAM) components to the Knowledge Base, providing authentication and security to your applications.

You will be able to manage users, roles and permissions to control access to your application (authentication), and to grant access permissions to your programs (authorization).

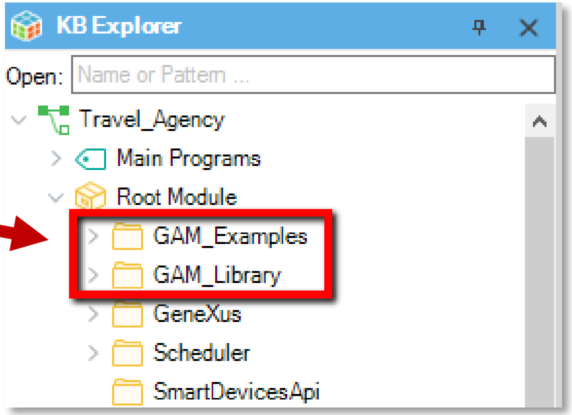
Please select the additional objects you want to import and how to manage their future upgrades.

Additional Components

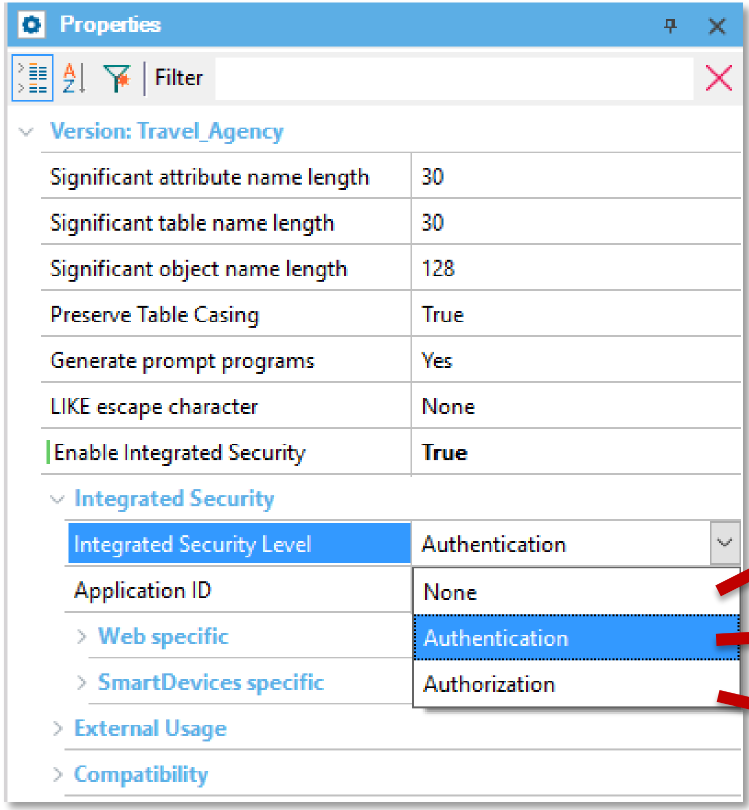
- [✓] Web backend panels and Web security objects. If you select this option, a new theme will be imported also.
- [ ] Smart Devices security panels

Select how to manage the future upgrades: Prompt me to update

Do you want to Enable Integrated Security?



Selecting integretind security level



Properties

Filter

Version: Travel_Agency

Significant attribute name length	30
Significant table name length	30
Significant object name length	128
Preserve Table Casing	True
Generate prompt programs	Yes
LIKE escape character	None
Enable Integrated Security	**True**

Integrated Security

Integrated Security Level	Authentication
Application ID	

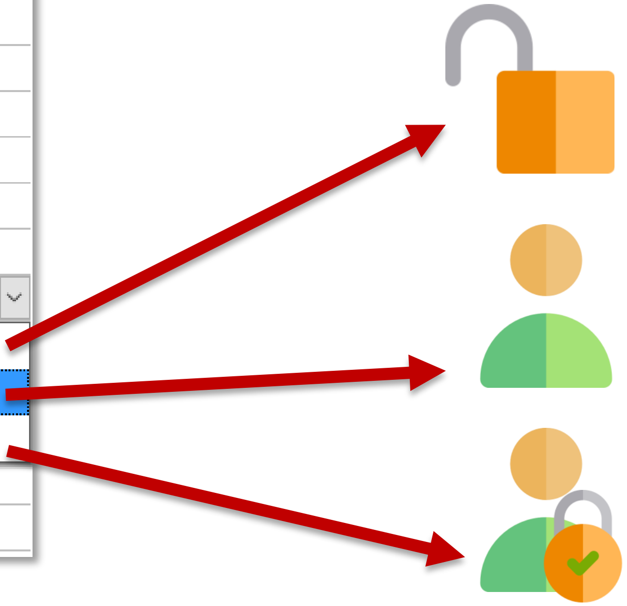
Options: None / Authentication / Authorization

Web specific

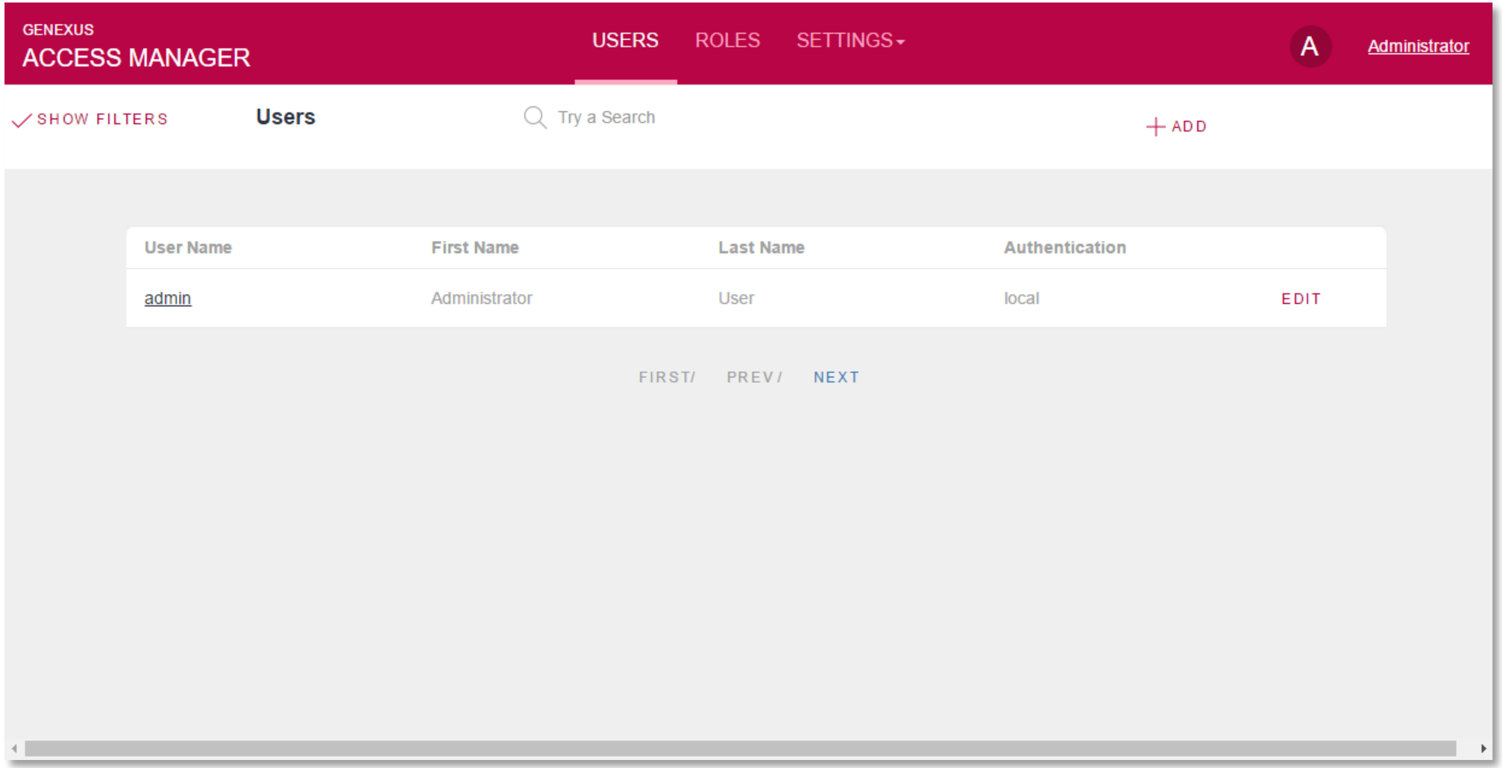
SmartDevices specific

External Usage

Compatibility



Accesing to GAM backend (GAM Home object)



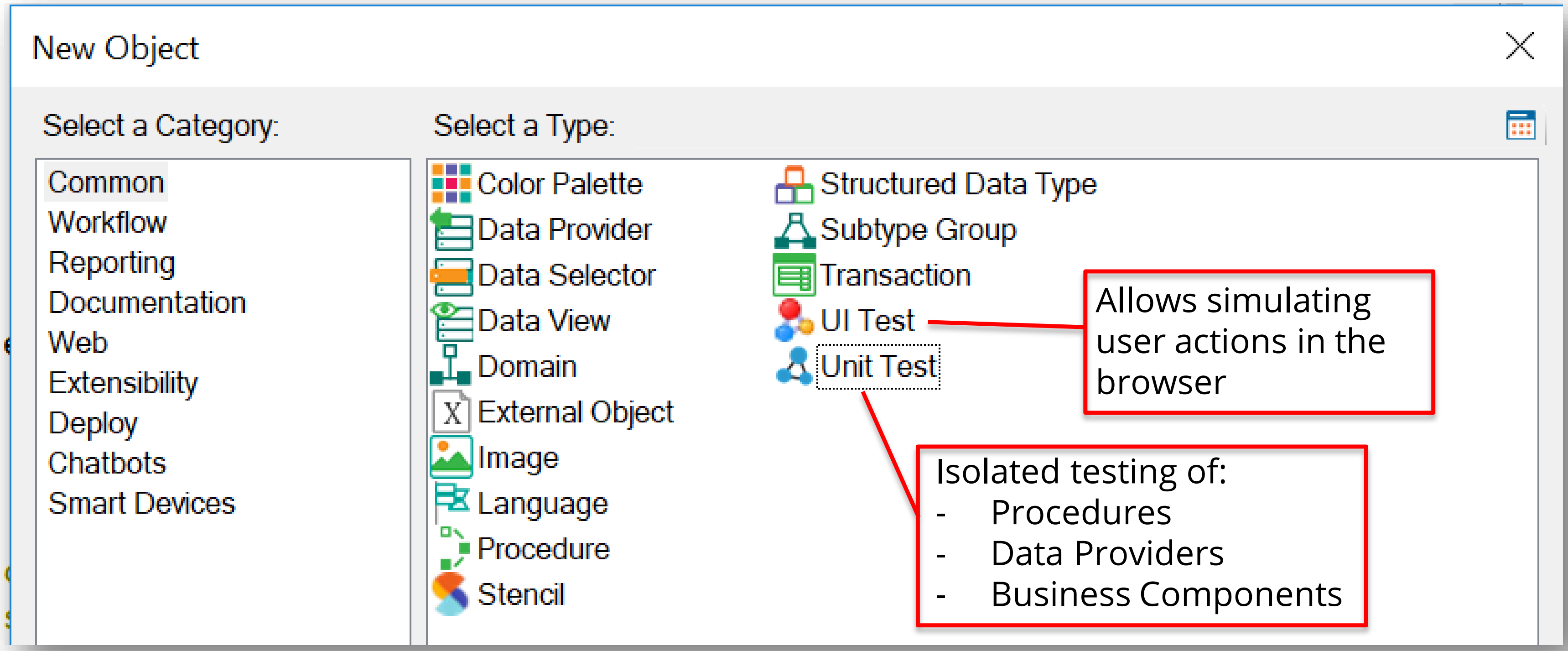
Users

User Name	First Name	Last Name	Authentication	
admin	Administrator	User	local	EDIT

