****

**SISTEMAS COMPUTACIONALES**

**Administración de bases de datos**

***Reporte final del “proyecto c”***

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | ***Estudiantes*** | | | | ***Yair Alberto Herrera Duarte*** | ***13041395*** | | | ***Martín Ramírez García*** | ***13041421*** | | | ***Aziel Eliú León Díaz*** | ***13041400*** | | | ***Mario Álvaro Castro Barrón*** | ***13041365*** | | | *Nombre del Facilitador* | | | | ***Jose Gabriel Rodriguez Rivas*** | | | |  | | ***Grupo: 7Y*** | |  |
|  |  |

Victoria de Durango, Dgo; A 30 de Enero del 2016

**Índice**

[Introducción 1](#_Toc452603942)

[Proyecto C: 2](#_Toc452603943)

[Definición de tablas 3](#_Toc452603944)

[Descripción de cada tabla y sus atributos 4](#_Toc452603945)

[Diagrama Chen entidad-relación 13](#_Toc452603946)

[Reportes 14](#_Toc452603947)

[Reporte 1 14](#_Toc452603948)

[Reporte 2: 15](#_Toc452603949)

[Reporte 3: 16](#_Toc452603950)

[Reporte 4 18](#_Toc452603951)

[Reporte 5 20](#_Toc452603952)

[Reporte 6: 22](#_Toc452603953)

[Definición del esquema de partición 24](#_Toc452603954)

[Proceso de respaldo 34](#_Toc452603955)

[Replicación de datos 42](#_Toc452603956)

[Servidor donde se replicaron los datos 57](#_Toc452603957)

[En el servidor donde queramos la replicación de estos datos, por medio de una publicación desde otra instancia, damos a “Nueva suscripción” 57](#_Toc452603958)

[Conclusiones 74](#_Toc452603959)

[Observaciones durante el desarrollo del proyecto 75](#_Toc452603960)

**Introducción**

Se dice que la información es poder: entonces, ¿Que será el control total de la información?

En plena era de la información, cada vez es más fácil ver la importancia de las bases de datos y su uso desde nuestra vida cotidiana, hasta en situaciones que invocaran valores monetarias.

Durante este curso comprobamos la importancia de un buen manejador de base de datos y a la vez lo elemental de un administrador de base de datos.

Este proyecto trata de evidenciar el aprendizaje durante el curso que nosotros como alumnos obtuvimos tanto de manera teórica como práctica.

El presente documento también muestra el resultado de otras materias que han sido los escalones del conocimiento de las bases de datos, que con anterioridad nos ayudaron a comprender y aplicar desde los fundamentos de las bases de datos, hasta la creación y gestión de las mismas.

Con orgullo presentamos el informe final del proyecto C para concluir con esta materia.

Proyecto C:

Desarrollar un sistema de base de datos, aplicando esquemas de seguridad, rendimiento y alta disponibilidad (Replicación de datos), especificando la forma de almacenamiento de los archivos de datos y de registro para la base de datos, además de la configuración de las bitácoras de registro del proyecto optimizando los recursos económicos y de infraestructura tecnológica a partir del siguiente enunciado.

**PROTECKT** empresa de seguridad integral, dedicada a la vigilancia desea tener una base de datos para facilitar la gestión de la información de sus clientes, es decir, de los distintos fraccionamientos que administra. La información debe contener los aspectos que a continuación se mencionan.

La empresa tiene varios jefes de seguridad y cada uno de ellos ejerce de administrador de uno o más fraccionamientos de vecinos, por lo que cobra a cada uno de ellos honorarios mensuales. Un fraccionamiento es gestionado por un único jefe de seguridad (Nombre, NSS, RFC, domicilio, teléfono). Las funciones de un jefe de seguridad sobre las que interesa guardar información, consisten en llevar la contabilidad del fraccionamiento, gestionando los recibos que pagan los vecinos mensualmente, así como los pagos a las distintas compañías que proporcionan algún servicio al fraccionamiento (limpieza, gas, jardinería, cable, sky, internet, teléfono, etc).

De las empresas que tiene contratadas los distintos fraccionamientos (ej, Telmex, megacable, CFE, GlobalGas, etc) se guarda su razón social, nombre, dirección y persona de contacto. Además interesa tener estas compañías agrupadas por sectores (luz, servicio TV paga, etc).

De cada fraccionamiento interesa almacenar algún código identificador, nombre, ubicación, CP, localidad, ciudad y estado. Cada fraccionamiento consta de una serie de propiedades que pueden ser de tres tipos (vivienda particular, local comercial y oficina). Cada propiedad se caracteriza por un número de portal, planta y letra, un nombre y apellidos del propietario con su dirección completa (que puede ser esta u otra) y un teléfono de contacto, un porcentaje de participación en los gastos del fraccionamiento así como los datos de la cuenta bancaria en la que el propietario desea se le domicilie el pago de los recibos.

Si el propietario no habita en su propiedad entonces se necesita sus datos (nombre, domicilio, teléfono de contacto, imagen de la credencial del IFE y foto) así como los del inquilino que la habita (nombre, domicilio, teléfono de contacto, imagen de la credencial del IFE y foto). Si el propietario habita en la propiedad solo son necesarios los datos (nombre)

Instituto Tecnológico de Durango M.T.I. José Gabriel Rodríguez Rivas Página 4 Si se trata de vivienda se almacena el número de habitaciones, si es local comercial se almacena el tipo de comercio que se desarrolla en él, y el horario. Si es una oficina se guarda la actividad a la que se destina.

Cada fraccionamiento tiene además un presidente y 3 vocales (nombre y apellidos y propiedad de la que son dueños) que se encargan de tratar directamente con el jefe de seguridad los problemas que puedan surgir.

En cuanto a la contabilidad, cada fraccionamiento tiene una cuenta en un banco. De los distintos bancos se guarda la sucursal, nombre, dirección y una persona de contacto. Mientras que para una cuenta bancaria se guarda un código de cuenta (consta de un código de sucursal, 2 dígitos de control y un número de cuenta y un saldo. Para identificar una cuenta es necesario añadir al código de cuenta el código el banco donde se encuentra.

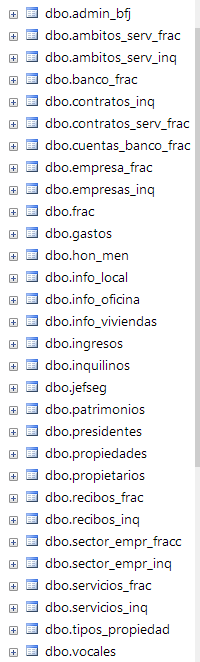
Es necesario almacenar dos tipos de cuentas (ingresos y gastos) para la contabilidad de cada fraccionamiento.

Por un lado, aunque es el banco el que emite los recibos de las cuotas a los vecinos de los fraccionamientos, el jefe de seguridad guarda información sobre dichos recibos que se ingresan en las cuentas bancarias de los fraccionamientos, es decir, número de recibo, fecha, importe y si se ha podido cobrar o no. Esta última información es importante para realizar a final de cada trimestre una relación de cuentas no pagadas.

En cuanto a las cuentas relativas a los gastos se tienen los impuestos que cobran las empresas contratadas por cada fraccionamiento. Las compañías cobran sus recibos

Definición de tablas

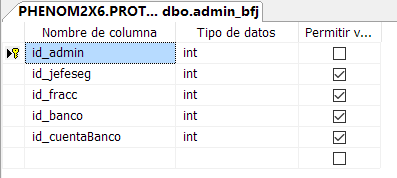
A continuación se muestran las propiedades de las tablas y su descripción:

En total, son 30 tablas normalizadas las que componen el proyecto.

