

Mocking de Base de Datos

Testing en GeneXus

Copyright ♥ Abstracta

All rights reserved. This document may not be reproduced by any means without the express permission of Abstracta. The information contained herein is intended for personal use only

Introducción

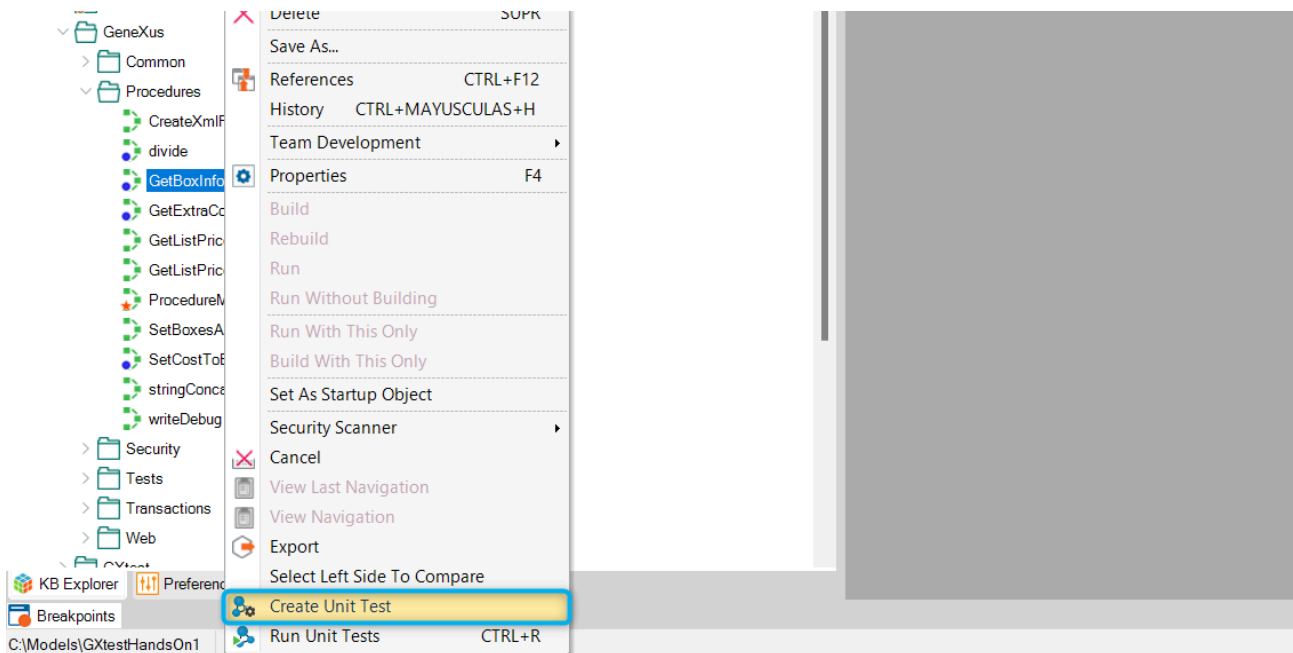
En esta capacitación práctica dispondrá de una guía para poner en práctica la funcionalidad de Mocking de Base de datos para realizar simulaciones de base de datos en la ejecución de las pruebas (Mocking).

Ejecución con mocking de Base de Datos

Supongamos que para nuestras pruebas necesitamos que el *Box 1* del *Container 1* siempre nos devuelva el precio seteado en el ejercicio anterior, independientemente de los cambios que puedan realizarse sobre la base de datos.

Para ello grabaremos el estado actual de dicho valor utilizando la funcionalidad **Database Mocking**.

- 1) Crear un Unit Test para el procedimiento “**GetBoxInfo**” de nuestra KB, haciendo clic derecho sobre el mismo y seleccionando la opción ‘*Create Unit Test*’.

