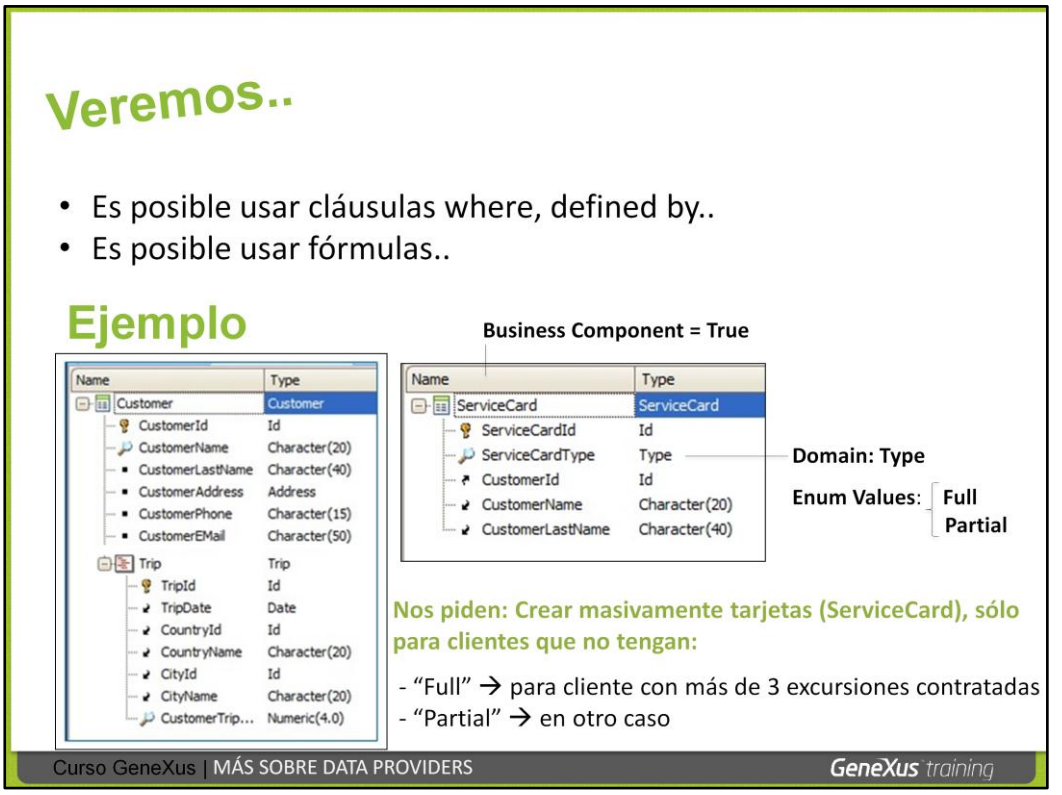
Introduciremos nuevos conocimientos sobre el uso de los Data Providers.