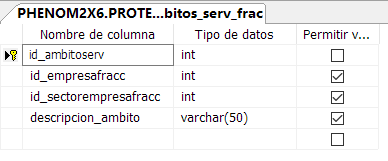
## Descripción de cada tabla y sus atributos

A continuación se muestran las imágenes de los resultados de cada uno de los 6 reportes que involucran este proyecto:

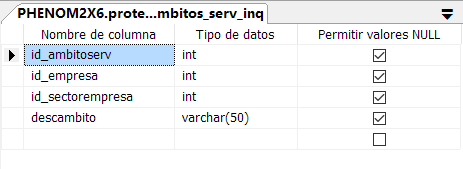
**Admin\_bfg**

Esta tabla contiene las ids que permiten enlazar a los bancos que usa cada fraccionamiento, los fraccionamientos y su respectivo jefe de seguridad.

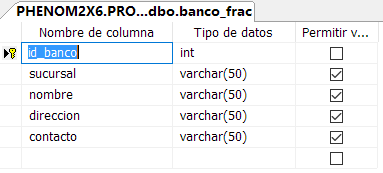
**Ámbitos\_serv\_frac**

Esta tabla permite que cada empresa contratada por un fraccionamiento, tenga la capacidad de prestar más de un tipo de servicio.

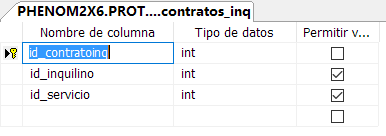
**ambitos\_serv\_inq**

Esta tabla permite que cada empresa contratada por un fraccionamiento, tenga la capacidad de prestar más de un tipo de servicio.

**banco\_frac**

Contiene todos los datos del banco que gestiona las cuentas bancarias de los fraccionamientos.

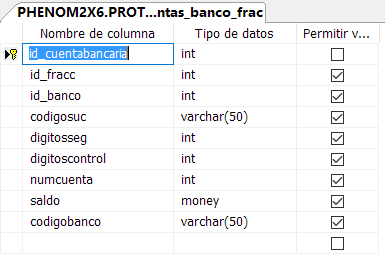
**Contratos\_inq**

Permite que un inquilino tenga contratados varios servicios a diferentes empresas.

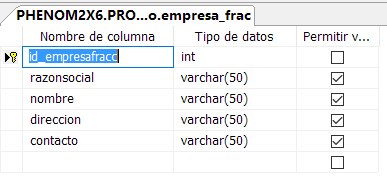
**Contratos\_serv\_frac**

Permite que un fraccionamiento pueda contratar varios servicios a diferentes empresas

**Cuentas\_banco-frac**

Contiene los datos de la cuenta bancaria que usan los fraccionamientos.

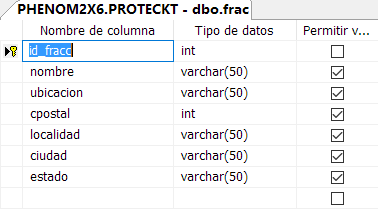
**Empresa\_fracc**

Contiene los datos de las empresas que ofrecen algún servicio a un fraccionamiento

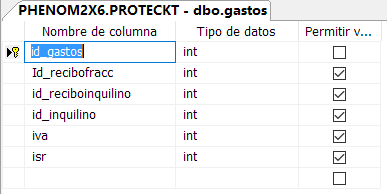
**Empresa\_inq**

Contiene los datos de las empresas que ofrecen algún servicio a un inquilino

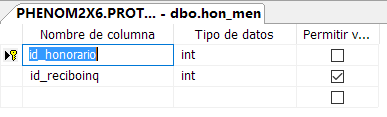
**frac**

Contiene todos los datos de los fraccionamientos que gestionan los jefes de seguridad

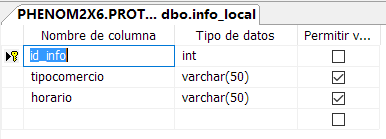
**gastos**

Contiene los datos para recuperar aquellos recibos que se consideran gastos pendientes por pagar por parte de los inquilinos de un fraccionamiento. También incluye los impuestos y el porcentaje de cada recibo.

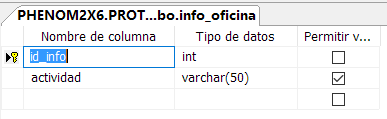
**Hon\_men**

contiene el id necesario para saber cuáles recibos de pagos, por parte de los inquilinos, son “honorarios” directos al jefe de seguridad.

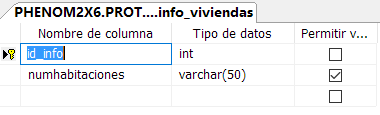
**info\_local**

Contiene los datos de aquellas propiedades que son del tipo “local comercial”.

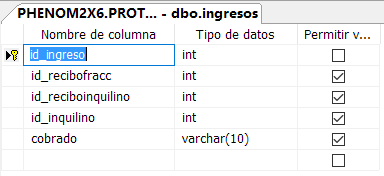
**Info\_oficina**

Contiene los datos de aquellas propiedades que son del tipo “oficina”.

Info\_viviendas

Contiene los datos de aquellas propiedades que son del tipo “vivienda”.

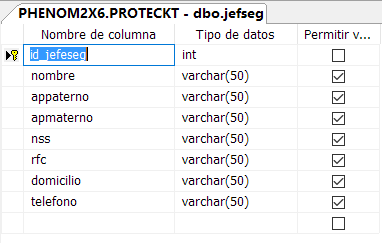
**Ingresos**

Contiene los ids necesarios para recuperar los datos de los recibos que son considerados como “ingresos” y muestra su estado de “cobrado”.

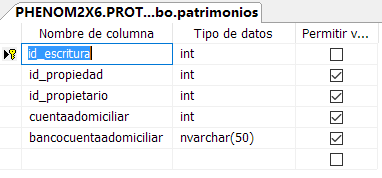
**Inquilinos**

Contiene todos los datos de los “inquilinos” de una propiedad.

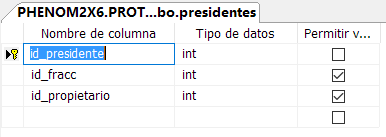
**Jefseg**

Contiene todos los datos de los jefes de seguridad de los fraccionamientos gestionados por la compañía “proteckt”.

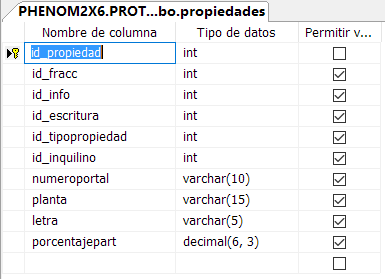
**Patrimonio:**

Permite que un “propietario” pueda poseer varias “propiedades” y rentarlas, a la vez de contener una cuenta bancaria y el nombre del banco para que el “inquilino” que renta la propiedad, pueda efectuar sus “rentas”.

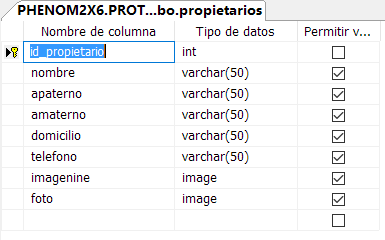
**Presidentes**

Contiene los ids tanto del “propietario” que será el presidente de un fraccionamiento, como el id de dicho fraccionamiento.

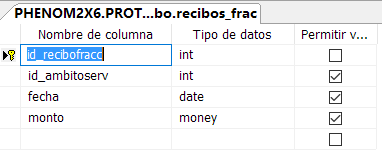
**Propiedades**

Contiene toda la información para saber tanto los datos de una propiedad, como su fraccionamiento, su información propia de su tipo, escritura y el inquilino que la habita.

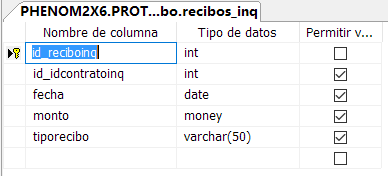
**propietarios**

Contiene toda la información de aquellos que son “propietarios” de una o varias propiedades.