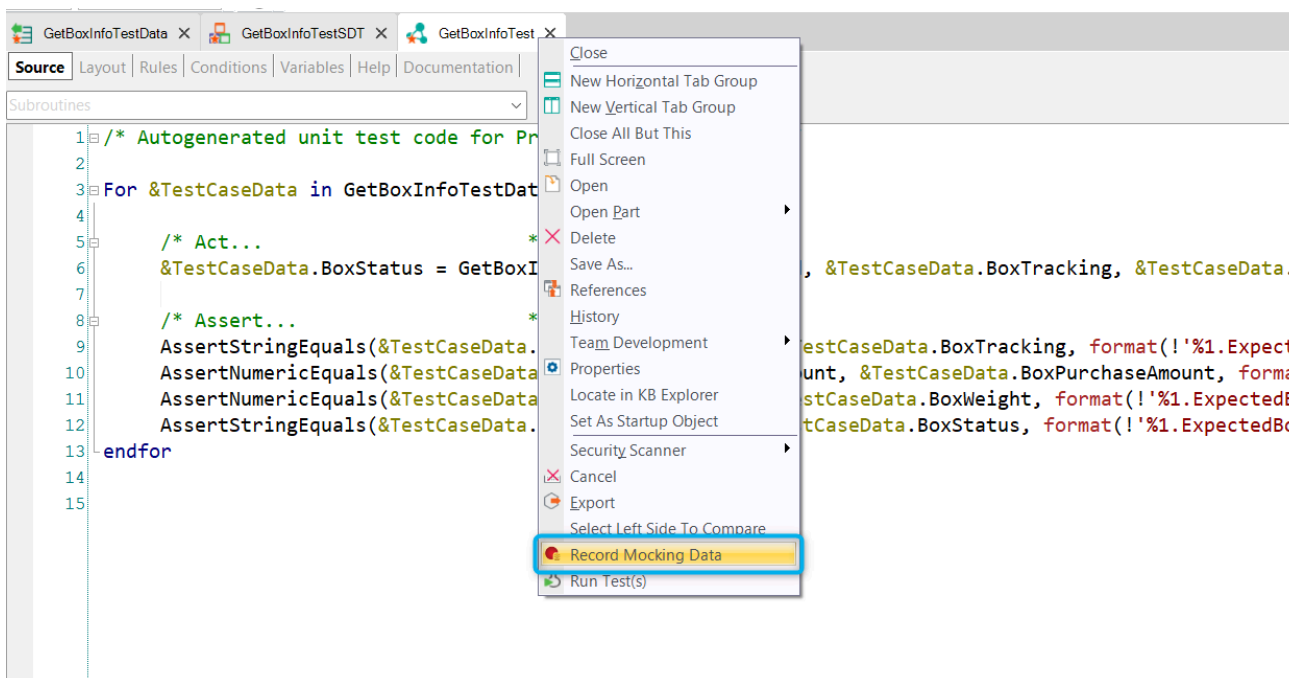


- 2) Abrir el objeto “GetBoxInfoTestData” y editar el valor **ExpectedBoxCost = 48.3**. Si queremos que todas las assertions pasen, también completamos el resto de los valores con los datos de **BoxId=1** que se pueden ver en la aplicación.