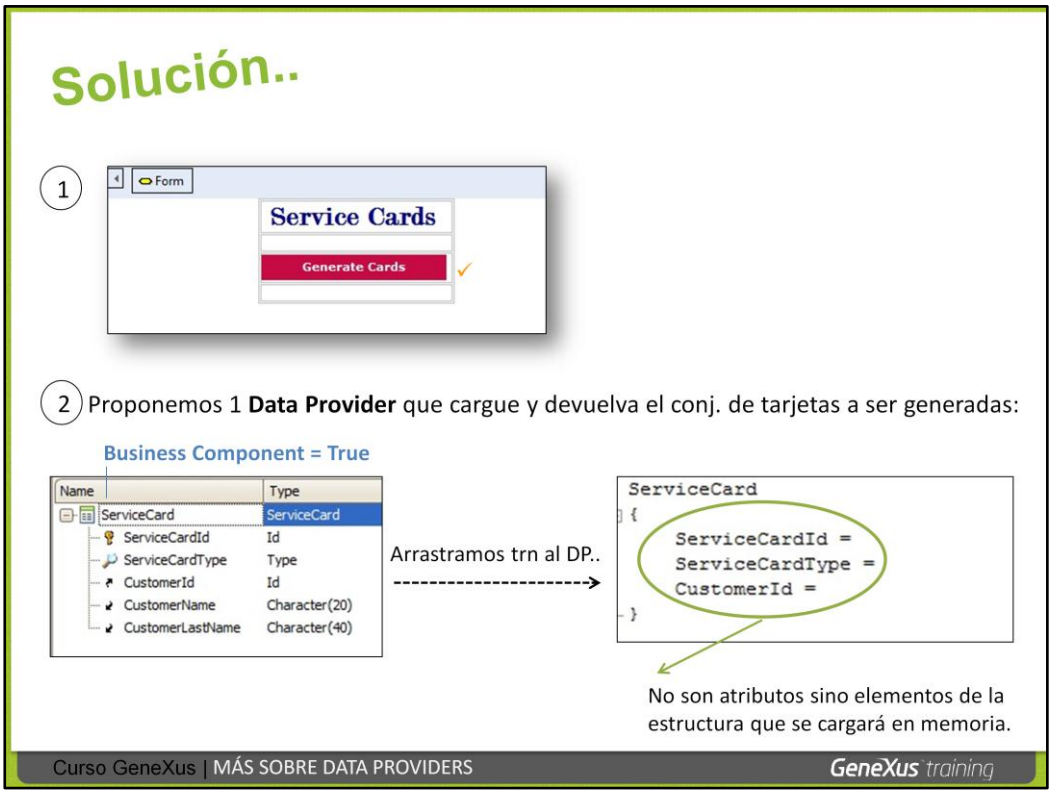


Aprenderemos nuevos conocimientos sobre Data Providers. Para ello propondremos una implementación práctica en nuestra aplicación.

Supongamos que la agencia de viajes decide que todos los clientes que han contratado más de 3 excursiones, recibirán una tarjeta especial de tipo "Full Services", que les permitirá disfrutar de todos los servicios en forma gratuita. Y en caso de haber contratado menos de 3 excursiones, recibirán la tarjeta de tipo "Partial Services".

Observemos las transacciones con las cuales contamos. La transacción "Customer" y la transacción "ServiceCard" que define a las tarjetas.

Cada tarjeta tiene un identificador que se autonumera, un cliente y hemos definido el atributo ServiceCardType basado en el dominio Type, enumerado (que admite solamente los valores "Full" o "Partial").



Hemos definido el web panel WPCards, que simplemente ofrece un botón que disparará el proceso automático de generación y visualización de nuevas tarjetas.

¿Qué deberá suceder al presionar el botón?

Para todos aquellos clientes que aún no tengan tarjeta emitida, se les deberá crear una tarjeta del tipo que corresponda, teniendo en cuenta la cantidad de excursiones que han contratado a la agencia.

Propondremos una solución **usando un Data Provider** que cargue y devuelva la colección de tarjetas a ser generadas. **Y luego recorreremos la colección y grabaremos las tarjetas en la base de datos.**

¿Cómo hacemos esto?

En primer lugar, configuramos la transacción ServiceCard como Business Component, para grabar las tarjetas haciendo uso del concepto de Business Component.

Después creamos un objeto Data Provider de nombre DPCards y arrastramos la transacción ServiceCard hacia el source del DP.

En este caso particular, no hemos definido un SDT en la base de conocimiento, pero una transacción que es Business Component (ServiceCard) puede arrastrarse a un DP y la sintaxis nos permite deducir que se cargará una estructura en memoria con ítems de igual nombre y tipo que los atributos de la transacción.