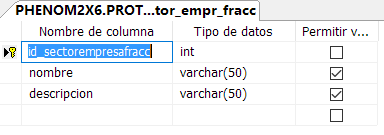
**recibos\_frac**

Contiene la fecha y monto de aquellos recibos generados por las empresas contratadas por un fraccionamiento.

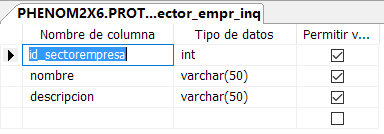
**Recibos\_inq**

Contiene la fecha y monto y tipo de recibo de aquellos recibos generados por las empresas contratadas por los inquilinos.

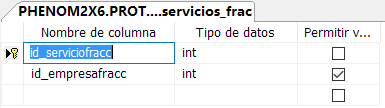
**Sector\_empr\_frac**

Contiene los datos que ayudan a catalogar a las empresas contratadas por un fraccionamiento, en sus respectivos sectores de servicio (agua, luz, teléfono, etc.)

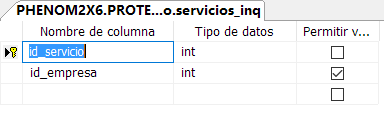
**Sector\_empr\_inq**

Contiene los datos que ayudan a catalogar a las empresas contratadas por los inquilinos en sus respectivos sectores de servicio (agua, luz, teléfono, etc.)

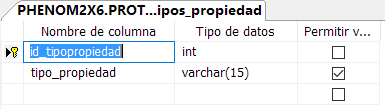
**Servicios\_frac**

Permite que una empresa pueda brindar varios servicios a un fraccionamiento, o que un servicio pueda ser brindado por varias empresas distintas.

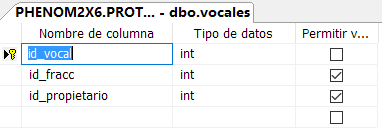
**servicios\_inq**

Permite que una empresa pueda brindar varios servicios a un inquilino, o que un servicio pueda ser brindado por varias empresas distintas.

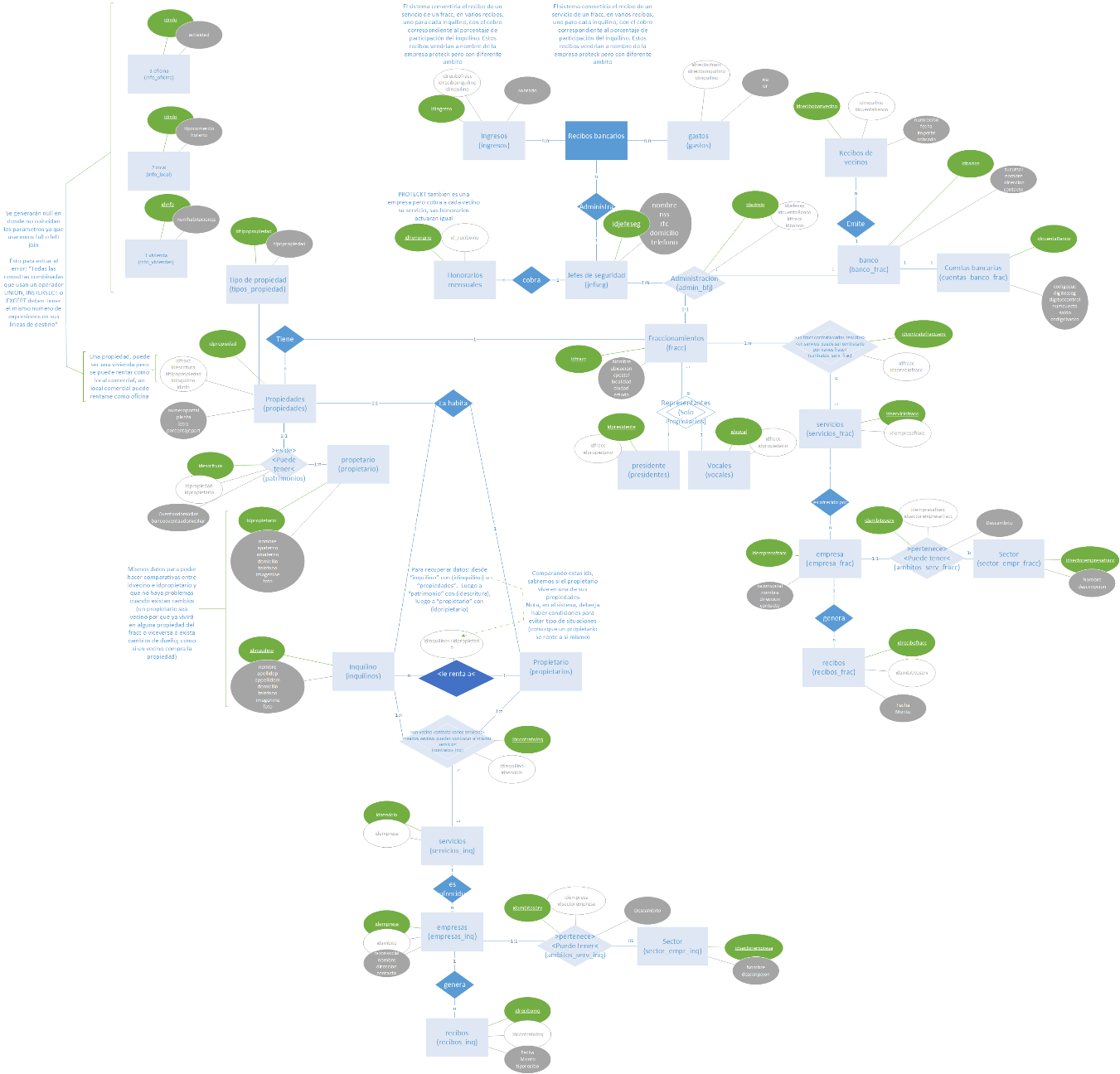
**Tipos\_propiedad**

Contiene el nombre de los tipos de propiedad, para evitar repetir este dato

**Vocales**

Contiene los ids para relacionar a aquellos propietarios que sirven como “vocales” en cierto fraccionamiento

## Diagrama Chen entidad-relación



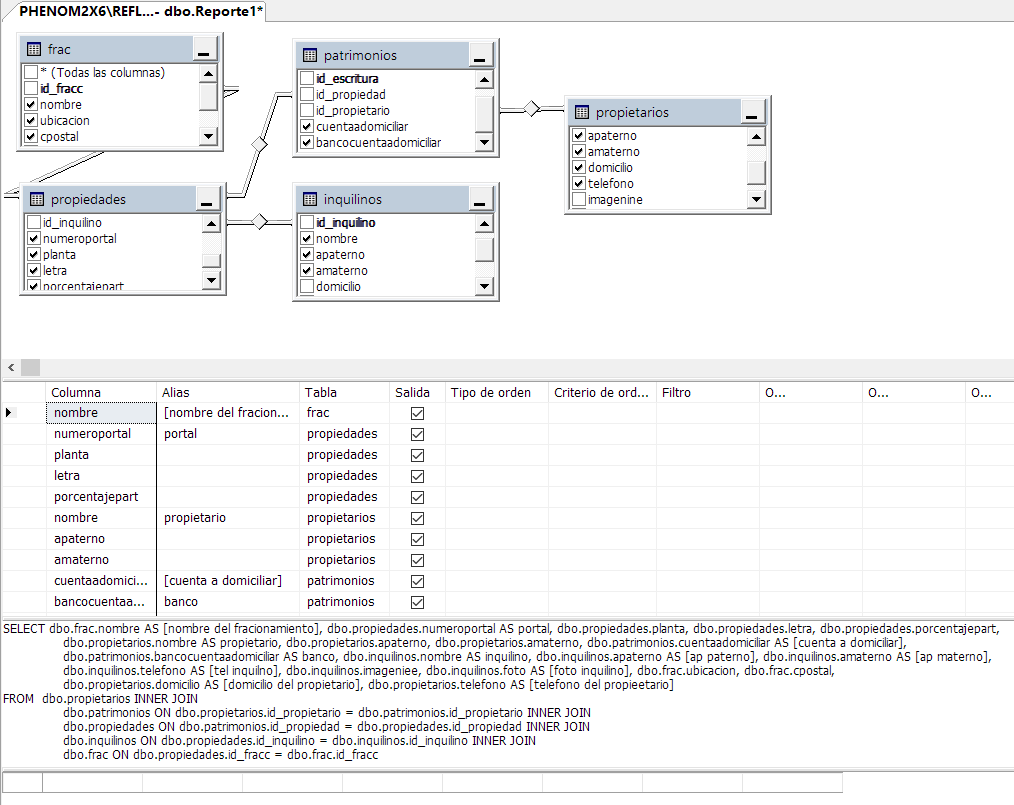
Reportes

A continuación se muestran los reportes y sus resultados:

