

Artículo técnico: AX2012 – Herencia de tablas (I)

Revisión: 0.1

Fecha: 11/08/2011

Autor: Manel Querol (mquerol@gmail.com)



<http://www.trucosax.com>



Introducción

En breve se va a producir el lanzamiento de Dynamics AX 2012, la última versión de nuestro querido ERP de Microsoft.

Desde hace un tiempo, podemos disfrutar de la beta pública y hemos podido ya comprobar el salto tecnológico y funcional que se produce respecto a la versión anterior: ¡Es sencillamente espectacular!

En este artículo me gustaría mostrar una de las nuevas características técnicas que incorpora esta nueva versión: La herencia de tablas.

En las versiones anteriores de Dynamics AX, las tablas ya eran tratadas como objetos:

- Todas las tablas heredaban de la clase “Common”
- Todas tenían sus propios métodos tanto de instancia como estáticos.
- Tenían propiedades que se heredaban de la clase “Common”

Pero aunque eran, en esencia, como una clase, no estaba permitido usar la herencia para crear descendientes de las tablas (lo que se conoce como una clase final ya que no permite crear descendientes).

Para mí, se trataba de una limitación importante ya que, si estas trabajando en un entorno orientado a objetos, se supone que todo son objetos y se debería tratar como tal. De hecho, ya en el año 2000 con un equipo de desarrolladores realizamos un proyecto que consistía en una herramienta para crear aplicaciones y era totalmente orientada a objetos: Las tablas, los formularios, la propia aplicación... todo eran objetos y por tanto, todo era heredable.

Bien, pues Dynamics AX 2012 incorpora el concepto de herencia de tablas con todo lo que ello conlleva.

Mi intención será mostrar en que consiste la herencia de tablas, como se usa y como funciona por dentro en la medida de lo posible.

¿En que consiste la herencia de tablas?

Igual que como si de cualquier otra clase se tratara, AX2012 nos permite realizar herencia de tablas. Es decir, crear tablas derivadas que heredan campos, métodos y propiedades de una tabla base y extienden dicha tabla base con sus propios campos y métodos adicionales que las hacen especiales.

¿Cuándo usar herencia de tablas?

Podemos plantearnos usar la herencia entre dos tablas, entendiendo una de ellas como tabla base y la otra como tabla derivada, cuando:

- Un registro de la tabla base y el registro correspondiente en la tabla derivada hacen referencia al mismo elemento del mundo real y las dos tablas hacen referencia a diferentes atributos del mismo elemento.



- Cada registro existente en la tabla base se corresponde con un registro en la tabla derivada, y si se elimina uno de los dos registros de una de las tablas, deberá eliminarse el registro correspondiente en la otra tabla.
- Existirán al menos 2 tablas derivadas de la tabla base, cada una de ellas con sus diferentes atributos. Las dos tablas derivadas representan diferentes variaciones de los elementos generales que se encuentran en la tabla base.
- Un elemento de la tabla base no puede estar a la vez representado en más de una de las tablas derivadas.
- No existirá la posibilidad de establecer una relación de 1 a N o de N a N entre los registros de la tabla base y la tabla derivada.

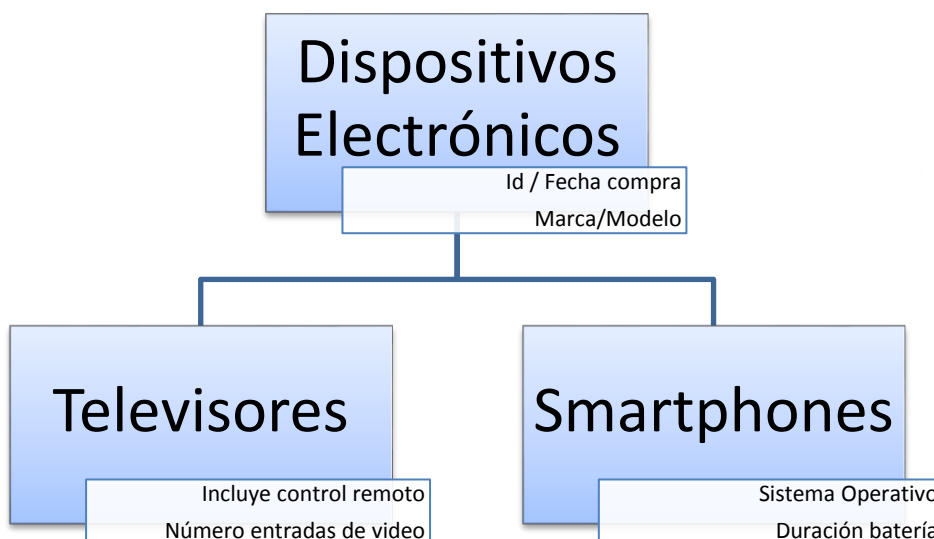
Con un ejemplo lo veremos más claro.