Ahora necesitamos recorrer la tabla CUSTOMER, filtrar los clientes que aún no tienen una tarjeta emitida **y para ellos** agregar una tarjeta en la colección.
La tabla base de un DP, es decir la tabla que navegará, se determina por los atributos que referenciemos del lado derecho de los signos de

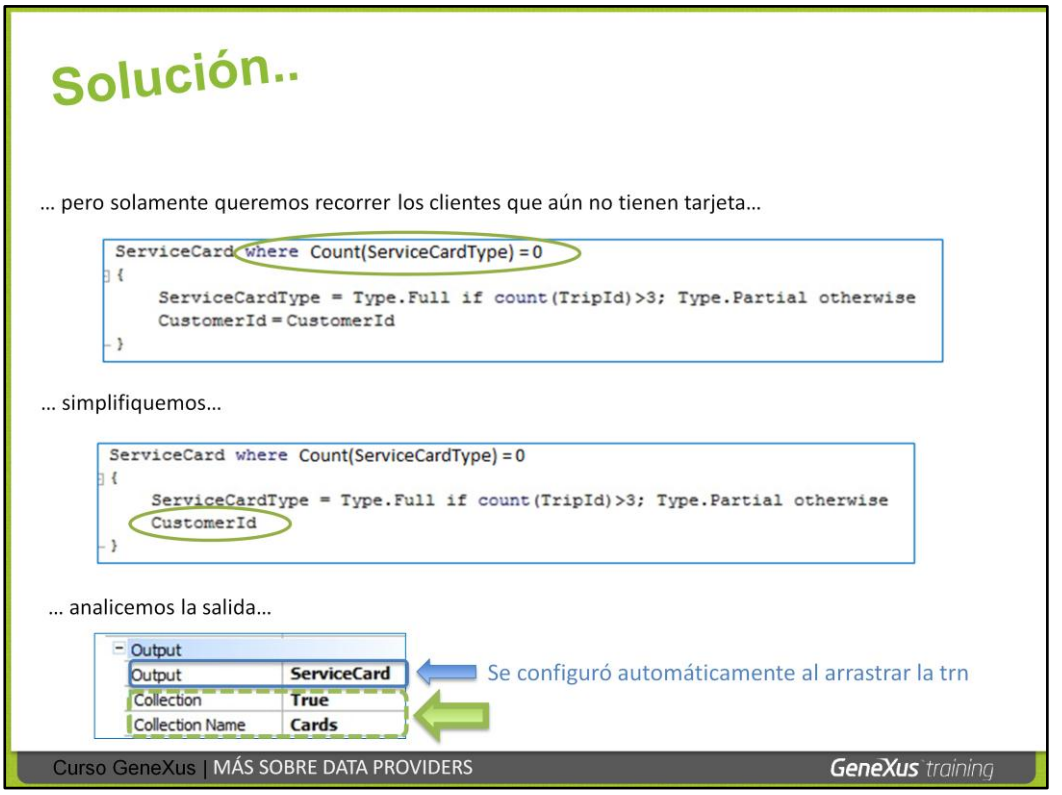
asignación.

Al ítem CustomerId, es claro que le vamos a asignar el atributo CustomerId.

Al ítem ServiceCardId no le necesitamos asignar un valor específico, porque recordemos que esta estructura a ser cargada fue arrastrada de una transacción declarada como BC.. y por lo tanto este ítem está basado en el atributo ServiceCardId, que pertenece al dominio Id y su propiedad Autonumber está configurada en Yes.

Veamos que a ServiceCardType le podemos asignar el valor que retorne una fórmula condicional, es decir que la fórmula devolverá el tipo "Full" o "Partial" de acuerdo a la cantidad de excursiones que ha contratado el cliente.

Observemos cuáles atributos figuran del lado derecho de los signos de asignación. Solamente CustomerId, porque los atributos que estan dentro de fórmulas, no se tienen en cuenta para la determinación de esta tabla base sino que determinan la tabla a ser navegada por la fórmula. Así que la tabla navegada por el DP es CUSTOMER.