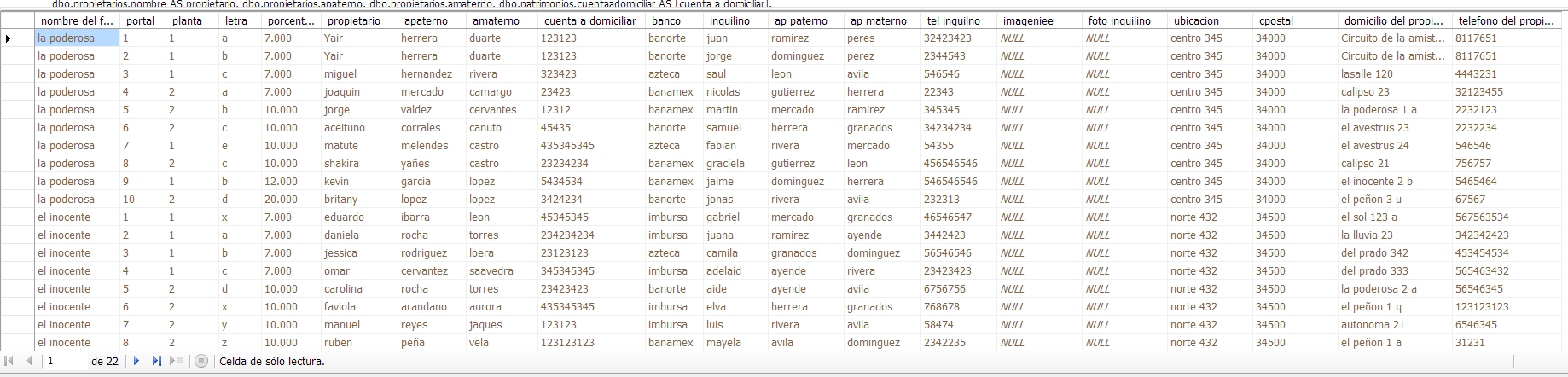
## Reporte 1

Reporte de las casas que se están rentando.

• Incluir los datos de la propiedad, propietario y del inquilino.



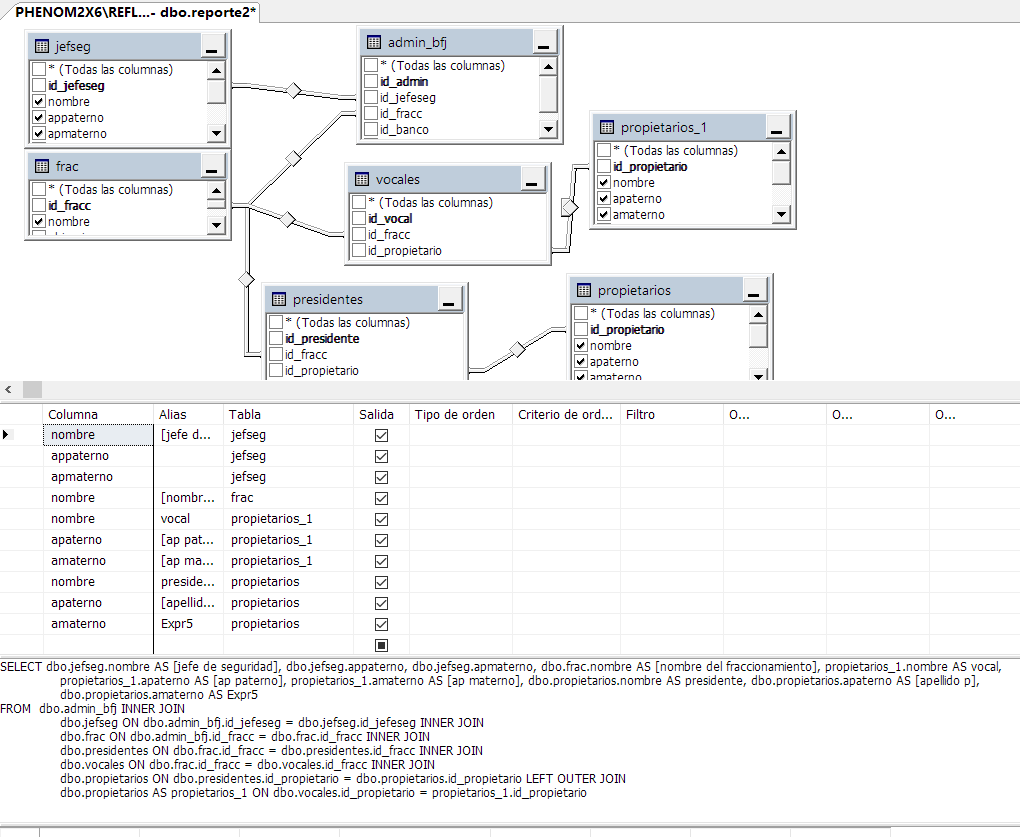
Resultados:



## Reporte 2:

Reporte de todos los fraccionamientos.

• Incluir Nombre del fraccionamiento, jefe de seguridad, presidente y vocales de cada fraccionamiento.

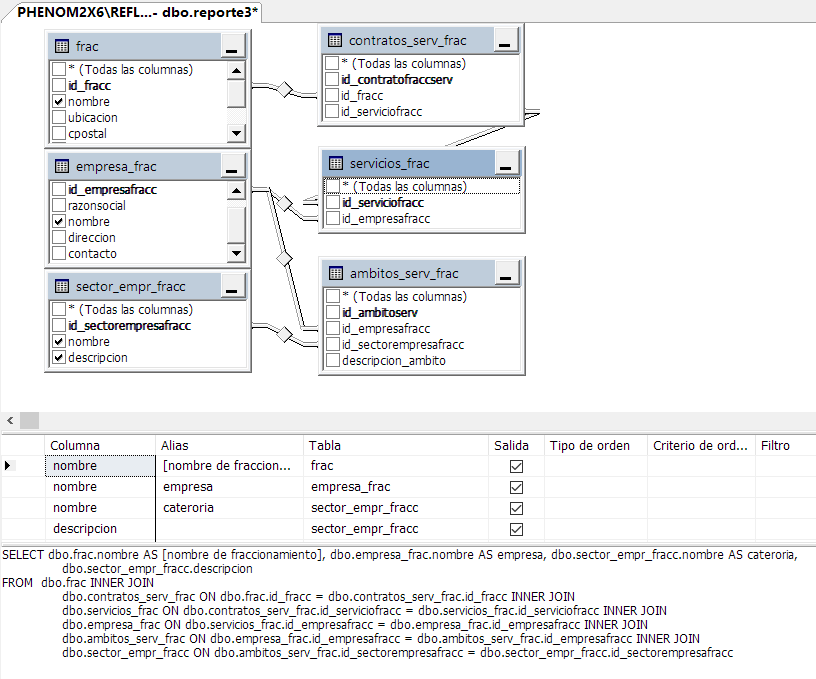


Resultados:



## Reporte 3:

Reporte de servicios por categoría. • Incluir categoría y los servicios que presenta cada uno.