Testing

GeneXus™

Objects to generate unit tests and interface tests


New Object

Select a Category:

- Common
- Workflow
- Reporting
- Documentation
- Web
- Extensibility
- Deploy
- Chatbots
- Smart Devices

Select a Type:

- Color Palette
- Data Provider
- Data Selector
- Data View
- Domain
- External Object
- Image
- Language
- Procedure
- Stencil
- Structured Data Type
- Subtype Group
- Transaction
- UI Test
- Unit Test

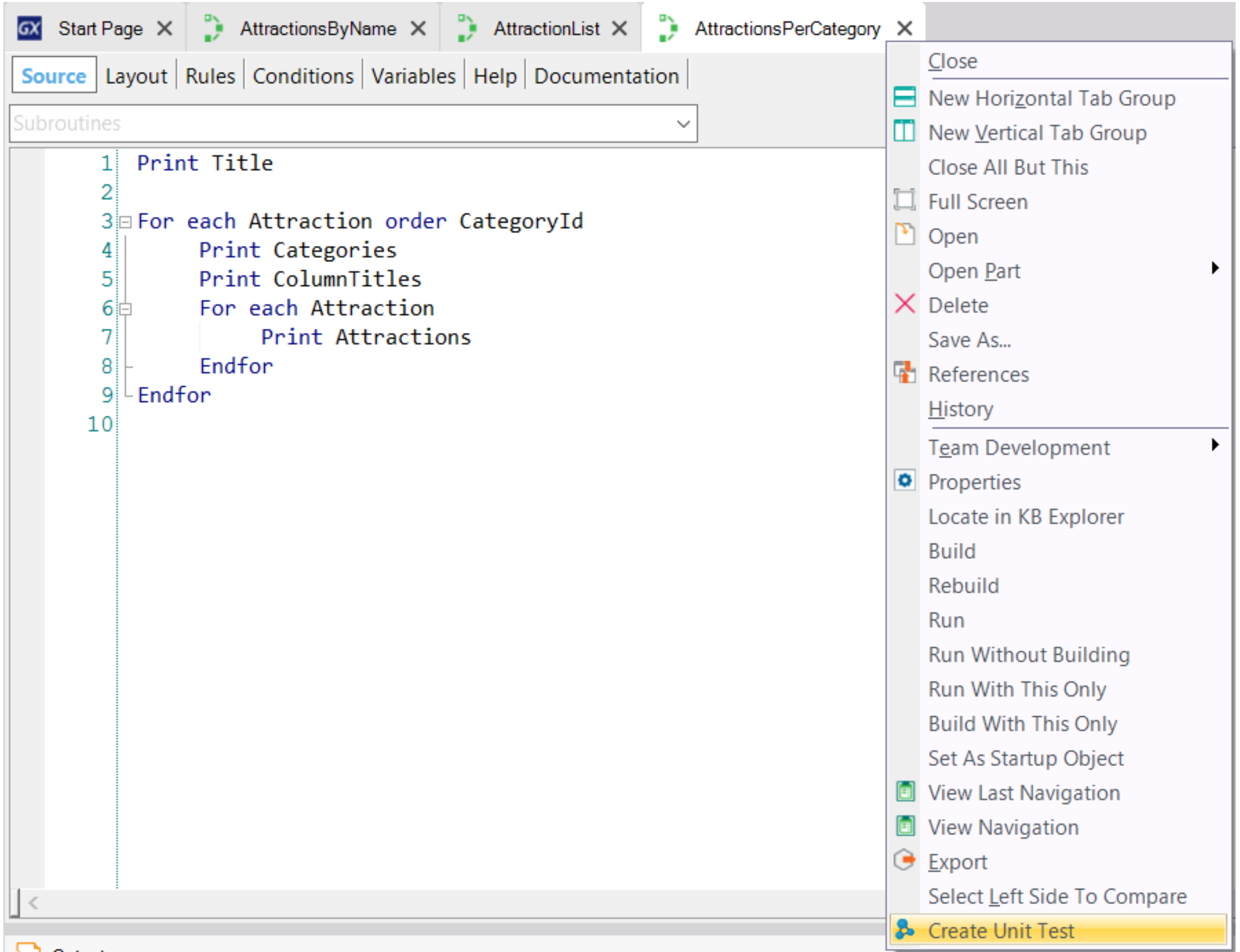
Allows simulating user actions in the browser

Isolated testing of:
- Procedures
- Data Providers
- Business Components

Unit Test

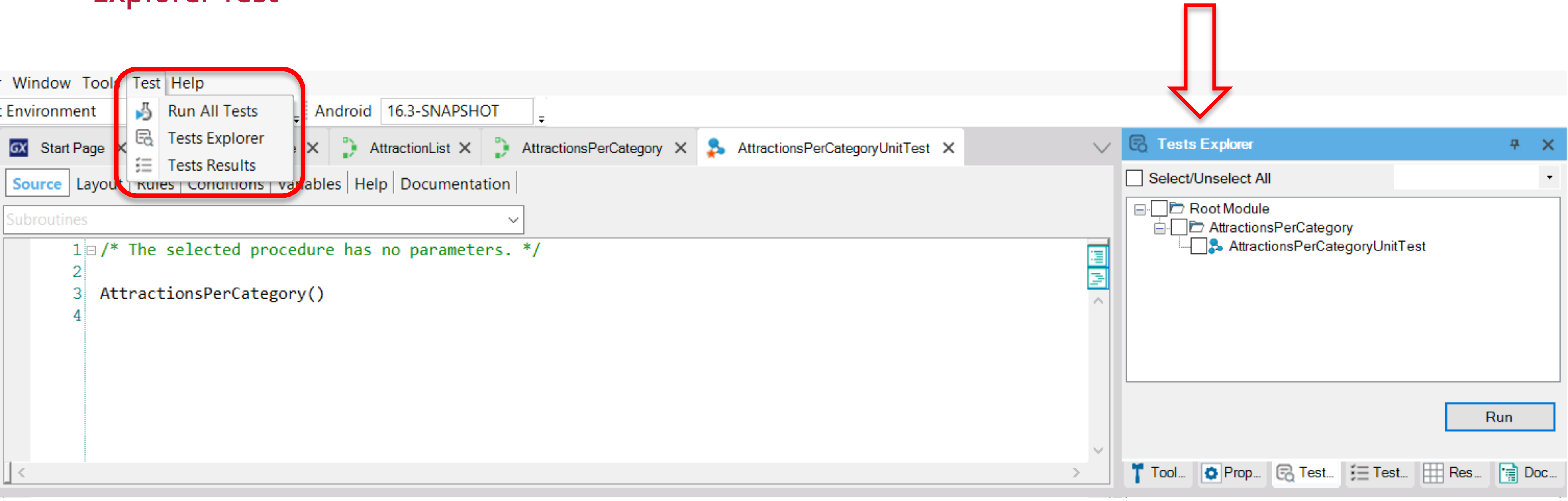
These objects are created:

- <ObjectName>UnitTest
- <ObjectName>UnitTestSDT
- <ObjectName>UnitTestData



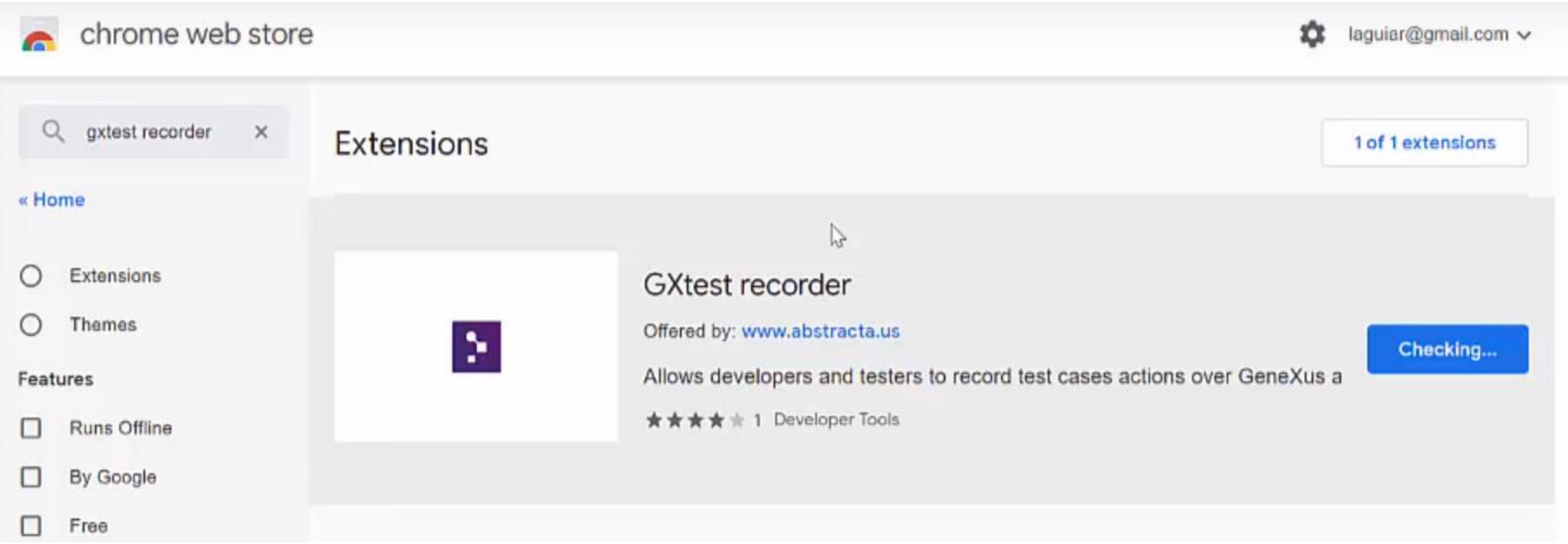
```
Print Title

For each Attraction order CategoryId
    Print Categories
    Print ColumnTitles
    For each Attraction
        Print Attractions
    Endfor
Endfor
```