Un ejemplo

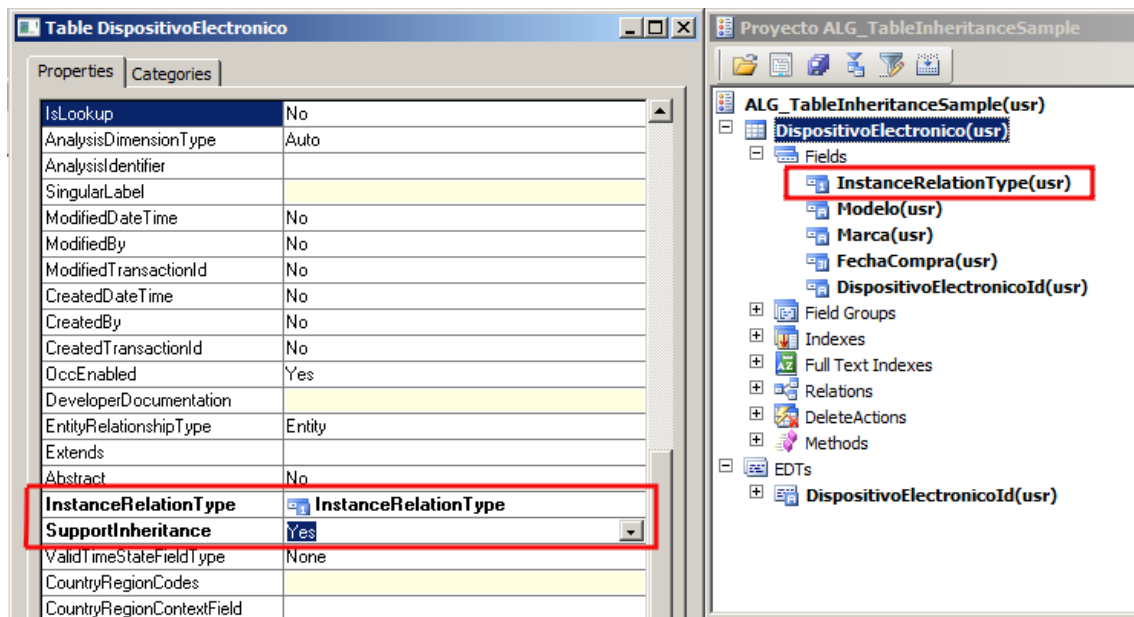
Imaginemos que disponemos de toda una colección de dispositivos electrónicos de segunda mano entre los que se encuentran “Smartphones” y “Televisores” y realizamos una lista con sus propiedades más relevantes.

Televisores	Smartphones
<ul style="list-style-type: none"> • Id • Fecha de compra • Marca • Modelo • Incluye control remoto (S/N) • Numero de entradas de video 	<ul style="list-style-type: none"> • Id • Fecha de compra • Marca • Modelo • Sistema Operativo • Duración batería

A simple vista nos damos cuenta de que ambos dispositivos comparten algunas propiedades y, por tanto, podemos establecer una tabla base “DispositivoElectronico” con las propiedades comunes y luego crear dos tablas derivadas, una para televisores y la otra para los smartphones.



¡Manos a la obra!, abrimos un entorno de desarrollo (Ctrl+Shift+W) y creamos la tabla base...