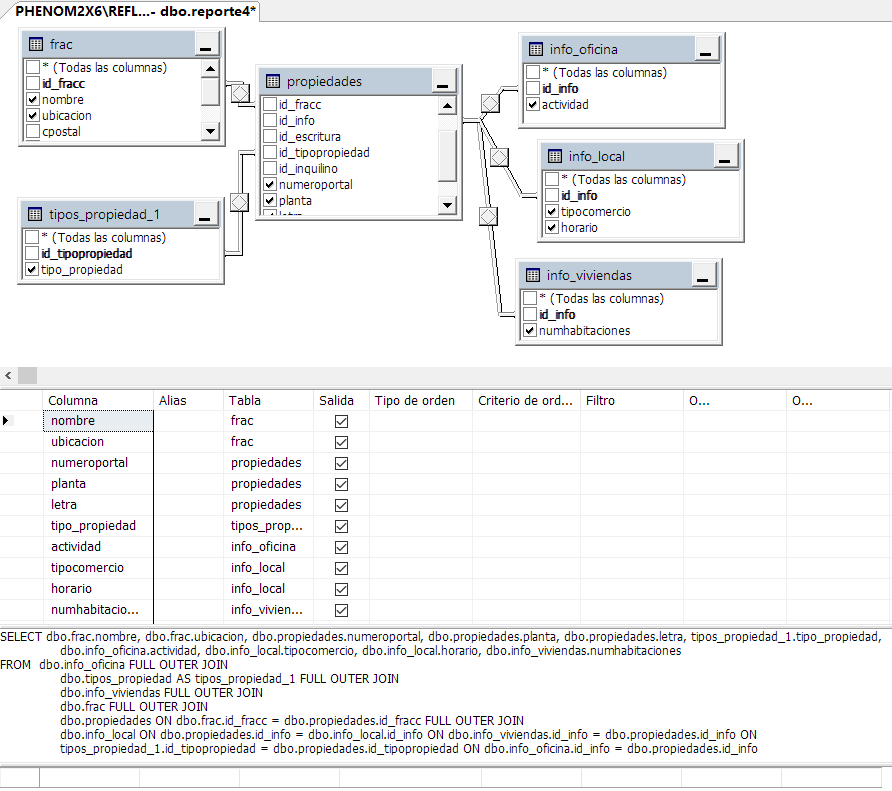
Resultado:



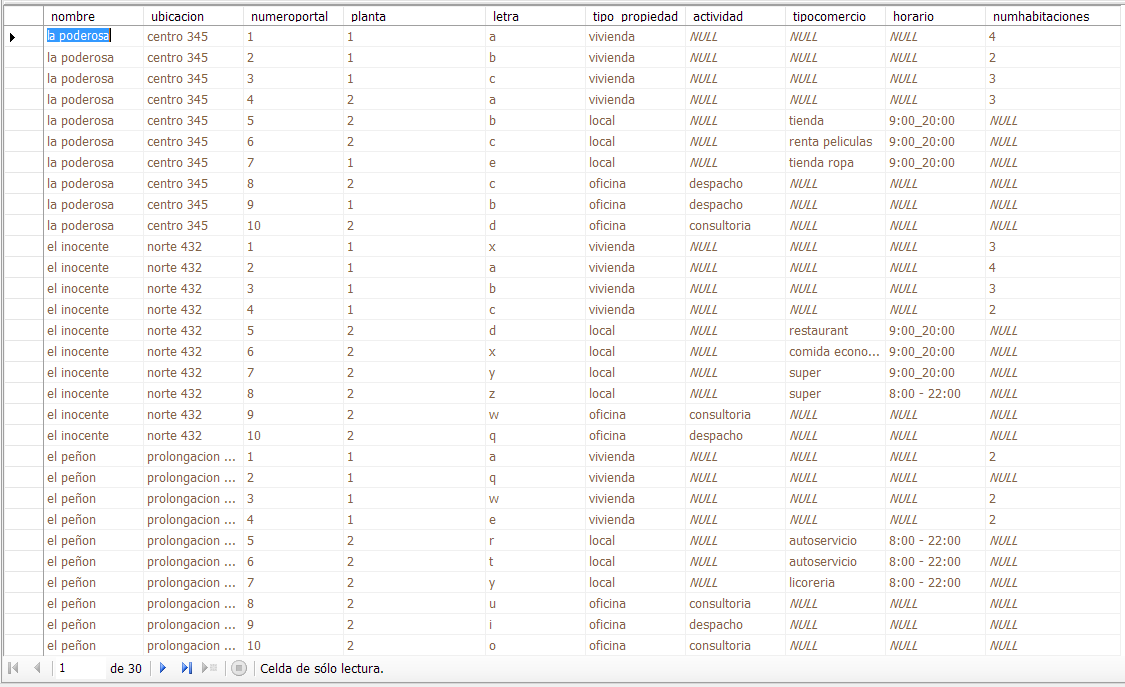
## Reporte 4

Reporte de uso de cada propiedad.

• Incluir fraccionamiento, dirección, indicar si se trata de vivienda y el número de habitaciones, si es local comercial indicar el tipo de comercio y si es una oficina la actividad a la que se destina.



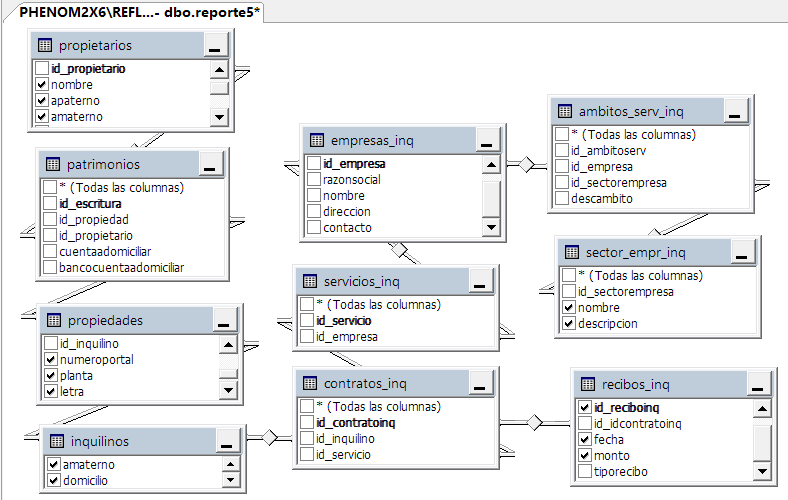
Resultado:

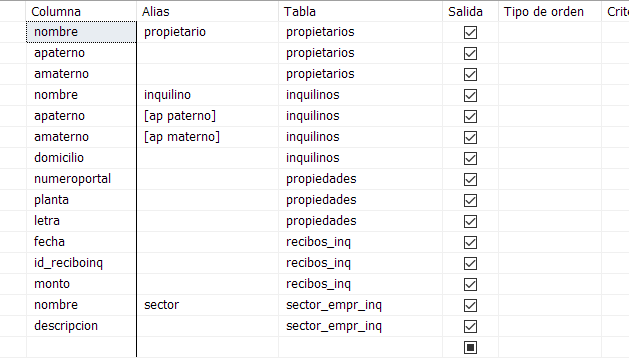


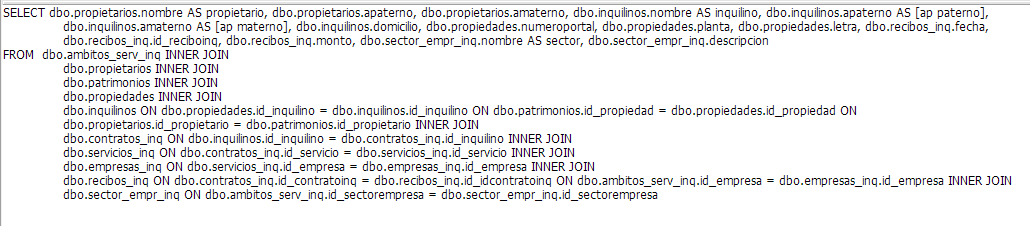
## Reporte 5

Reporte desglosado de pagos por propiedad y fecha

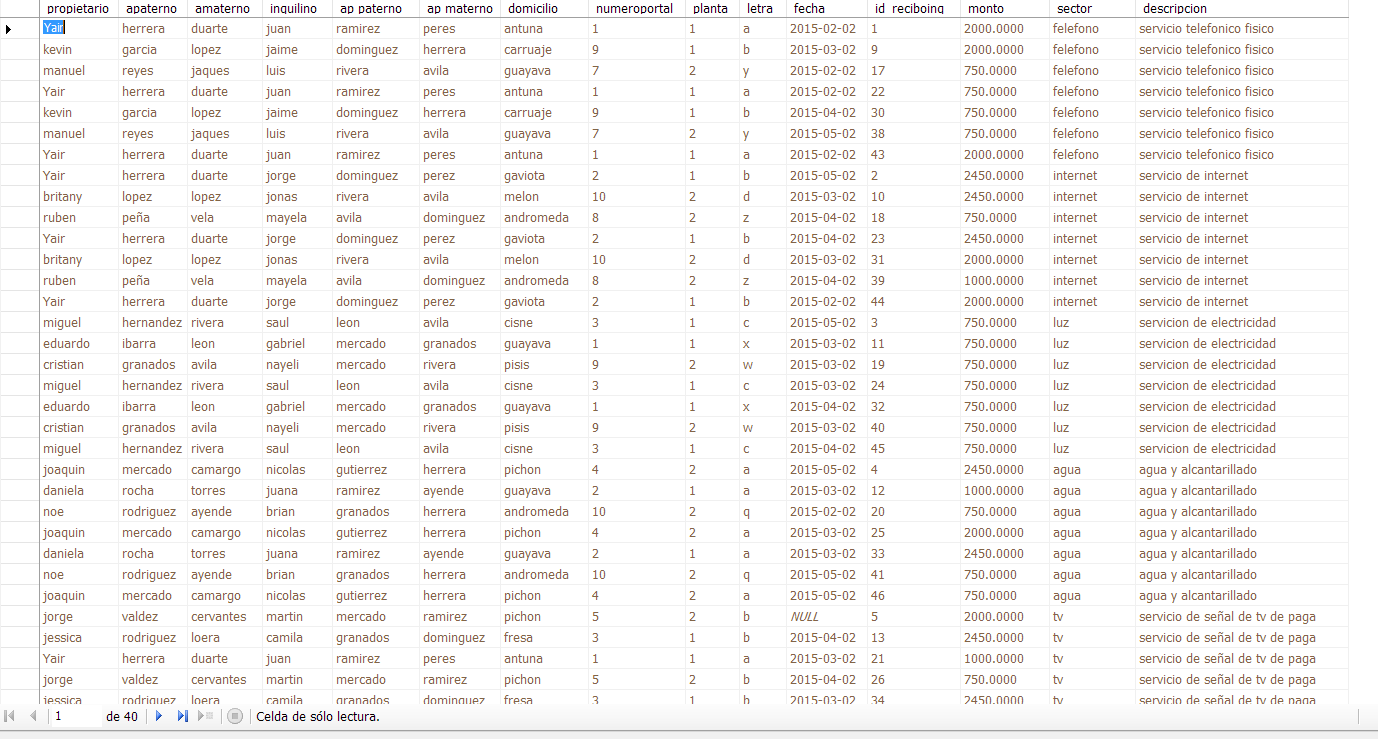
• Incluir propietario, domicilio, fecha y los pagos que se efectúa por la propiedad (luz, teléfono, agua , etc.)







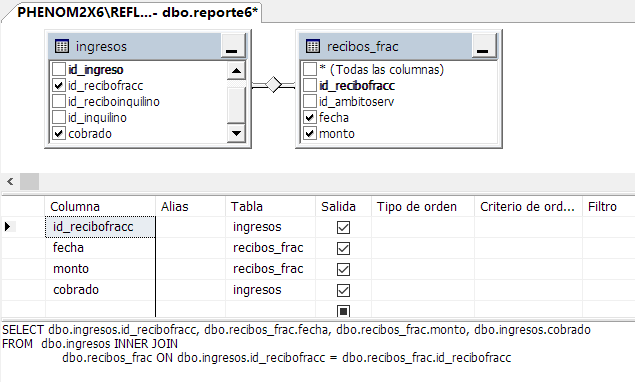
Resultado



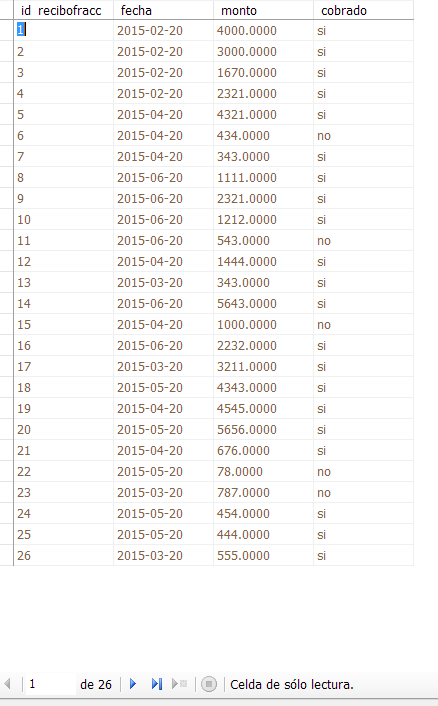
## Reporte 6:

Reporte de la contabilidad del fraccionamiento

• Incluir número de recibo, fecha, importe y si se ha podido cobrar o no.



Resultado:

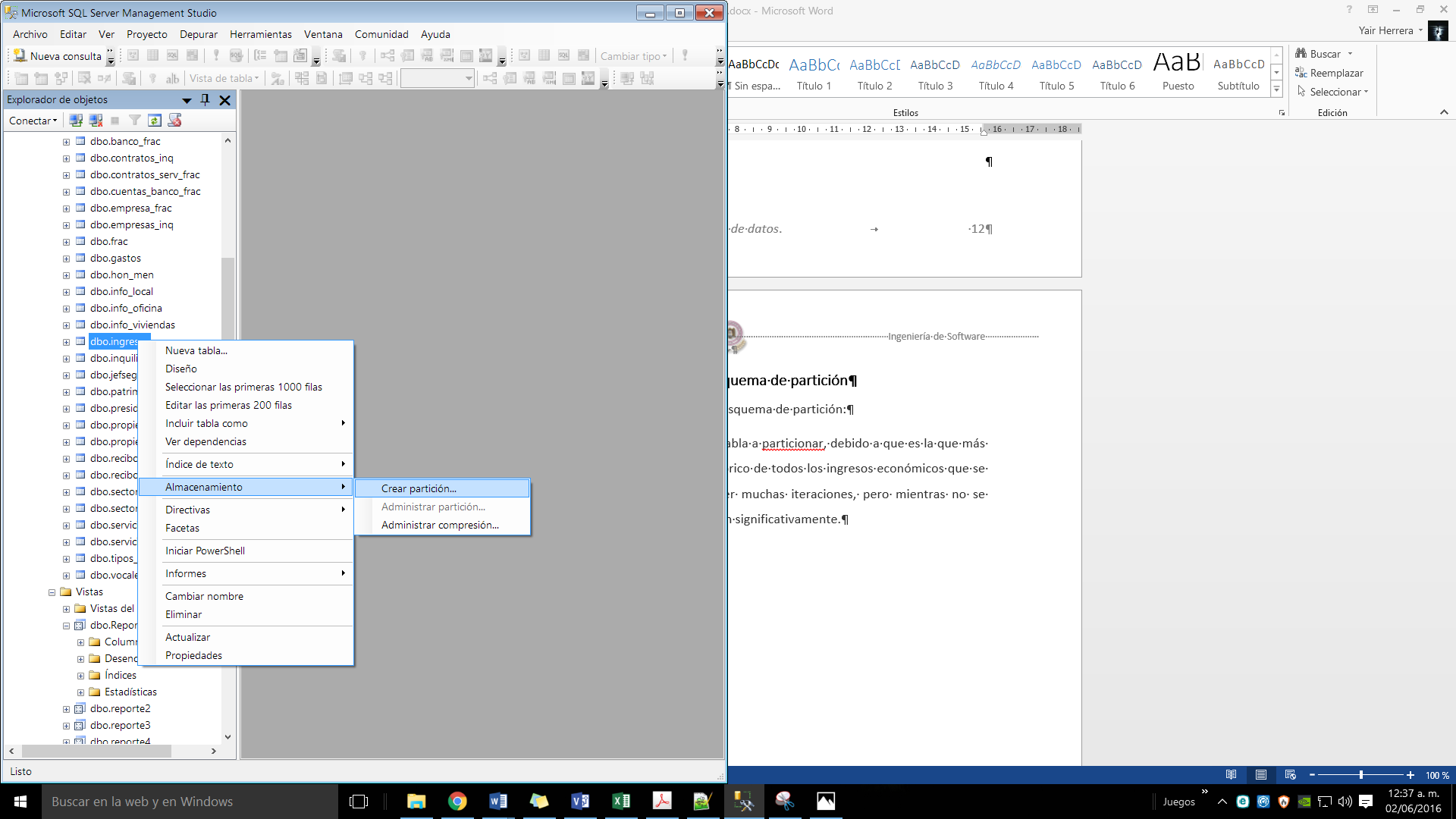


Definición del esquema de partición

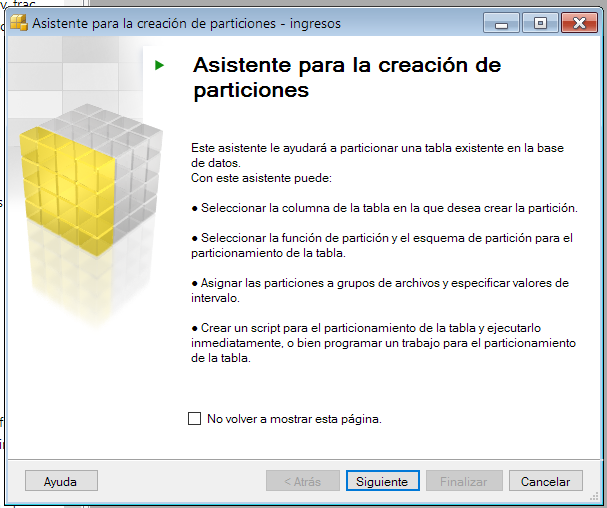
A continuación se muestra la definición del esquema de partición:

Seleccionamos la tabla “**ingresos**” como la tabla a particionar, debido a que es la que más crecería porque sirve como un registro histórico de todos los ingresos económicos que se han administrado. Las demás podrán tener muchas iteraciones, pero mientras no se gestionen más fraccionamientos, no crecerán significativamente.

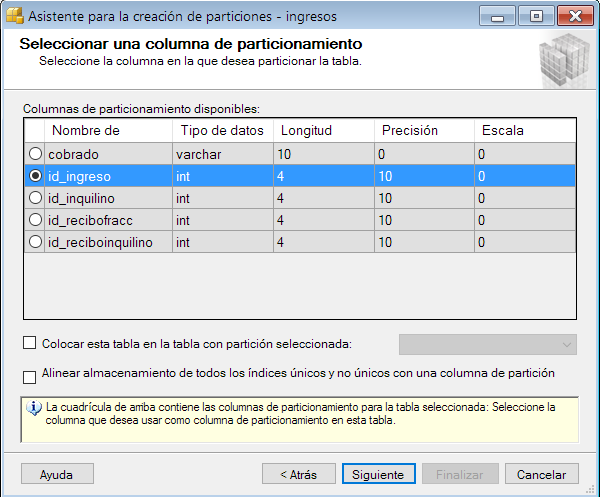
Paso 1: seleccionamos la tabla “ingresos”, botón secundario: Almacenamiento/Crear partición



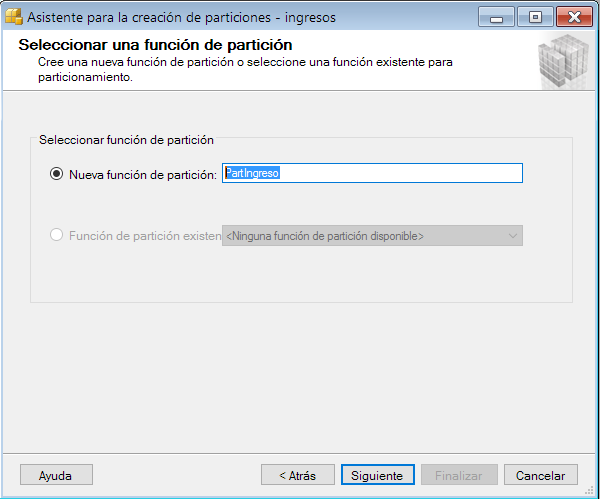
Paso 2: Utilizando el asistente para la creación de particiones, damos click en “siguente:



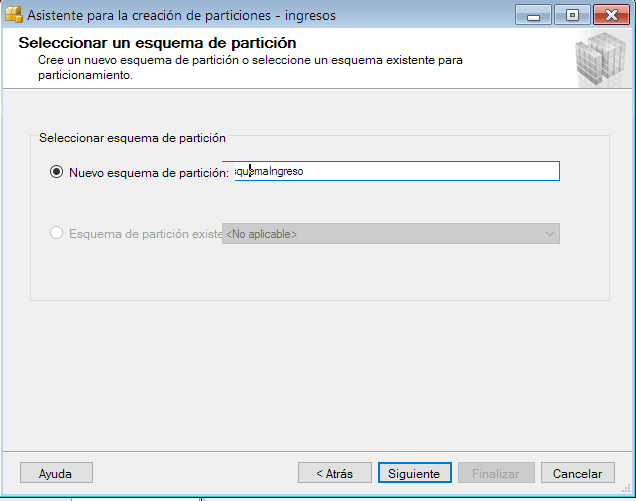
Paso 3: Seleccionamos la “columna de particionamiento”. En este caso, seleccionamos el “id\_ingreso” del tipo “numérico” como es recomendado y damos click en siguiente:



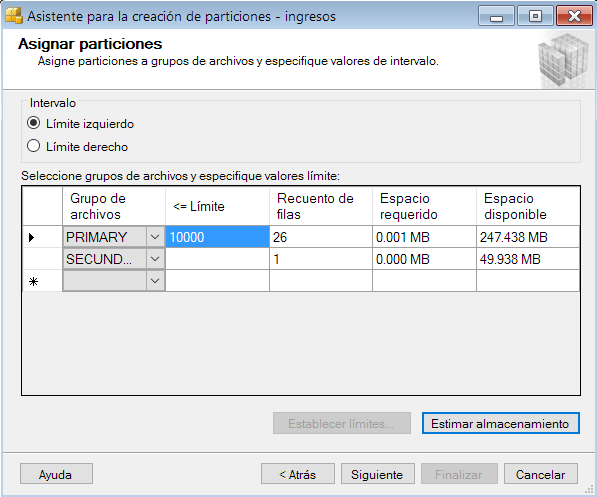
Paso 4: Creamos la “función de particion”, nombrándolo como “PartIngreso”