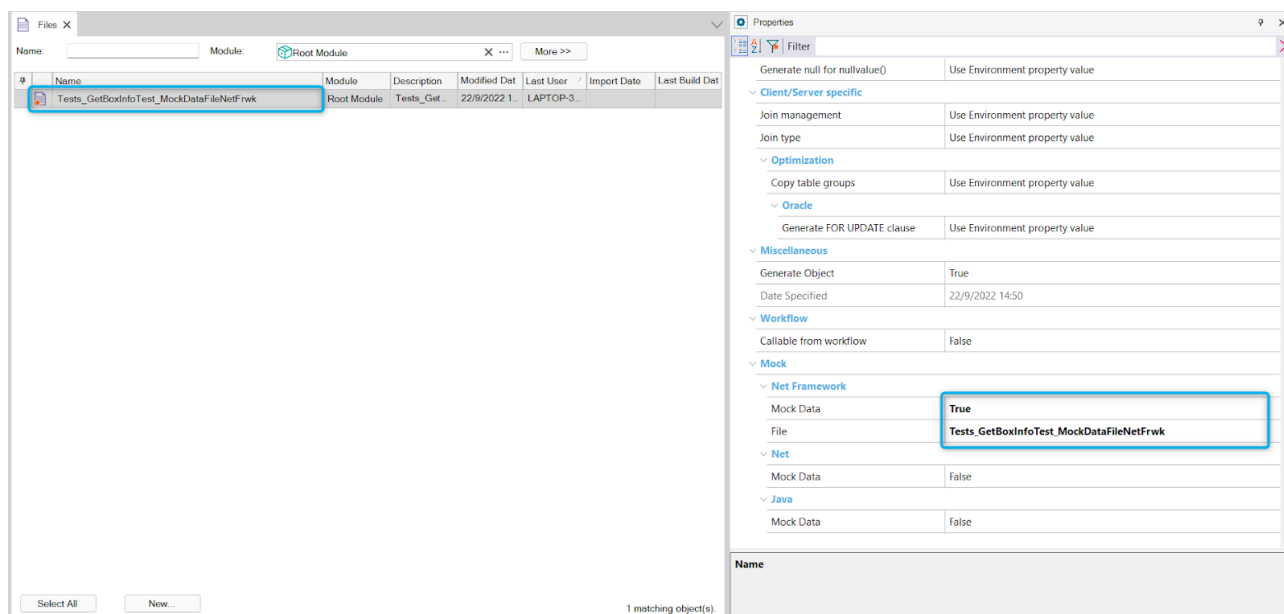
El código del Data Provider es:

```
GetBoxInfoTestSDT
{
    TestCaseId = '1'
    BoxId = 1
    ExpectedBoxTracking = '1146HFGDS3'
    MsgBoxTracking = '1146HFGDS3'
    ExpectedBoxPurchaseAmount = 180
    MsgBoxPurchaseAmount = '180'
    ExpectedBoxWeight = 2.3
    MsgBoxWeight = '2.3'
    ExpectedBoxStatus = Status.ATDESTINATION
    MsgBoxStatus = 'Status.ATDESTINATION'
    ExpectedBoxOriginArrivalDate = #2018-10-22#
    MsgBoxOriginArrivalDate = '#2018-10-22#'
    ExpectedBoxCost = 48.3
    MsgBoxCost = '48.3'
}
```

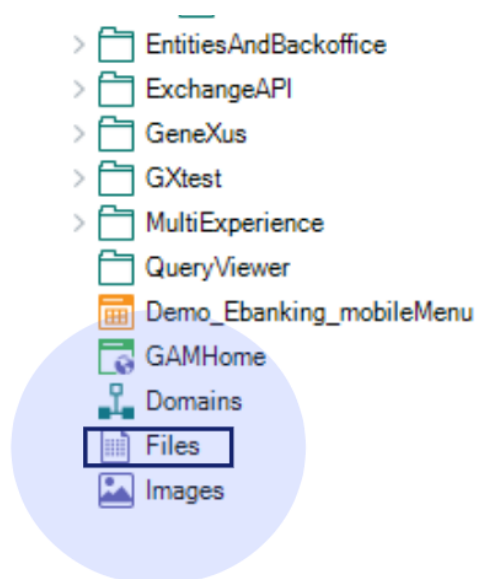
- 3) Guardar los cambios, hacer clic derecho sobre el unit test y seleccionar la opción “Record Mocking Data”.