Explorer Test



```
/* The selected procedure has no parameters. */

AttractionsPerCategory()
```

Interface Test: GXTest



Extensions

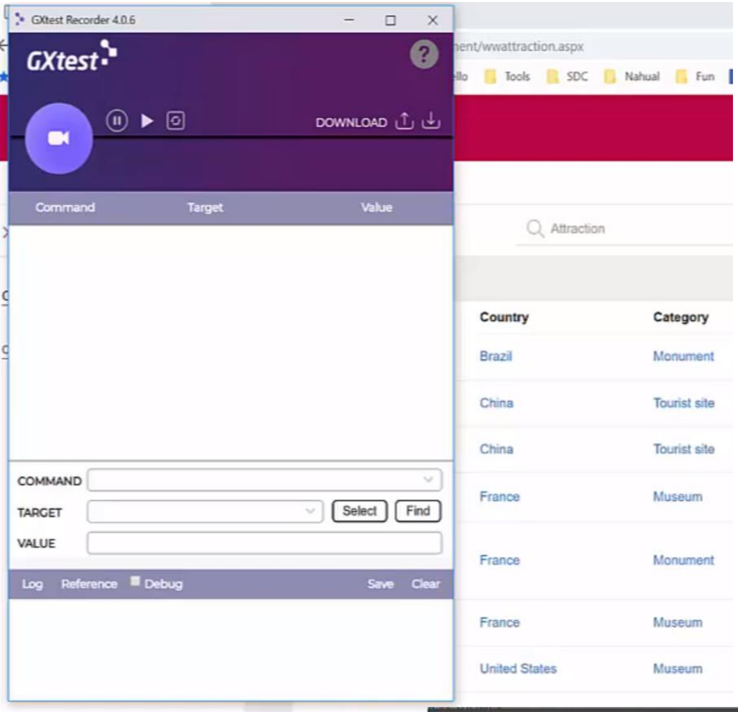
GXtest recorder

Offered by: www.abstracta.us

Allows developers and testers to record test cases actions over GeneXus a

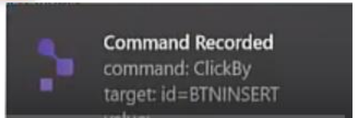
★★★★☆ 1 Developer Tools

Interface Test

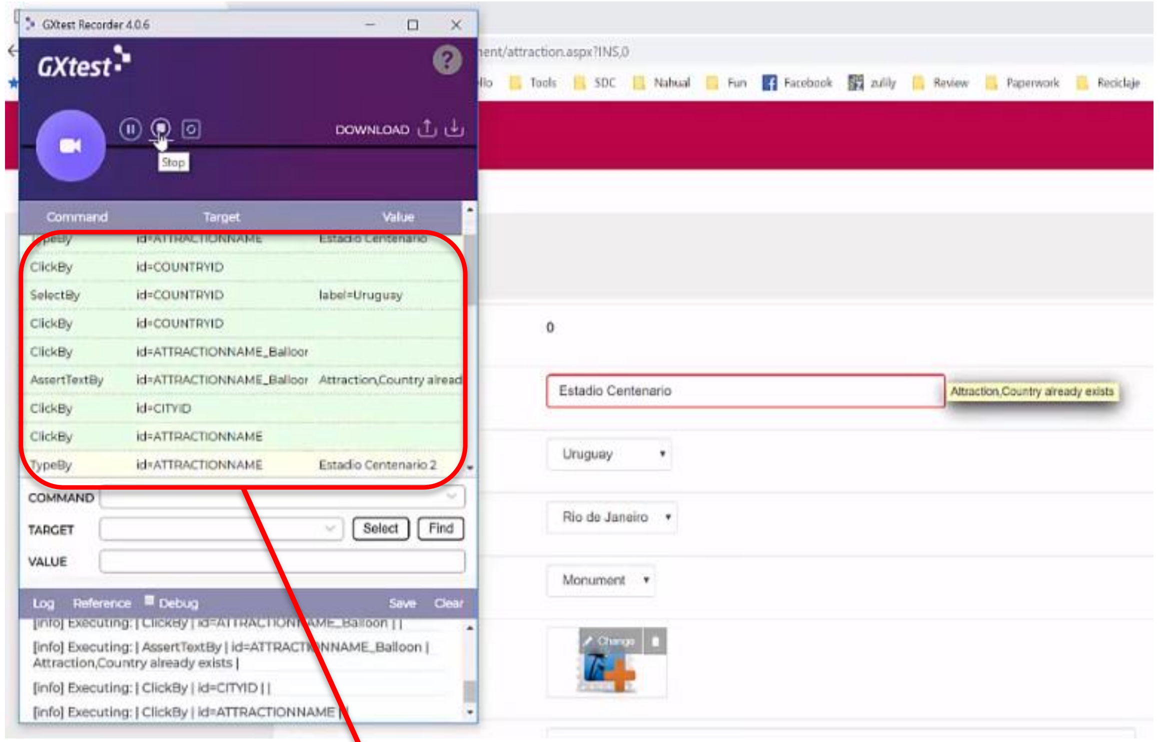


Command	Target	Value

Country	Category
Brazil	Monument
China	Tourist site
China	Tourist site
France	Museum
France	Monument
France	Museum
United States	Museum



Command Recorded
command: ClickBy
target: id=BTNINSERT



Command	Target	Value
TypeBy	id=ATTRACTIONNAME	Estadio Centenario
ClickBy	id=COUNTRYID	
SelectBy	id=COUNTRYID	label=Uruguay
ClickBy	id=COUNTRYID	
ClickBy	id=ATTRACTIONNAME_Balloon	
AssertTextBy	id=ATTRACTIONNAME_Balloon	Attraction,Country alread
ClickBy	id=CITYID	
ClickBy	id=ATTRACTIONNAME	
TypeBy	id=ATTRACTIONNAME	Estadio Centenario 2

```
[info] Executing: | ClickBy | id=ATTRACTIONNAME_Balloon | |
[info] Executing: | AssertTextBy | id=ATTRACTIONNAME_Balloon |
Attraction,Country already exists |
[info] Executing: | ClickBy | id=CITYID | |
[info] Executing: | ClickBy | id=ATTRACTIONNAME |
```

0

Estadio Centenario

Attraction,Country already exists

Uruguay

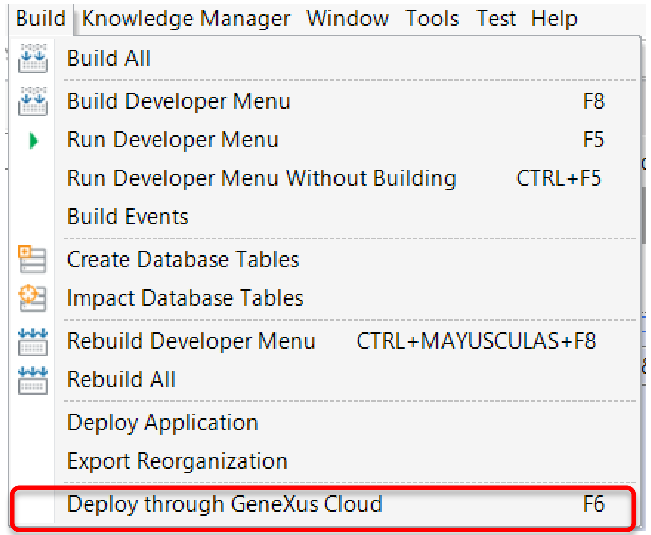
Rio de Janeiro

Monument

UI Test (GXTest license required)

Deployment

Automatic deployment with F6


Build | Knowledge Manager | Window | Tools | Test | Help

Build All

Build Developer Menu F8

Run Developer Menu F5

Run Developer Menu Without Building CTRL+F5

Build Events

Create Database Tables

Impact Database Tables

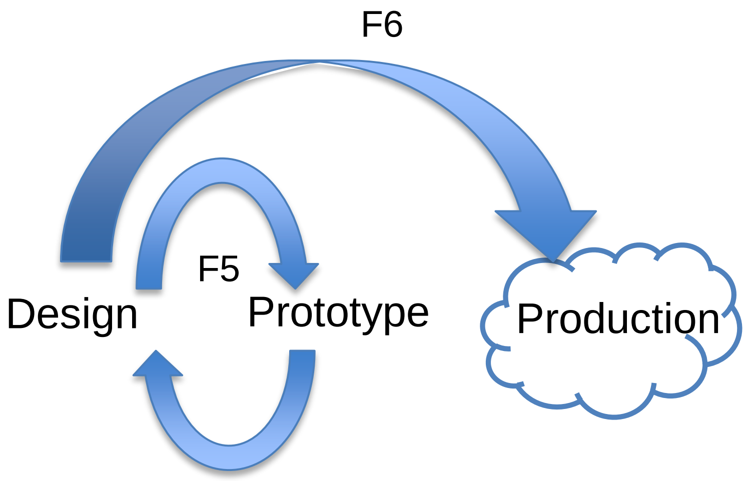
Rebuild Developer Menu CTRL+MAYUSCULAS+F8