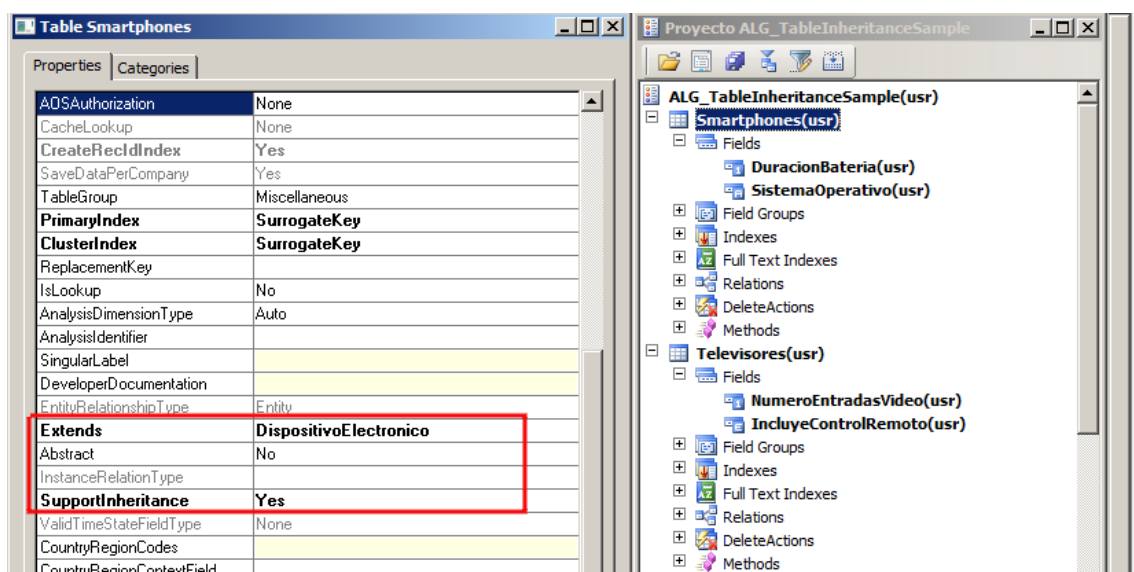


Tal como hemos comentado, creamos la tabla con los campos comunes entre todos los dispositivos y prestamos mucha atención a las zonas remarcadas en rojo en la imagen:

- Debemos crear un campo de tipo **Int64** que se encargará de apuntar a la tabla derivada que implementa la especialización del elemento (no os preocupéis, esto lo gestionará internamente el sistema, pero debemos crear el campo)
- Asignamos este campo nuevo a la propiedad **InstanceRelationType** de la tabla
- Marcamos la propiedad **SupportInheritance** de la tabla a YES.

NOTA: Se puede observar que existe la propiedad **ABSTRACT**. Estableceremos el valor **YES** para esta propiedad cuando estemos definiendo un tipo abstracto que no va a contener datos y solo sirve como base para sus derivadas. No abordaremos este tema en este artículo.

Ahora vamos a crear las dos tablas derivadas...



Prestamos atención a que en cada una de las tablas derivadas debemos indicar:

- Indicamos que usaremos herencia. Propiedad SupportInheritance = YES
- Indicamos la tabla base. Propiedad Extends = DispositivoElectronico

Una vez ya hemos creado la estructura de tablas, ahora vamos a introducir algunos datos para poder jugar un poco (lo hago mediante un job para que sea fácil para el lector de este artículo poder recrear el escenario sin tener que introducir los datos a mano).

```
static void ALG_TableInheritanceDataSample(Args _args)
{
    Smartphones SmartPhones;
    Televisores Televisores;
    ;

    // Insertamos un Iphone 4
    Smartphones.clear();
    Smartphones.initValue();
    Smartphones.DispositivoElectronicoId="DVC001";
    Smartphones.Marca = "Apple";
    Smartphones.Modelo = "IPhone 4";
    Smartphones.FechaCompra = 01\06\2011;
    Smartphones.DuracionBateria = 24;
    Smartphones.SistemaOperativo = "iOS 4";
    Smartphones.insert();

    // Insertamos un Samsung Galaxy S II
    Smartphones.clear();
    Smartphones.initValue();
    Smartphones.DispositivoElectronicoId="DVC002";
    Smartphones.Marca = "Samsung";
    Smartphones.Modelo = "Galaxy S II";
    Smartphones.FechaCompra = 15\06\2011;
    Smartphones.DuracionBateria = 22;
    Smartphones.SistemaOperativo = "Android";
    Smartphones.insert();

    // Insertamos un HTC HD7
    Smartphones.clear();
    Smartphones.initValue();
    Smartphones.DispositivoElectronicoId="DVC003";
    Smartphones.Marca = "HTC";
    Smartphones.Modelo = "HD7";
    Smartphones.FechaCompra = 01\08\2011;
    Smartphones.DuracionBateria = 30;
    Smartphones.SistemaOperativo = "Windows Phone 7";
    Smartphones.insert();

    // Insertamos un televisor Sony
    Televisores.clear();
    Televisores.initValue();
    Televisores.DispositivoElectronicoId="DVC004";
    Televisores.Marca = "Sony";
    Televisores.Modelo = "KDL-40EX720";
    Televisores.FechaCompra = 05\06\2011;
    Televisores.IncluyeControlRemoto = NoYes::Yes;
    Televisores.NumeroEntradasVideo = 5;
    Televisores.insert();

    // Insertamos un televisor LG
    Televisores.clear();
    Televisores.initValue();
    Televisores.DispositivoElectronicoId="DVC005";
```



```

Televisores.Marca = "LG";
Televisores.Modelo = "42LW5500";
Televisores.FechaCompra = 21\07\2011;
Televisores.IncluyeControlRemoto = NoYes::Yes;
Televisores.NumeroEntradasVideo = 6;
Televisores.insert();
}
    
```