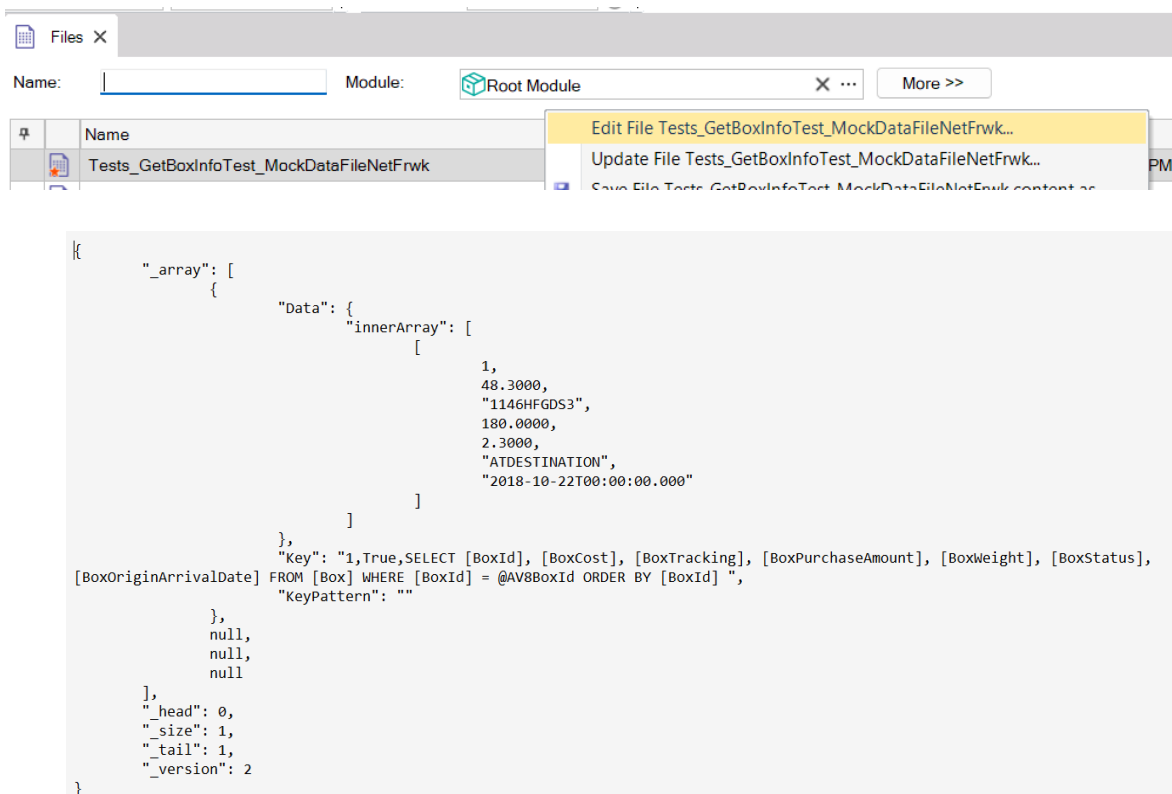
- 4) Vemos que se crea un archivo en *Files* con el nombre *'Tests_GetBoxInfoTest_MockDataFileNetFrwk'*. La propiedad **Mock Data** del test unitario cambia a 'True' y en File se asigna automáticamente el archivo creado.



- 5) Hacer doble click en KB Explorer -> Files para identificar en el listado y abrir el archivo grabado



6) Seleccione “Edit file.....” haciendo click derecho en el archivo grabado



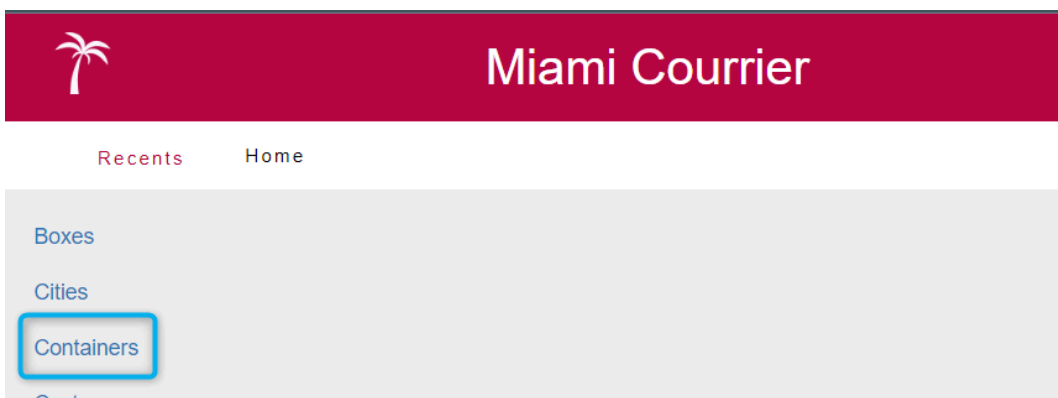
Observe que se graba en el campo Key la sentencia SQL SELECT ejecutada por el procedure y el resultado de la misma en el campo Data, el cual fue obtenido desde la base de datos de la aplicación.