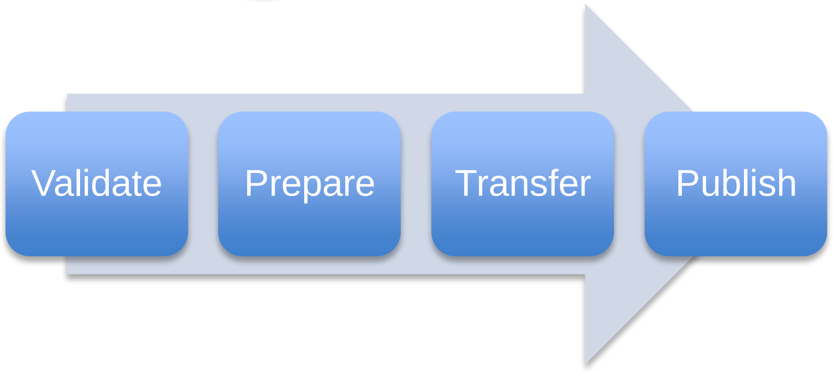
Rebuild All

Deploy Application

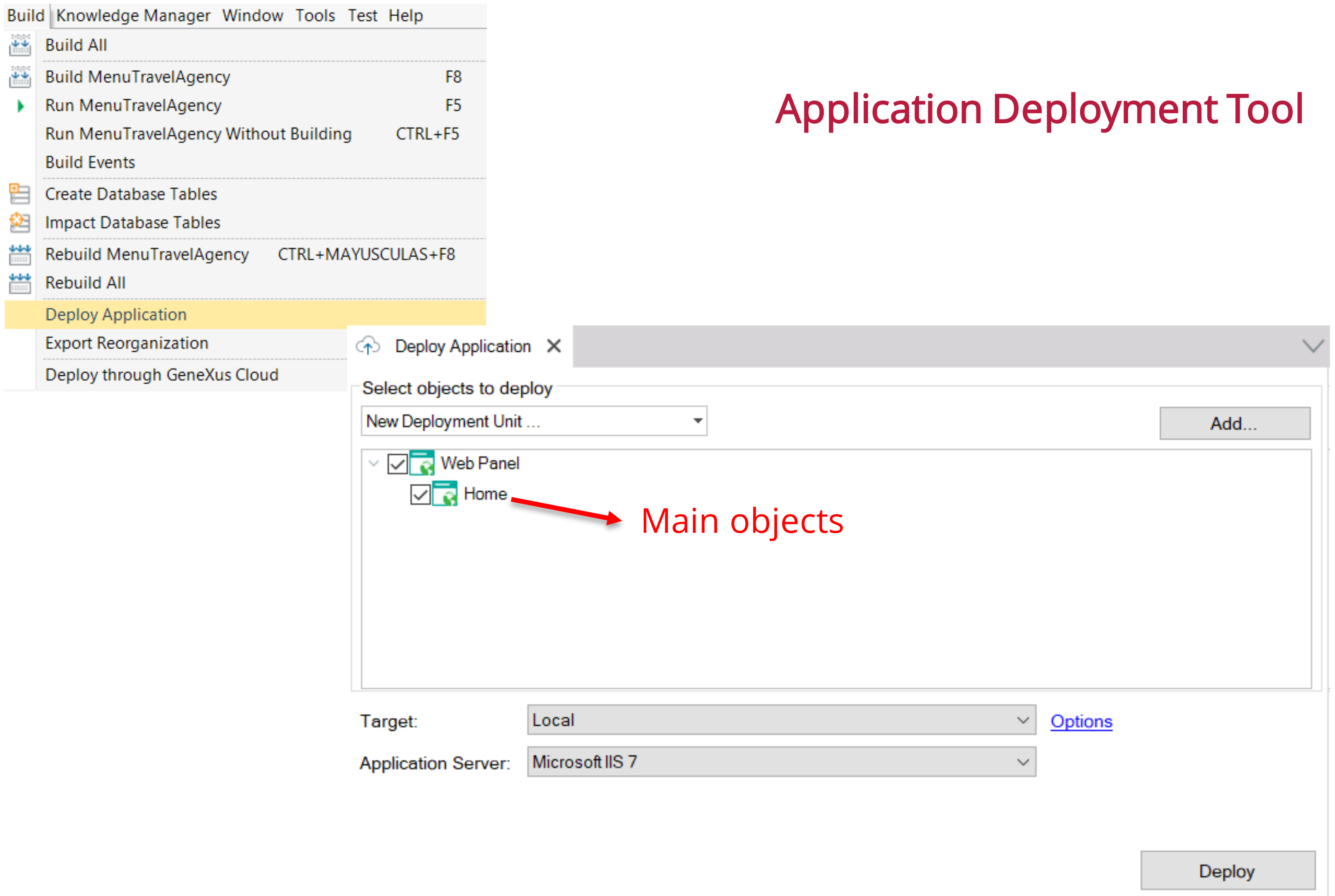
Export Reorganization

Deploy through GeneXus Cloud F6





Application Deployment Tool



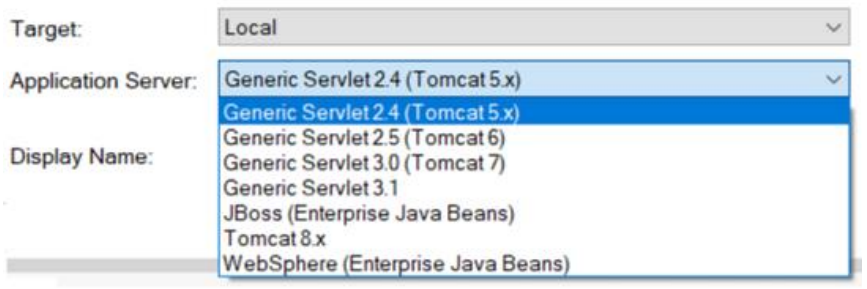
Target: Local

Application Server: Microsoft IIS 7

GeneXus™

Application Deployment Tool

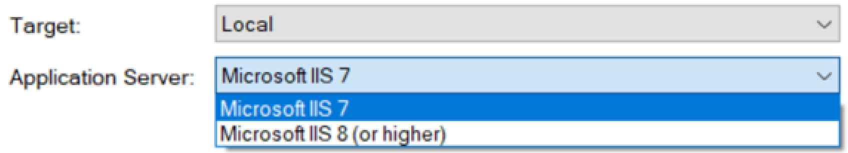
Local deployment (packages creation):


Target: Local

Application Server: Generic Servlet 2.4 (Tomcat 5.x)

- Generic Servlet 2.4 (Tomcat 5.x)
- Generic Servlet 2.5 (Tomcat 6)
- Generic Servlet 3.0 (Tomcat 7)
- Generic Servlet 3.1
- JBoss (Enterprise Java Beans)
- Tomcat 8.x
- WebSphere (Enterprise Java Beans)

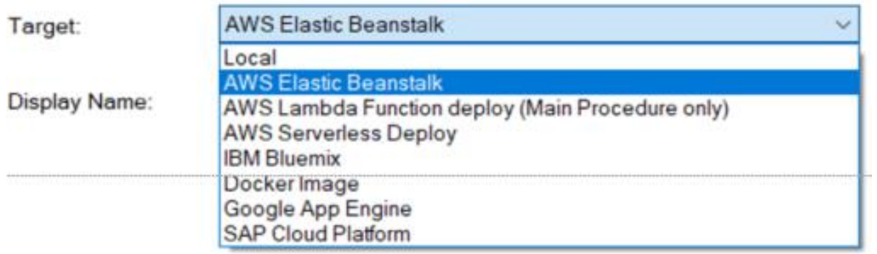
Display Name:


Target: Local

Application Server: Microsoft IIS 7

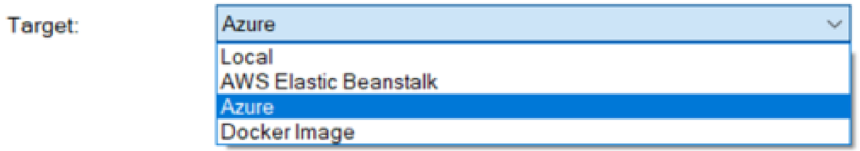
- Microsoft IIS 7
- Microsoft IIS 8 (or higher)

Deployment to PAAS (Platform As A Service) servers


Target: AWS Elastic Beanstalk

Display Name:

- Local
- AWS Elastic Beanstalk
- AWS Lambda Function deploy (Main Procedure only)
- AWS Serverless Deploy
- IBM Bluemix
- Docker Image
- Google App Engine
- SAP Cloud Platform


Target: Azure

- Local
- AWS Elastic Beanstalk
- Azure
- Docker Image

GeneXus™ Cloud

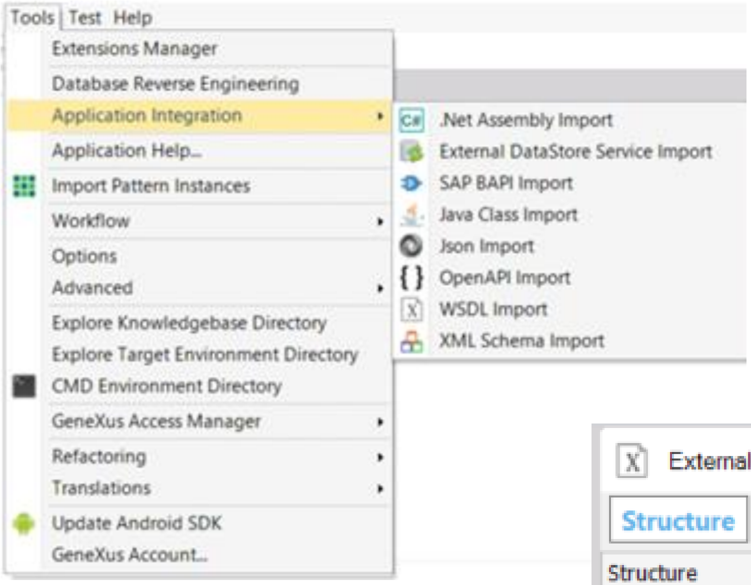
Deployment Services

- ✓ Building the environment in the Cloud
- ✓ Automatic deployment
- ✓ Deployment management
- ✓ Maximizing application uptime

Integration

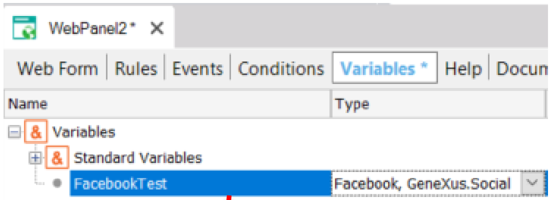
GeneXus™

External Objcs



Tools | Test | Help

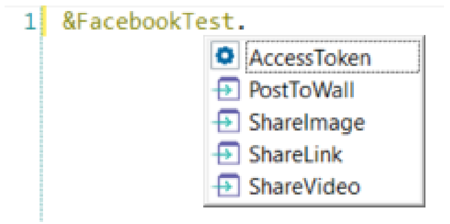
- Extensions Manager
- Database Reverse Engineering
- Application Integration ▸
  - .Net Assembly Import
  - External DataStore Service Import
  - SAP BAPI Import
  - Java Class Import
  - Json Import
  - OpenAPI Import
  - WSDL Import
  - XML Schema Import
- Application Help...
- Import Pattern Instances
- Workflow ▸
- Options
- Advanced ▸
- Explore Knowledgebase Directory
- Explore Target Environment Directory
- CMD Environment Directory
- GeneXus Access Manager ▸
- Refactoring ▸
- Translations ▸
- Update Android SDK
- GeneXus Account...