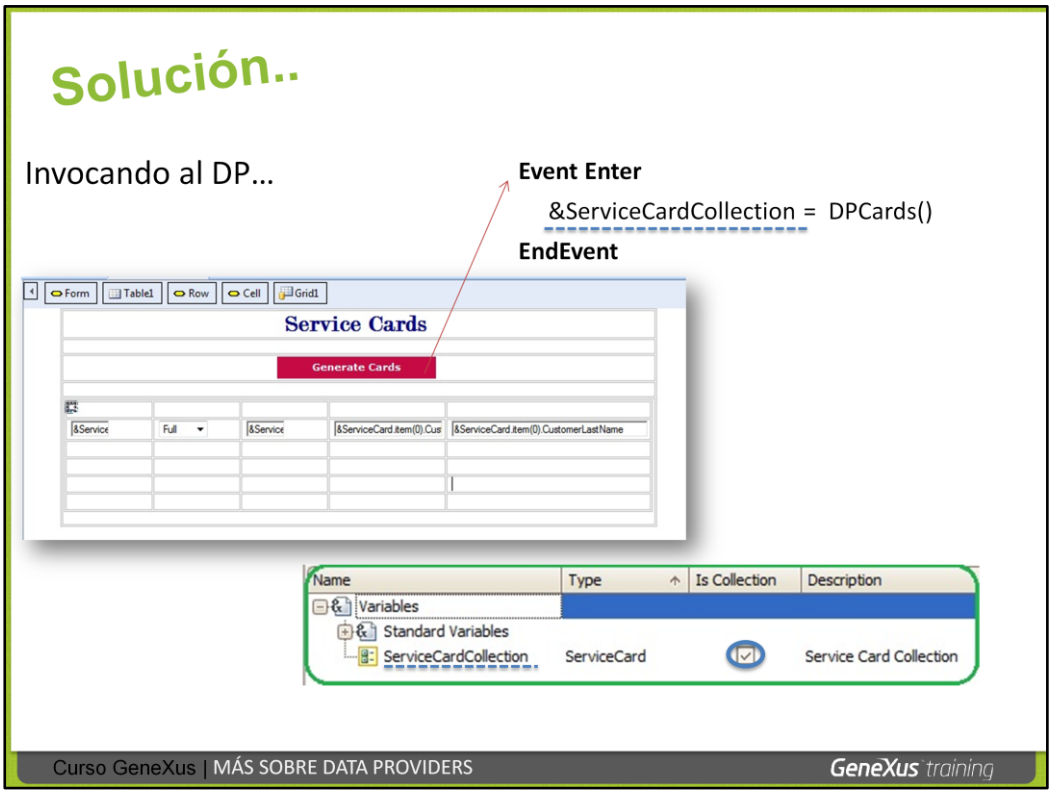
Pero no queremos navegar todos los clientes y para cada uno cargar una tarjeta, sino que queremos navegar solamente aquellos clientes que aún no tienen tarjeta.

Los DP permiten en su sintaxis, que les incluyamos todas las cláusulas permitidas en For each, así que agregamos la cláusula Where mostrada en la diapositiva.

Dado que el único atributo referenciado en el Where, está dentro de 1 fórmula, como ya hemos dicho, no participa en la determinación de la tabla base del DP. De haberse evaluado directamente, hubiera participado.

Ahora analicemos la salida, el output, del DP: Al haber arrastrado la transaccion ServiceCard, automáticamente, la propiedad Output quedó asociada a dicha transacción.

Y la propiedad Collection? La estructura que estamos cargando no representa una colección. Solamente representa una instancia en memoria, con la estructura de un registro de la tabla asociada a la transaccion ServiceCard. Sin embargo necesitamos obtener una **colección de tarjetas generadas**, entonces configuramos la propiedad Collection con valor True… e indicamos un nombre para la colección que devolverá cargada este DP.



Volvamos ahora al web panel para invocar al DP.

En el evento Enter asociado al botón, le asignamos a una variable (&ServiceCardCollection) definida como una colección de tarjetas (ServiceCard), lo que devolverá el DP.