Comentario:

El JSON generado para el ambiente Java es diferente al de la captura ya que el de la captura es el JSON generado para el ambiente .Net.

A continuación, editaremos los datos de la base de datos para verificar que el test está efectivamente ejecutando sobre el json y no sobre la base de datos:

7) Hacer clic en “Containers” y luego clic sobre el link “00128” del contenedor de Id 1 para ver la información del contenedor.



Miami Courier

Home — Containers

Containers

Internal Number

Id	Internal Number	Departure Date	Arrival Date	Airline	Total Boxes
1	00128	09/10/22	09/13/22	Tampa Cargo	9 XML
2	00129	09/15/22	//	DHL Cargo	5 XML

- 8) Hacer clic en la pestaña Box. Podrá observar en la columna Cost que todas las cajas tienen los valores seteados por el Proc *SetCostToBoxesInContainer*.

Container Information

← CONTAINERS

Internal Number
00128

General

Box

Tracking	Type	Weight	Status	Purchase Amount	Cost	Is Retained	Be Delivered	Customer
1146HFGDS3	Clothes	2.30	AT DESTINATION	180.00	48.30	<input type="checkbox"/>	false	Pablo Romero
KJS7587F541	Electronics	0.80	AT DESTINATION	30.00	18.40	<input type="checkbox"/>	false	Analia Gutierrez
290DKJHHXA3	Baby Accs.	1.70	AT DESTINATION	90.00	37.40	<input type="checkbox"/>	false	Joaquin Martinez
7343GFDW953	Home decor	3.60	AT DESTINATION	165.00	72.00	<input type="checkbox"/>	false	Sofia Perez
BCHS837MSM1	Clothes	1.80	AT DESTINATION	130.00	39.60	<input type="checkbox"/>	false	Pablo Romero
08KSDX73BD1	Furniture	5.00	AT DESTINATION	180.00	95.00	<input type="checkbox"/>	false	Analia Gutierrez
9BFIEN3832B	Cam Lenses	3.00	AT DESTINATION	180.00	60.00	<input type="checkbox"/>	false	Analia Gutierrez
SCYUETB98796	Stickers	1.00	AT DESTINATION	180.00	22.00	<input type="checkbox"/>	false	Sofia Perez
SCYUETB98796	Toys	2.00	AT DESTINATION	55.00	42.00	<input type="checkbox"/>	false	Joaquin Martinez