Si ejecutamos el código anterior, se nos crearán 3 Smartphones y 2 Televisores en nuestras tablas y por tanto, ahora podemos formularnos estas sencillas preguntas :

- ¿Cuántos televisores tenemos? : Dos
- ¿Cuántos Smartphones tenemos? : Tres
- ¿Cuántos dispositivos tenemos en total? : Cinco

Esto queda claramente reflejado si abrimos el examinador de tablas para cada una de nuestras tres tablas:

TELEVISORES

Examinador de tablas: Televisores (2 - cere)

Dispositiv...	FechaCompra	I.	InstanceRelationType	Marca	Modelo	Numer...	dataAr...	recV
DVC004	05/06/2011	<input checked="" type="checkbox"/>	100547	Sony	KDL-40EX720	5	cere	1
DVC005	21/07/2011	<input checked="" type="checkbox"/>	100547	LG	42LW5500	6	cere	1

SELECT * FROM Televisores

SMARTPHONES

Examinador de tablas: Smartphones (2 - cere)

Dispositiv...	Duracio...	FechaCompra	InstanceRelationType	Marca	Modelo	SistemaOperativo
DVC001	24	01/06/2011	100548	Apple	IPhone 4	iOS 4
DVC002	22	15/06/2011	100548	Samsung	Galaxy S II	Android
DVC003	30	01/08/2011	100548	HTC	HD7	Windows Phone 7

SELECT * FROM Smartphones



DISPOSITIVOS ELECTRÓNICOS

Examinador de tablas: DispositivoElectronico (2 - cere)

Dispositiv...	FechaCompra	InstanceRelationType	Marca	Modelo	dataAr...	recV...	relationTy
DVC001	01/06/2011	100548	Apple	IPhone 4	cere	1	100548
DVC002	15/06/2011	100548	Samsung	Galaxy S II	cere	1	100548
DVC003	01/08/2011	100548	HTC	HD7	cere	1	100548
DVC004	05/06/2011	100547	Sony	KDL-40EX720	cere	1	100547
DVC005	21/07/2011	100547	LG	42LW5500	cere	1	100547

SELECT * FROM DispositivoElectronico

¿Cómo funciona?

Como podemos comprobar en la sección anterior, al insertar registros en las tablas de Smartphones y Televisores, realmente estábamos también insertando esos registros en la tabla de dispositivos electrónicos (que es la tabla base de las dos anteriores).

Pero ¿cómo?, ¿se están entonces duplicando los datos en las 3 tablas? Vamos a echar un vistazo directamente desde SQL para comprobarlo.

Disp.Electrónico (tableid: 100546)

	DATAAREAD	RE...	RECID	DISPOSI...	FECHACOMPRA	MARCA	MODELO	INSTANCERELATIONTYPE	RELATIONTYPE
1	cere	1	5637144576	DVC001	2011-06-01 00:00:00.000	Apple	IPhone 4	100548	100548
2	cere	1	5637144577	DVC002	2011-06-15 00:00:00.000	Samsung	Galaxy S II	100548	100548
3	cere	1	5637144578	DVC003	2011-08-01 00:00:00.000	HTC	HD7	100548	100548
4	cere	1	5637144579	DVC004	2011-06-05 00:00:00.000	Sony	KDL-40EX720	100547	100547
5	cere	1	5637144580	DVC005	2011-07-21 00:00:00.000	LG	42LW5500	100547	100547