WebPanel2 * ✕

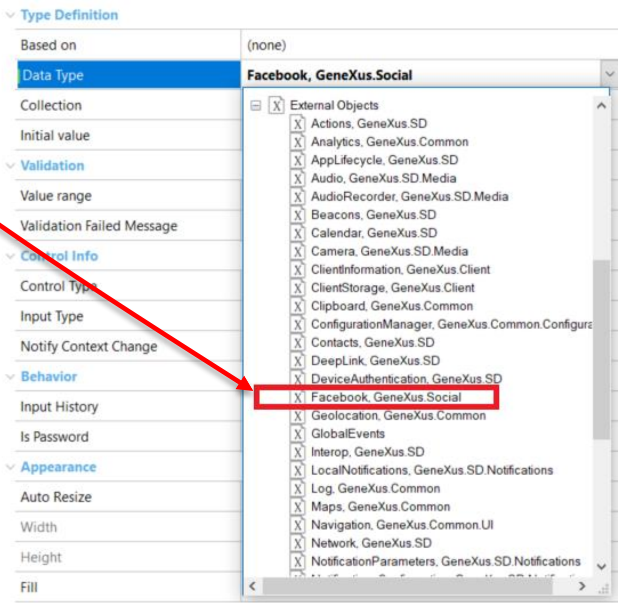
Web Form | Rules | Events | Conditions | Variables * | Help | Docum

Name	Type
Variables	
Standard Variables	
FacebookTest	Facebook, GeneXus.Social



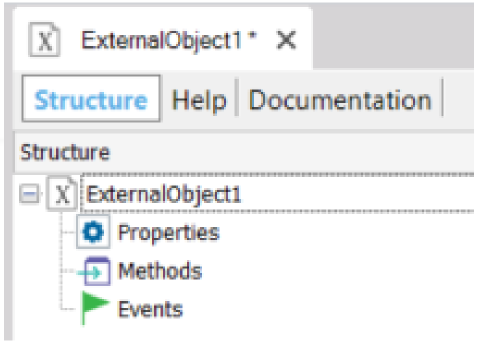
```
&FacebookTest.
```

- AccessToken
- PostToWall
- ShareImage
- ShareLink
- ShareVideo



Type Definition	
Based on	(none)
Data Type	Facebook, GeneXus.Social
Collection	
Initial value	
Validation	
Value range	
Validation Failed Message	
Control Info	
Control Type	
Input Type	
Notify Context Change	
Behavior	
Input History	
Is Password	
Appearance	
Auto Resize	
Width	
Height	
Fill	

External Objects
- Actions, GeneXus.SD
- Analytics, GeneXus.Common
- AppLifecycle, GeneXus.SD
- Audio, GeneXus.SD.Media
- AudioRecorder, GeneXus.SD.Media
- Beacons, GeneXus.SD
- Calendar, GeneXus.SD
- Camera, GeneXus.SD.Media
- ClientInformation, GeneXus.Client
- ClientStorage, GeneXus.Client
- Clipboard, GeneXus.Common
- ConfigurationManager, GeneXus.Common.Configura
- Contacts, GeneXus.SD
- DeepLink, GeneXus.SD
- DeviceAuthentication, GeneXus.SD
- Facebook, GeneXus.Social
- Geolocation, GeneXus.Common
- GlobalEvents
- Interop, GeneXus.SD
- LocalNotifications, GeneXus.SD.Notifications
- Log, GeneXus.Common
- Maps, GeneXus.Common
- Navigation, GeneXus.Common.UI
- Network, GeneXus.SD
- NotificationParameters, GeneXus.SD.Notifications

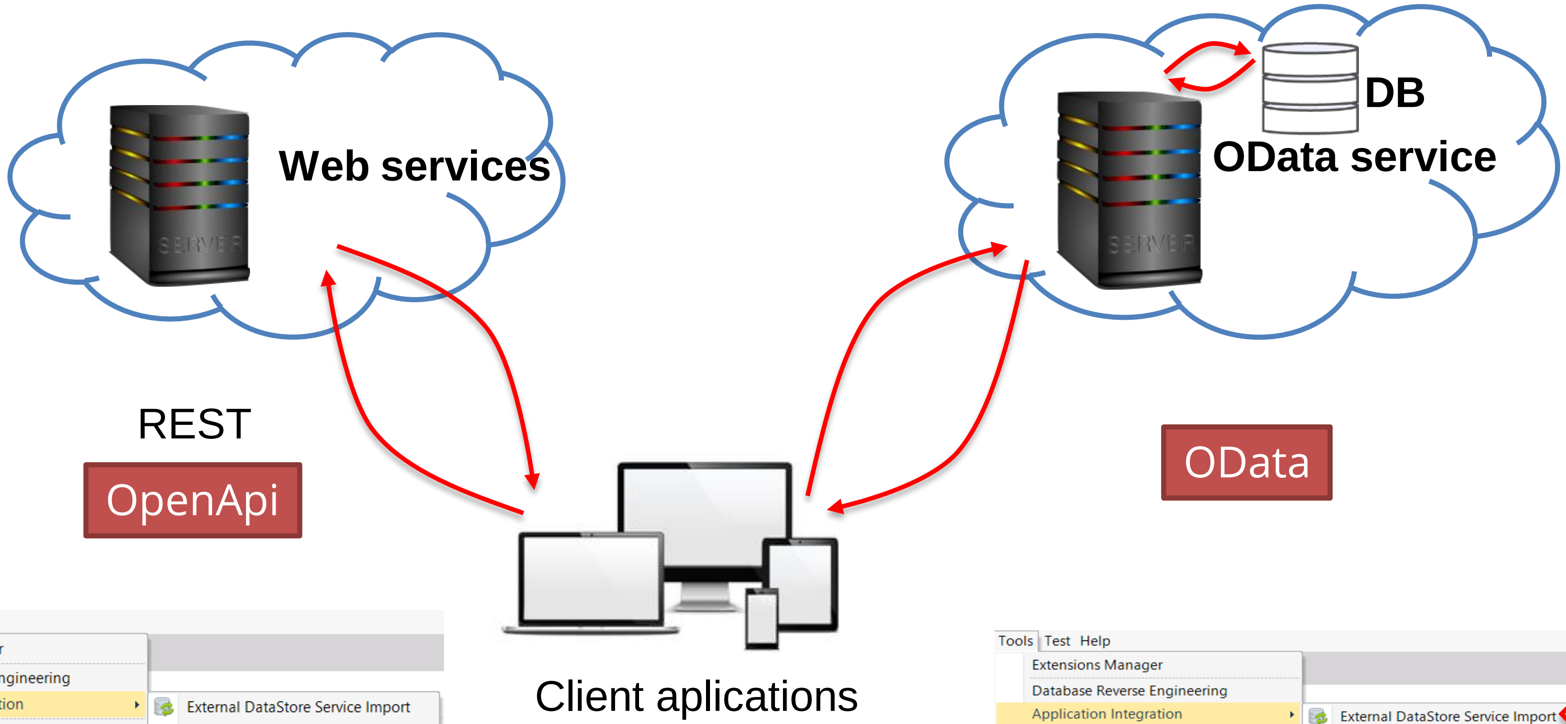


ExternalObject1 * ✕

Structure | Help | Documentation

Structure

- ExternalObject1
  - Properties
  - Methods
  - Events



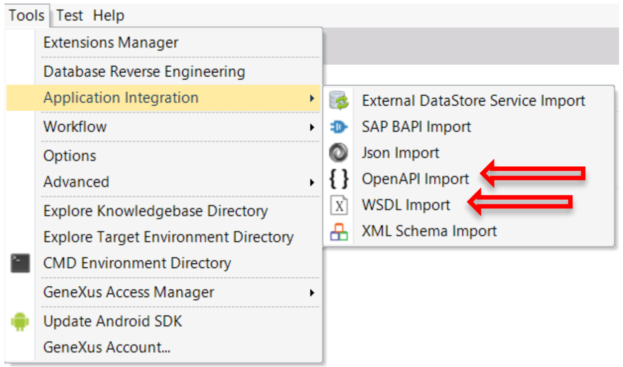
SOAP

WSDL

REST

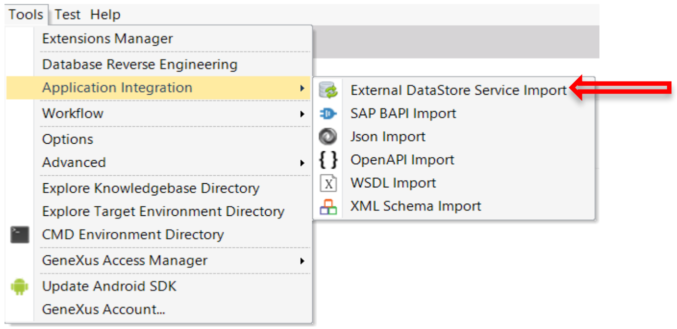
OpenApi

OData



Tools Test Help

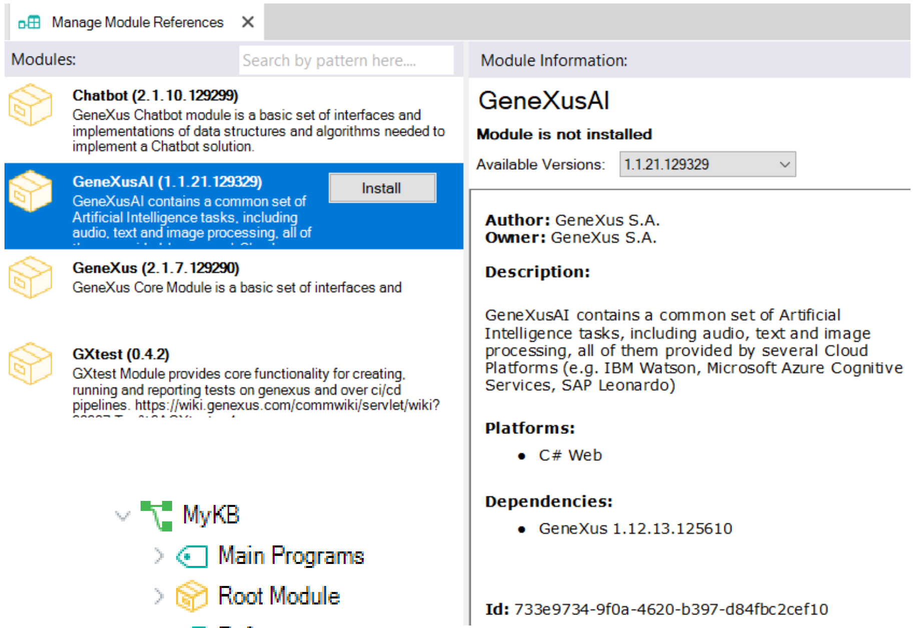
- Extensions Manager
- Database Reverse Engineering
- Application Integration ▸
  - External DataStore Service Import
  - SAP BAPI Import
  - Json Import
  - OpenAPI Import
  - WSDL Import
  - XML Schema Import
- Workflow ▸
- Options
- Advanced ▸
- Explore Knowledgebase Directory
- Explore Target Environment Directory
- CMD Environment Directory
- GeneXus Access Manager ▸
- Update Android SDK
- GeneXus Account...