9) Seleccionar la Box 1 y hacer clic en Update

Box Information← BOXES

Tracking1146HFGDS3

General

UPDATE

DELETE

Id	1
Tracking	1146HFGDS3
Type	Clothes

10) En la celda Cost cambiar el valor 48.30 por 0.00 y hacer clic en Confirm

Cost0.00

Arrival Date10/22/18

Arrival Date09/13/22

Delivery Date//

Is Retained☐

Be Deliveredfalse

Customer Id1

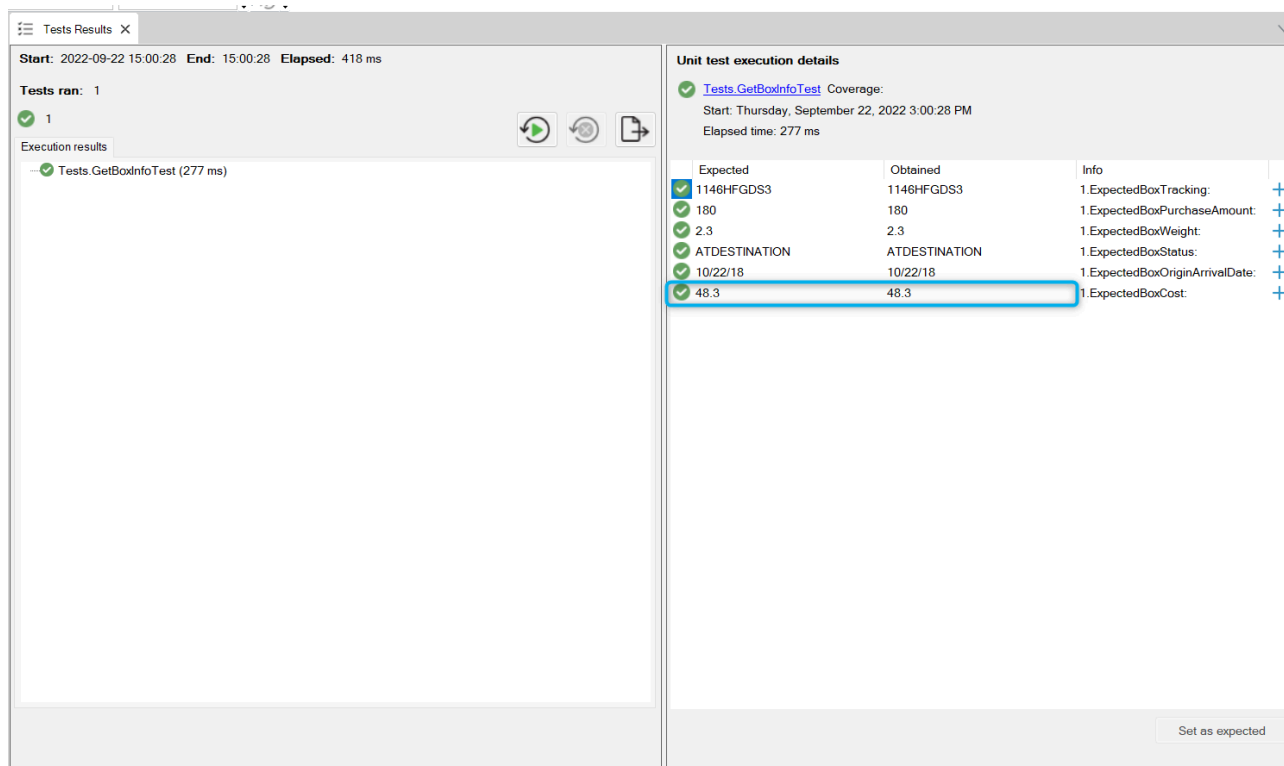
Container Id1

CONFIRM

CANCEL

- 11) Correr el test nuevamente haciendo clic en “Run Again” Esta vez se ejecutará con los datos grabados mediante mock.

Al terminar la ejecución del *test*, se mostrarán los resultados en el panel *Tests Results*. Vemos que el costo ‘48.3’, es obtenido desde el archivo grabado con mock, y no desde la base de datos editada anteriormente.



The screenshot displays the 'Tests Results' window in Visual Studio. The left pane shows the test 'Tests.GetBoxInfoTest' (277 ms) with a green checkmark indicating it passed. The right pane, titled 'Unit test execution details', shows the test's coverage and a table of expected vs. obtained values. The table has three columns: 'Expected', 'Obtained', and 'Info'. The 'Info' column lists various properties of the 'ExpectedBox' object. The row for '1.ExpectedBoxCost' shows both 'Expected' and 'Obtained' values as '48.3', which is highlighted with a blue box. Below the table is a 'Set as expected' button.

Expected	Obtained	Info
1146HFGDS3	1146HFGDS3	1.ExpectedBoxTracking: +
180	180	1.ExpectedBoxPurchaseAmount: +
2.3	2.3	1.ExpectedBoxWeight: +
ATDESTINATION	ATDESTINATION	1.ExpectedBoxStatus: +
10/22/18	10/22/18	1.ExpectedBoxOriginArrivalDate: +
48.3	48.3	1.ExpectedBoxCost: +

Comentarios adicionales al práctico:

La funcionalidad tiene como principal objetivo aislar los resultados de los tests del estado de la base de datos.

El archivo de mock es útil o sirve siempre y cuando las sentencias que genera GeneXus durante la ejecución del test sean las mismas que las que se grabaron en el archivo de mock.

En caso de que exista una diferencia entre las sentencias que genera GeneXus durante la ejecución y las sentencias grabadas en el archivo Mock, la ejecución se hará sobre la base de datos.



MONTEVIDEO - URUGUAY
SAN FRANCISCO - USA
LONDRES - UK

Sarmiento 2465, 11300
100 Pine St., Ste. 1250, CA 94111
71-75 Shelton Street, WC2H 9JQ

+598 2711 0561
+1 415 745 3678
+44 203 696 6682