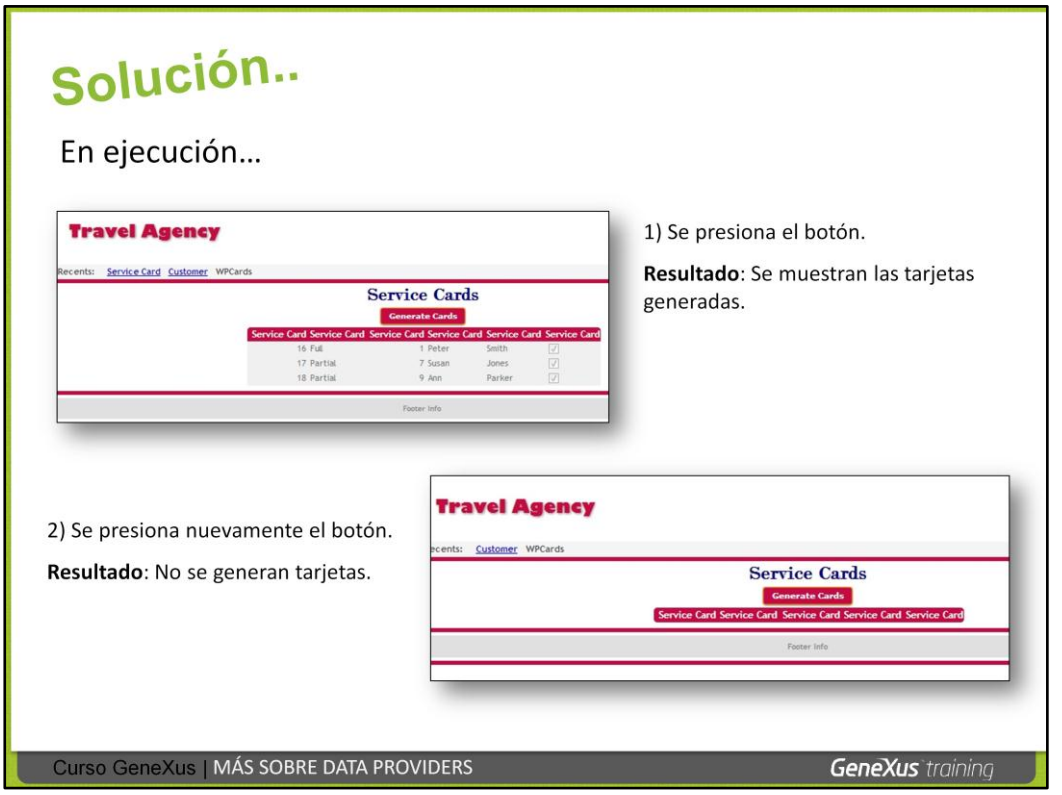
En el form del web panel, insertamos la variable &ServiceCardCollection. Por tratarse de una colección, automáticamente GeneXus entiende que debe mostrar el contenido en un grid.

Ahora, ¿esto alcanza para que las tarjetas devueltas por el DP efectivamente se graben en la tabla SERVICECARD asociada a la transaccion ServiceCard?

No. Por ahora las tarjetas están cargadas en memoria y hemos mostrado el contenido de la colección.

Cuando estudiamos la utilización de transacciones como BC, vimos que para grabar debemos usar el método Save y luego ejecutar Commit. Así que nos está faltando recorrer la colección devuelta por el DP y proceder a grabar cada elemento de la colección como registro en la tabla física. Y posteriormente a la grabación de todas las tarjetas, declaramos el comando Commit.

Para recorrer la colección de tarjetas devuelta por el DP, empleamos el comando **For elemento in Colección.** Esta variable &oneCard debe ser definida. Representa a cada elemento que se va iterando de la colección.



Ahora sí el desarrollo de lo que nos solicitaron está completo. Ejecutamos el web panel y presionamos el botón. Vemos en la grilla la lista de tarjetas que se generaron.

Puede surgirnos la pregunta: ¿Qué sucederá si volvemos a presionar el botón "Generate Cards"?
¿Se volverán a crear las tarjetas para los mismos clientes?

No, porque en el DP filtramos que solamente queríamos navegar los clientes sin tarjetas.

Vale mencionar que hay otras soluciones para resolver el mismo requisito en GeneXus. **Con esta implementación hemos usado el concepto de Business Component para actualizar la base de datos y hemos combinado su uso con el hecho de cargar previamente una estructura colección en memoria con los datos a grabar.**

El uso de un Data Provider para esto, es muy sencillo y nos ahorra escribir código explícito.