SmartPhones (tableid: 100548)

	DATAAREAD	RECVERSION	RECID	SISTEMAOPERATIVO	RELATIONTYPE	DURACIONBATERIA
1	cere	1	5637144576	iOS 4	0	24
2	cere	1	5637144577	Android	0	22
3	cere	1	5637144578	Windows Phone 7	0	30

Televisores (tableid: 100547)

	DATAAREAD	RECVERSION	RECID	RELATIONTYPE	INCLUYECONTROLREMOTO	NUMEROENTRADASVIDEO
1	cere	1	5637144579	0	1	5
2	cere	1	5637144580	0	1	6

Como podemos observar en las imágenes, cada una de las tablas contiene únicamente aquellos campos que le son propios y una tabla base siempre apunta a la tabla derivada que representa a cada uno de sus registros (esto se cumple siempre incluso de forma “recursiva” si tenemos en cuenta que podemos crear N niveles de herencia).

He pintado con círculos de colores aquellos campos que relacionan estas tablas entre sí de forma que tenemos :

- La tabla base usa el campo que le hemos indicado “InstanceRelationId” para almacenar el **TABLEID** de la tabla derivada que implementa la especialización de cada registro base.
- Los registros de la tabla base y la tabla derivada poseen el mismo **RECID**.

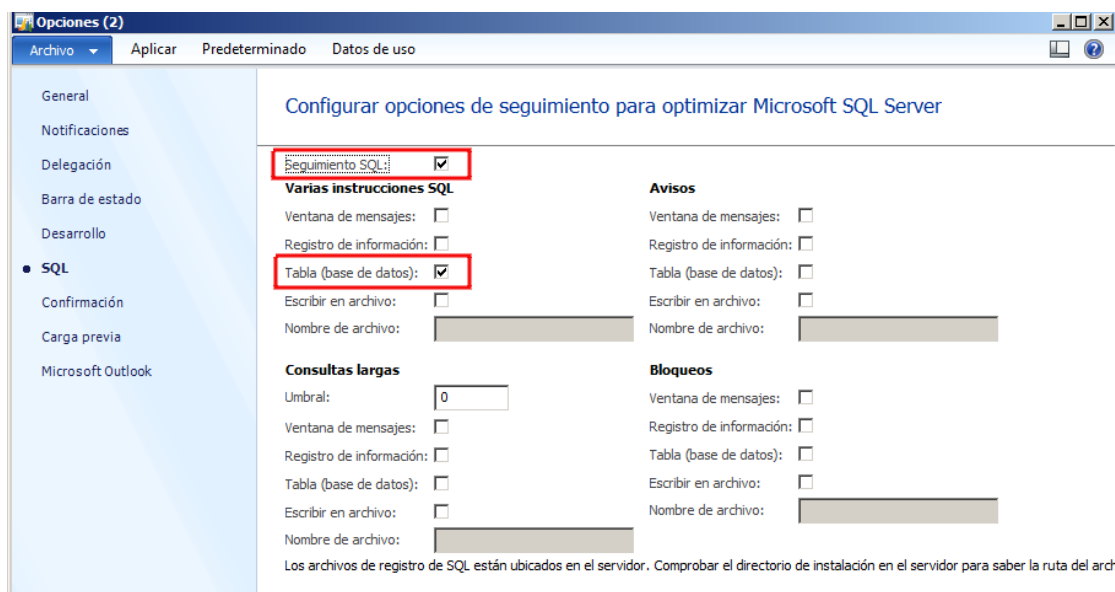
Por tanto, podemos concluir que cuando desde AX abrimos una de las tablas derivadas, el sistema monta la consulta de múltiples tablas y nos muestra los datos como si de una sola tabla se tratara. Vamos a comprobarlo...

Examinando las sentencias SQL

Vamos a marcar el seguimiento de SQL de AX para poder echar un vistazo a las instrucciones SQL que se ejecutan cada vez que abrimos una de las tablas que hemos creado.