Tools Test Help

- Extensions Manager
- Database Reverse Engineering
- Application Integration ▸
  - External DataStore Service Import
  - SAP BAPI Import
  - Json Import
  - OpenAPI Import
  - WSDL Import
  - XML Schema Import
- Workflow ▸
- Options
- Advanced ▸
- Explore Knowledgebase Directory
- Explore Target Environment Directory
- CMD Environment Directory
- GeneXus Access Manager ▸
- Update Android SDK
- GeneXus Account...

Artificial Intelligence Module


Manage Module References

Modules: Search by pattern here....

Chatbot (2.1.10.129299)
GeneXus Chatbot module is a basic set of interfaces and implementations of data structures and algorithms needed to implement a Chatbot solution.

GeneXusAI (1.1.21.129329) Install
GeneXusAI contains a common set of Artificial Intelligence tasks, including audio, text and image processing, all of

GeneXus (2.1.7.129290)
GeneXus Core Module is a basic set of interfaces and

GXtest (0.4.2)
GXtest Module provides core functionality for creating, running and reporting tests on genexus and over ci/cd pipelines. https://wiki.genexus.com/commwiki/servlet/wiki?

Module Information:

GeneXusAI
Module is not installed
Available Versions: 1.1.21.129329

Author: GeneXus S.A.
Owner: GeneXus S.A.

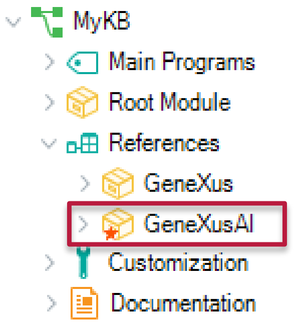
Description:

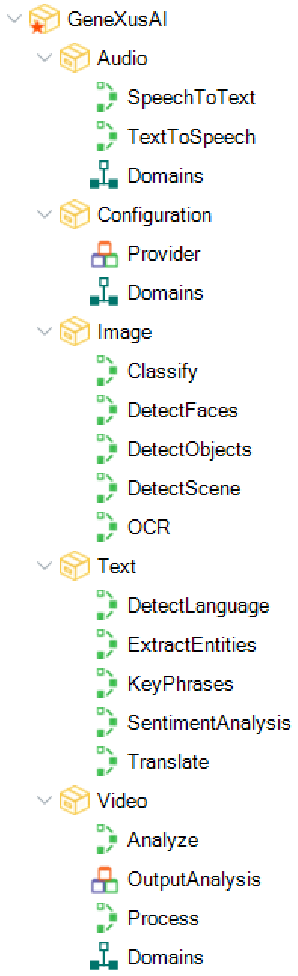
GeneXusAI contains a common set of Artificial Intelligence tasks, including audio, text and image processing, all of them provided by several Cloud Platforms (e.g. IBM Watson, Microsoft Azure Cognitive Services, SAP Leonardo)

Platforms:
- C# Web

Dependencies:
- GeneXus 1.12.13.125610

Id: 733e9734-9f0a-4620-b397-d84fbc2cef10


- MyKB
  - Main Programs
  - Root Module
  - References
    - GeneXus
    - GeneXusAI
  - Customization
  - Documentation


- GeneXusAI
  - Audio
    - SpeechToText
    - TextToSpeech
    - Domains
  - Configuration
    - Provider
    - Domains
  - Image
    - Classify
    - DetectFaces
    - DetectObjects
    - DetectScene
    - OCR
  - Text
    - DetectLanguage
    - ExtractEntities
    - KeyPhrases
    - SentimentAnalysis
    - Translate
  - Video
    - Analyze
    - OutputAnalysis
    - Process
    - Domains

IBM Watson

Microsoft Cognitive Services

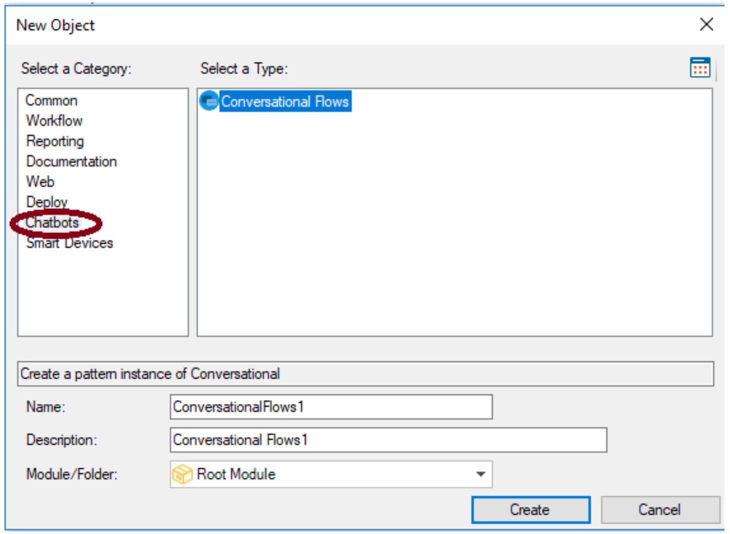
SAP Leonardo

Google Cloud Services

Amazon Web Services

GeneXus™

Chatbots: Conversational Flow object



New Object

Select a Category: Common, Workflow, Reporting, Documentation, Web, Deploy, Chatbots, Smart Devices

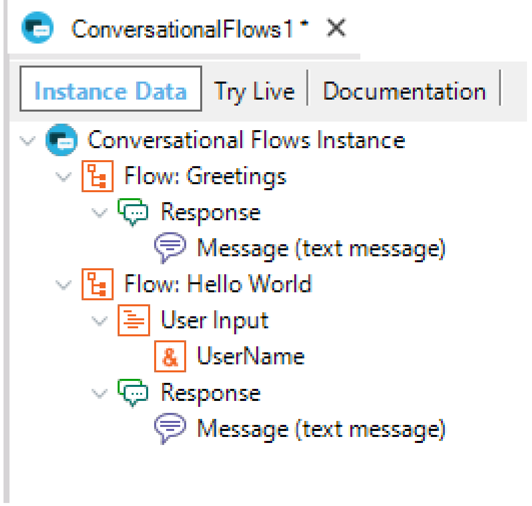
Select a Type: Conversational Flows

Create a pattern instance of Conversational

Name: ConversationalFlows1

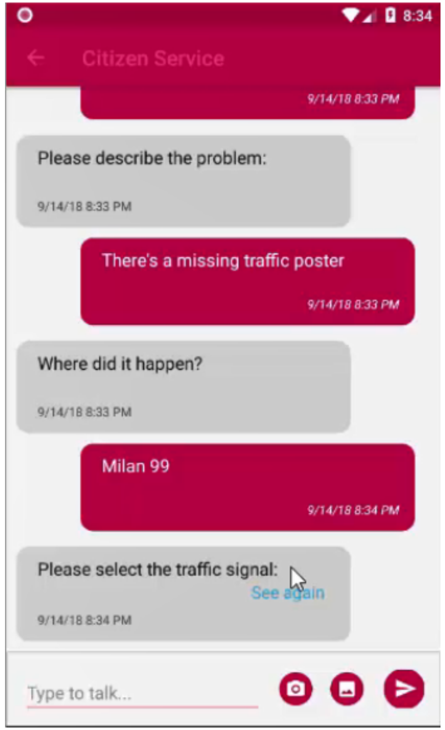
Description: Conversational Flows1

Module/Folder: Root Module



ConversationalFlows1 *

- Conversational Flows Instance
  - Flow: Greetings
    - Response
      - Message (text message)
  - Flow: Hello World
    - User Input
      - UserName
    - Response
      - Message (text message)