Seguimos estos pasos:

1. Dentro de AX, en Herramientas / Opciones activamos el seguimiento de SQL:



2. Luego, vamos al AOT y abrimos la tabla de Televisores con el examinador de tablas.
3. A continuación ya podemos volver a desactivar el seguimiento SQL (para que el sistema deje de guardar las consultas SQL en la tabla y nos sea más fácil encontrar lo que buscamos).
4. Ahora vamos a revisar las consultas guardadas en la opción de menú *[Administración del sistema/ Consultas / Base de datos / Registro de seguimiento de instrucción SQL]* (De entre todas las consultas existentes debemos encontrar la nuestra, la que realiza un “SELECT” sobre la tabla de Televisores)

Tenemos que, el hecho de abrir la tabla de Televisores, ha provocado que el sistema realice la siguiente consulta SQL:

```
SELECT
T1 . INSTANCERELATIONTYPE, T1 . DISPOSITIVOELECTRONICOID, T1 . FECHACOMPRA,
    T1 . MARCA, T1 . MODELO, T1 . RECVERSION, T1 . RELATIONTYPE, T1 . RECID, T2 . INC
LUYECONTROLREMOTO,
    T2 . NUMEROENTRADASVIDEO, T2 . RECVERSION, T2 . RELATIONTYPE, T2 . RECID
FROM DISPOSITIVOELECTRONICO T1 CROSS JOIN TELEVISORES T2
WHERE (T1 . DATAAREAID=? ) AND ( (T2 . DATAAREAID=? ) AND
(T2 . RECID=T1 . RECID) )
ORDER BY T1 . RECID OPTION (FAST 19)
```

Como podemos observar, cuando abrimos una de las tablas derivadas, AX se encarga de gestionar las consultas SQL internamente para obtener datos de la tabla base y mostrarlo todo como un solo registro.

Sigamos jugando: ¿Que pasaría si en un Job realizo un “SELECT” sobre la tabla de Dispositivos electrónicos (tabla base)?

Creemos un sencillo Job y lo ejecutamos:

```
static void ALG_TableInheritanceSelect1(Args _args)
{
    DispositivoElectronico DispositivoElectronico;
    ;

    select DispositivoElectronico;
}
```

Tenemos que la sentencia SQL generada es:

```
SELECT
T1 . INSTANCERELATIONTYPE, T1 . DISPOSITIVOELECTRONICOID, T1 . FECHACOMPRA, T1 .
MARCA, T1 . MODELO,
T1 . RECVERSION, T1 . RELATIONTYPE, T1 . RECID, T2 . SISTEMAOPERATIVO, T2 . DURACION
BATERIA, T2 . RECVERSION,
T2 . RELATIONTYPE, T2 . RECID, T3 . INCLUYECONTROLREMOTO, T3 . NUMEROENTRADASVIDE
O, T3 . RECVERSION,
T3 . RELATIONTYPE, T3 . RECID
FROM DISPOSITIVOELECTRONICO T1 LEFT OUTER JOIN SMARTPHONES T2
ON ( (T2 . DATAAREAID=? ) AND (T1 . RECID=T2 . RECID) ) LEFT OUTER JOIN
TELEVISORES T3
ON ( (T3 . DATAAREAID=? ) AND (T1 . RECID=T3 . RECID) )
WHERE (T1 . DATAAREAID=? )
```



Y eso significa que el sistema realiza una “join” de la tabla base con cada una de sus derivadas!

¿Y que pasa si modificamos nuestro job y en lugar de ejecutar un “Select DispositivoElectronico” (que equivale a “Select * from DispositivoElectronico”) seleccionamos los campos de la tabla de dispositivos electrónicos que realmente deseamos?

Modificamos el job y pedimos solo campos de la tabla base:

```
static void ALG_TableInheritanceSelect1(Args _args)
{
    DispositivoElectronico DispositivoElectronico;
    ;

    select DispositivoElectronicoId, Marca, Modelo, FechaCompra from DispositivoElectronico;
}

```

Y tenemos que:

```
SELECT
T1.DISPOSITIVOELECTRONICOID, T1.MARCA, T1.MODELO, T1.FECHACOMPRA, T1.RECID
, T1.INSTANCERELATIONTYPE
FROM DISPOSITIVOELECTRONICO T1
WHERE (T1.DATAAREAID=?)

```

El sistema esta vez solo ejecuta la consulta sobre la tabla base.

¡IMPORTANTE! : Si no somos cuidadosos a la hora de elegir los campos que deseamos obtener de una consulta SQL, podemos penalizar el rendimiento.