9/14/18 8:33 PM

Please describe the problem:
9/14/18 8:33 PM

There's a missing traffic poster
9/14/18 8:33 PM

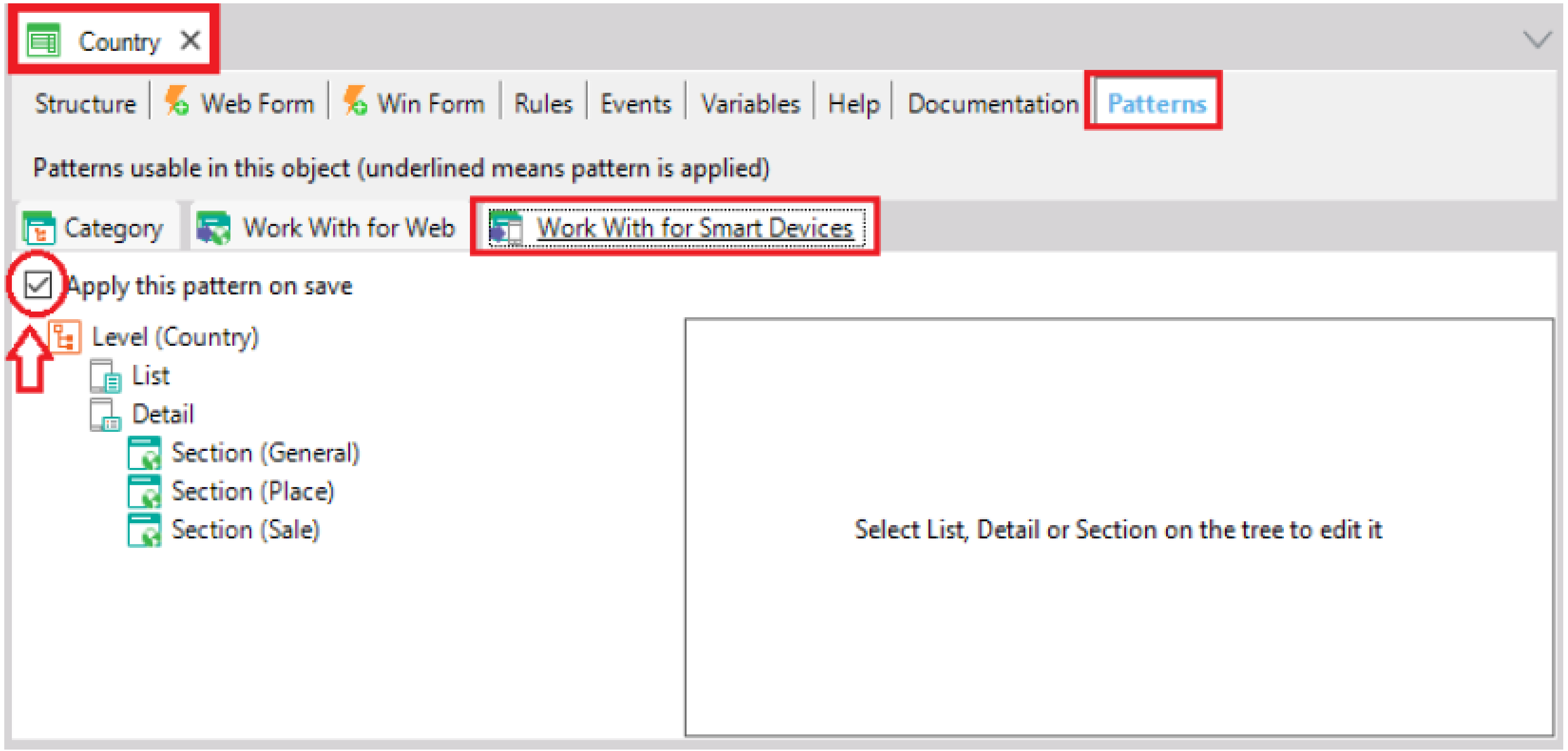
Where did it happen?
9/14/18 8:33 PM

Milan 99
9/14/18 8:34 PM

Please select the traffic signal:
See again
9/14/18 8:34 PM

Smart Devices

Applying a Pattern to a Transaction



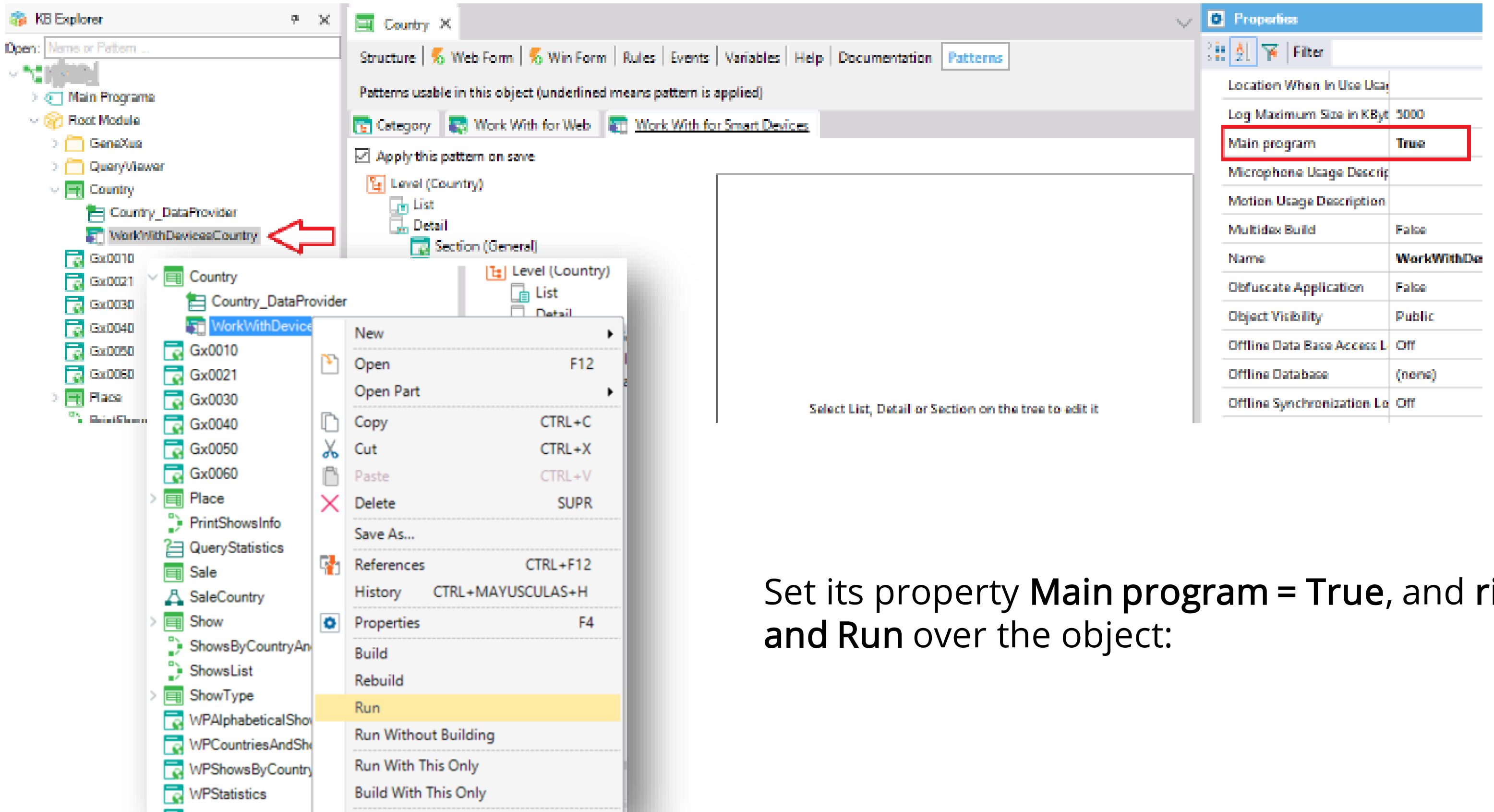
Patterns usable in this object (underlined means pattern is applied)

- [✓] Apply this pattern on save

```
Level (Country)
    List
    Detail
        Section (General)
        Section (Place)
        Section (Sale)
```

Select List, Detail or Section on the tree to edit it

Applying a Pattern to a Transaction



Patterns usable in this object (underlined means pattern is applied)

Apply this pattern on save

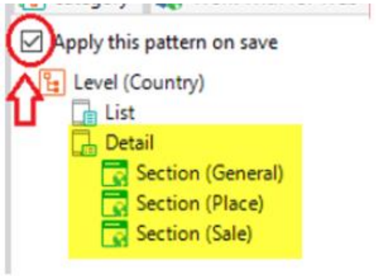
Select List, Detail or Section on the tree to edit it

Location When In Use Usa	
Log Maximum Size in KByt	5000
Main program	True
Microphone Usage Descrip	
Motion Usage Description	
Multidex Build	False
Name	WorkWithDe
Obfuscate Application	False
Object Visibility	Public
Offline Data Base Access L	Off
Offline Database	(none)
Offline Synchronization Lo	Off

Set its property **Main program = True**, and **right-click and Run** over the object:

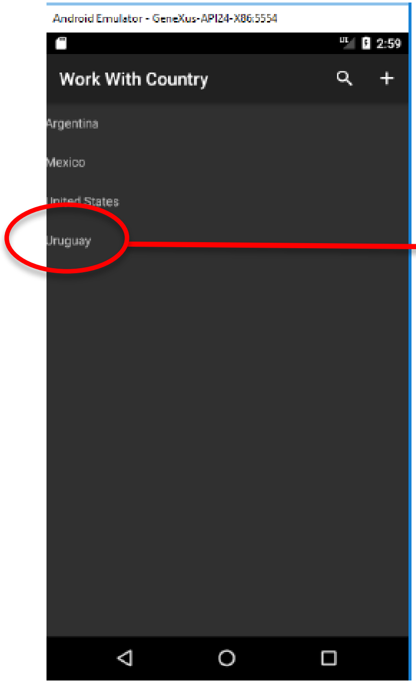
Applying a Pattern to a Transaction

Right-click and Run:



Apply this pattern on save

Level (Country)
- List
- Detail
  - Section (General)
  - Section (Place)
  - Section (Sale)



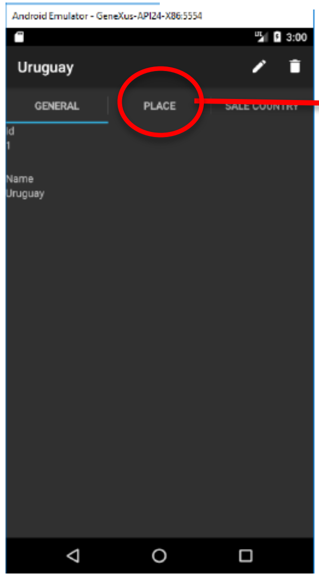
Android Emulator - GeneXus-API24-X86:5554

Work With Country

Argentina
Mexico
United States
Uruguay

GENERAL

Select 1 country



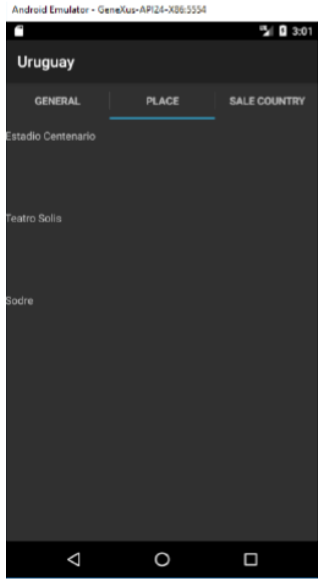
Android Emulator - GeneXus-API24-X86:5554

Uruguay

Id
1

Name
Uruguay

PLACE



Android Emulator - GeneXus-API24-X86:5554

Uruguay

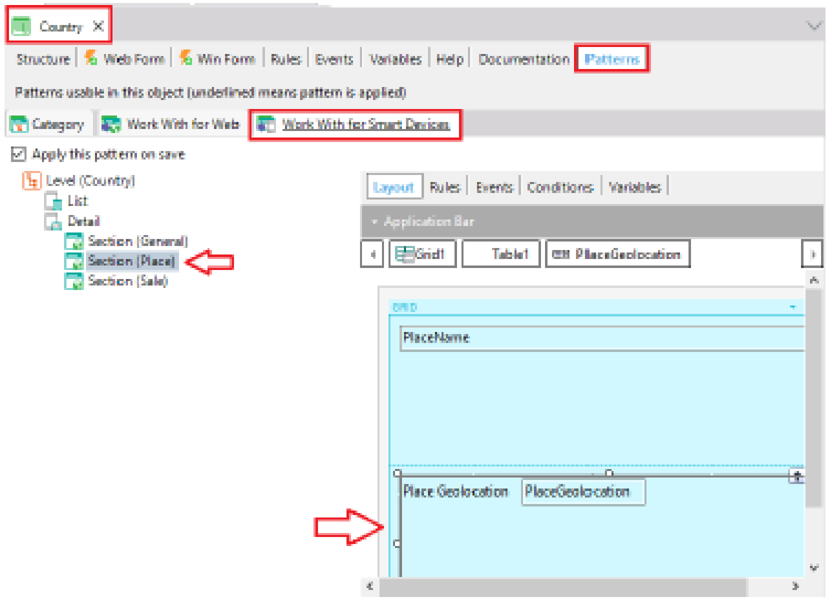
Estadio Centenario

Teatro Solis

Sodre

Applying a Pattern to a Transaction

For each place, in addition to its name, we want to see its geolocation.


Country
Structure | Web Form | Win Form | Rules | Events | Variables | Help | Documentation | Patterns
Category | Work With for Web | Work With for Smart Devices
Layout | Rules | Events | Conditions | Variables
Application Bar
Grid1 | Table1 | PlaceGeolocation
GRID

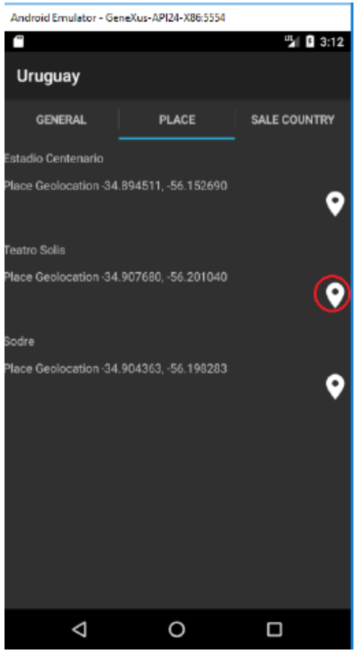
Patterns usable in this object (underlined means pattern is applied)

Apply this pattern on save

- Level (Country)
  - List
  - Detail
    - Section (General)
    - Section (Place)
    - Section (Sale)

PlaceName

Place Geolocation PlaceGeolocation


Android Emulator - GeneXus-API24-X86:5554

3:12

Uruguay

GENERAL | PLACE | SALE COUNTRY

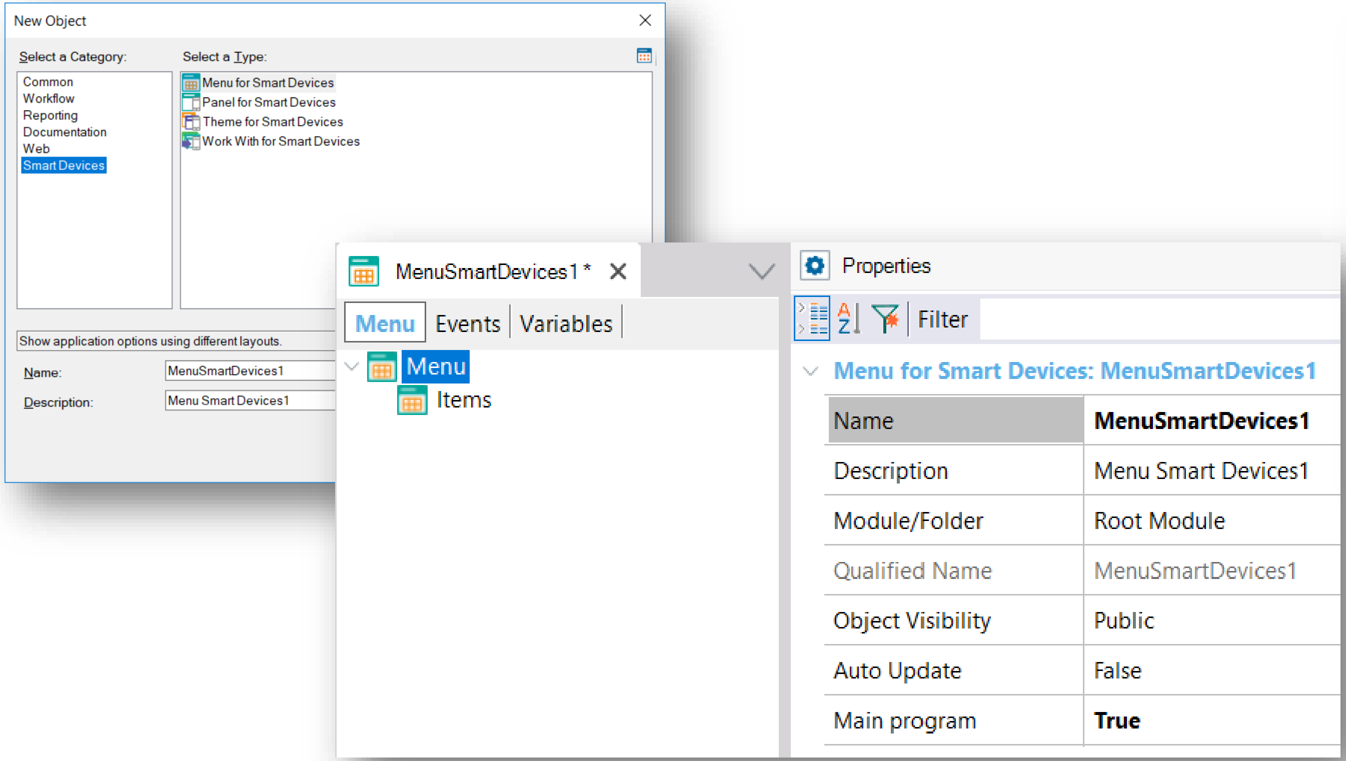
Estadio Centenario
Place Geolocation -34.894511, -56.152690

Teatro Solis
Place Geolocation -34.907680, -56.201040

Sodre
Place Geolocation -34.904363, -56.198283

Access Menu: Menu for Smart Devices

Creation:


New Object

Select a Category:

- Common
- Workflow
- Reporting
- Documentation
- Web
- Smart Devices

Select a Type:

- Menu for Smart Devices
- Panel for Smart Devices
- Theme for Smart Devices
- Work With for Smart Devices

Show application options using different layouts.

Name:	MenuSmartDevices1
Description:	Menu Smart Devices1

MenuSmartDevices1 *

Menu | Events | Variables

- Menu
  - Items

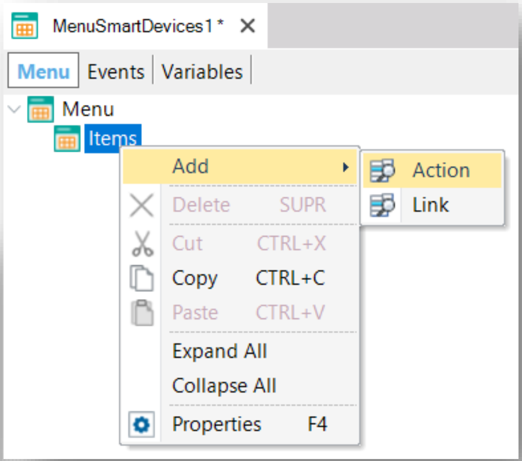
Properties

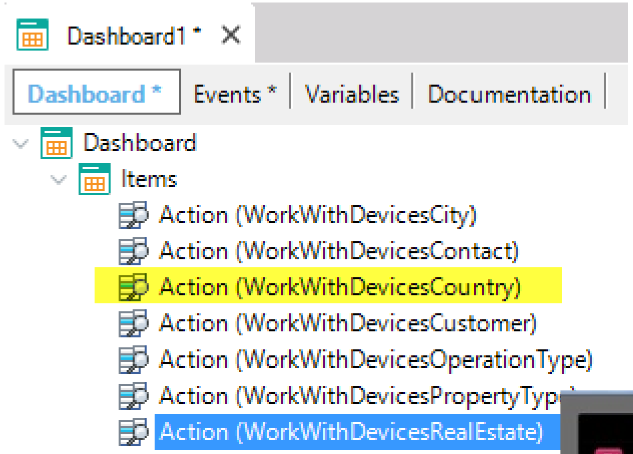
Filter

Menu for Smart Devices: MenuSmartDevices1

Name	**MenuSmartDevices1**
Description	Menu Smart Devices1
Module/Folder	Root Module
Qualified Name	MenuSmartDevices1
Object Visibility	Public
Auto Update	False
Main program	**True**

Add action…



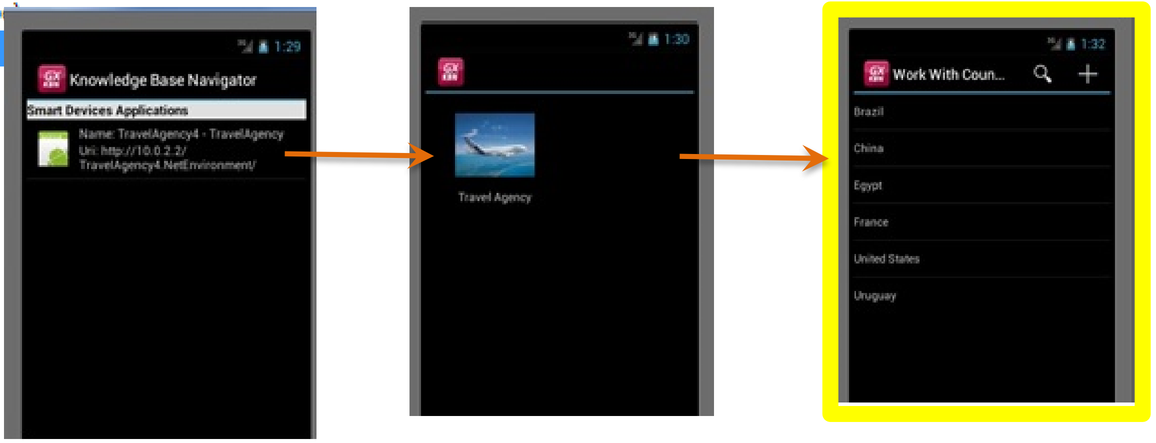


Dashboard1 *

Dashboard * | Events * | Variables | Documentation

- Dashboard
  - Items
    - Action (WorkWithDevicesCity)
    - Action (WorkWithDevicesContact)
    - Action (WorkWithDevicesCountry)
    - Action (WorkWithDevicesCustomer)
    - Action (WorkWithDevicesOperationType)
    - Action (WorkWithDevicesPropertyType)
    - Action (WorkWithDevicesRealEstate)

After pressing F5...



Knowledge Base Navigator

Smart Devices Applications

Name: TravelAgency4 - TravelAgency
Url: http://10.0.2.2/TravelAgency4.NetEnvironment/

Travel Agency

Work With Coun...

Brazil
China
Egypt
France
United States
Uruguay

GeneXus™

Videos	training.genexus.com
Documentation	wiki.genexus.com
Certifications	training.genexus.com/certifications