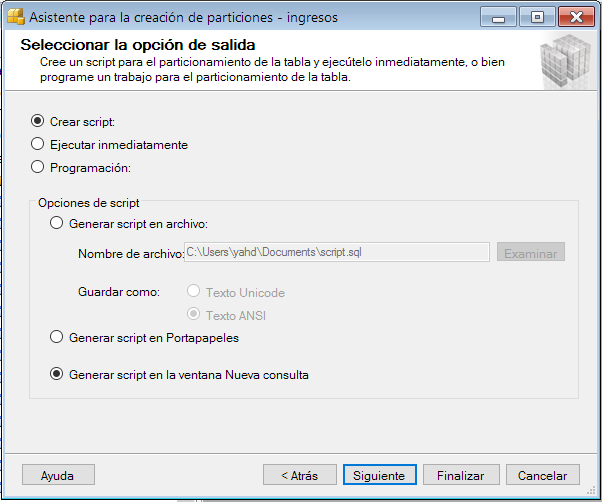
Paso 5: creamos el “esquema de partición” y lo nombramos “EsquemaIngreso”



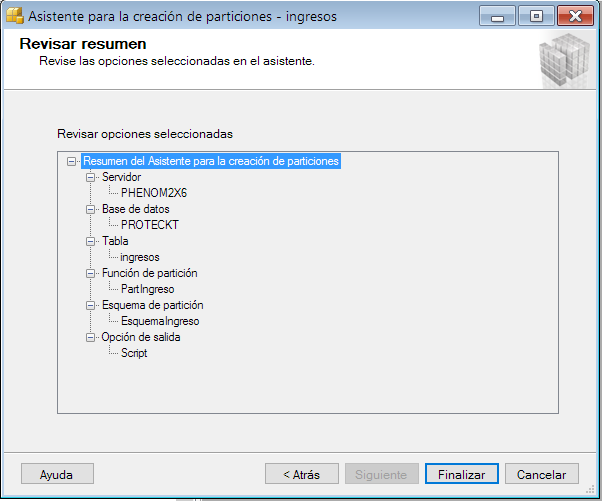
Paso 6 definimos los límites para la partición:. En este caso, en el grupo de archivo primario, guardará todos los registros con id del 1 al 10000. El resto en el Secundario



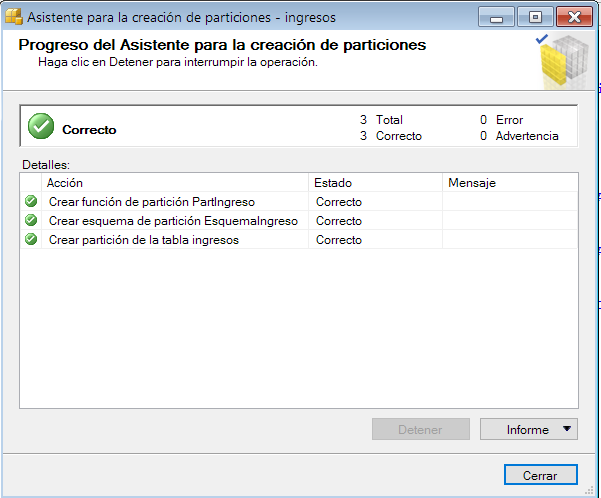
Paso 7: Ejecutamos inmediatamente el Script:



Paso 8: nos da un resumen de movimientos:



Paso 10: si todo está correcto, nos dará un resumen de movimientos realizados con éxito:



Este es el Script de partición:

USE [PROTECKT]

GO

BEGIN TRANSACTION

CREATE PARTITION FUNCTION [PartIngreso](int) AS RANGE LEFT FOR VALUES (N'10000')

CREATE PARTITION SCHEME [EsquemaIngreso] AS PARTITION [PartIngreso] TO ([PRIMARY], [SECUNDARIO])

ALTER TABLE [dbo].[ingresos] DROP CONSTRAINT [PK\_ingresos]

ALTER TABLE [dbo].[ingresos] ADD CONSTRAINT [PK\_ingresos] PRIMARY KEY CLUSTERED

(

[id\_ingreso] ASC

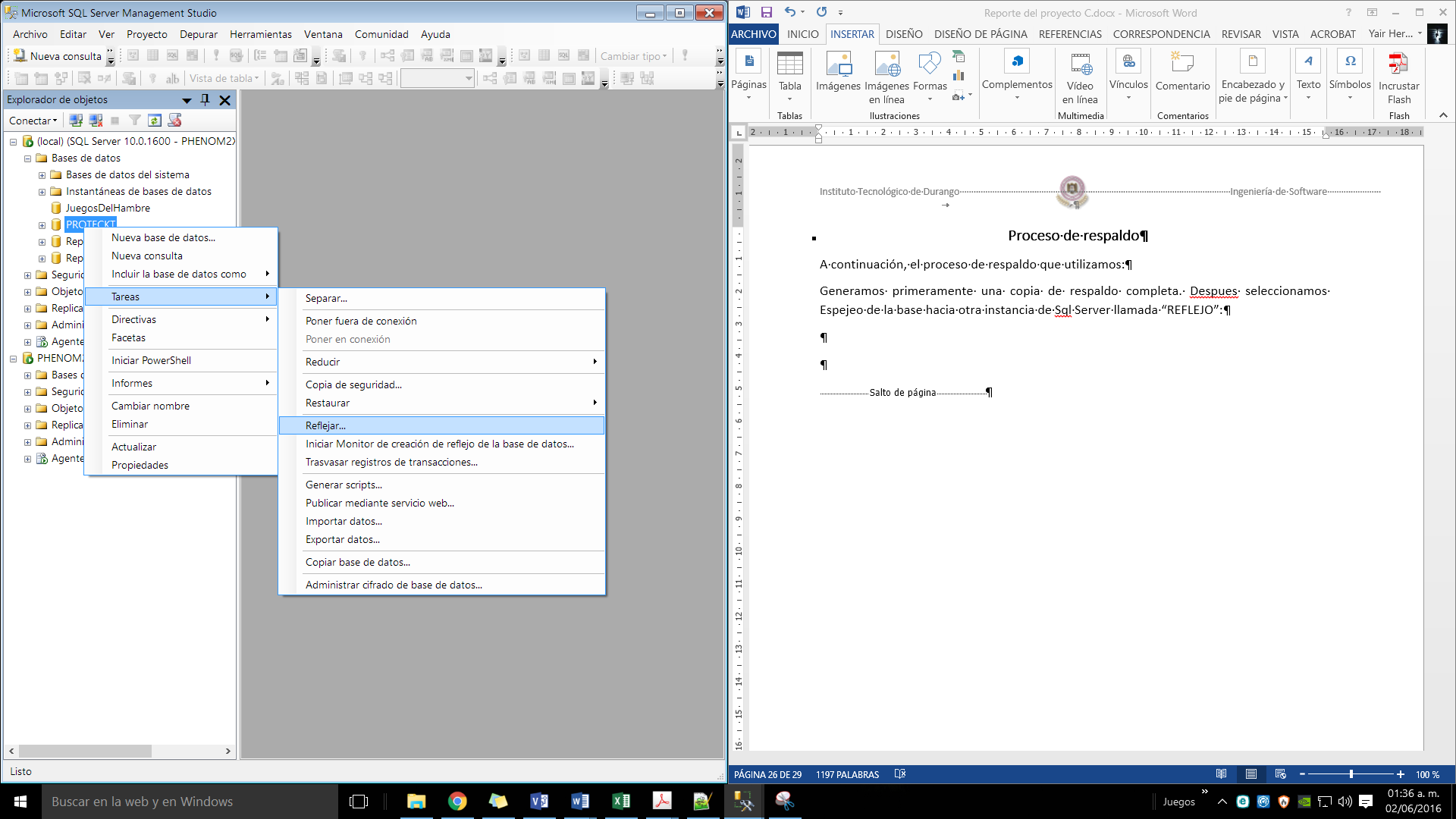
)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, SORT\_IN\_TEMPDB = OFF, IGNORE\_DUP\_KEY = OFF, ONLINE = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [EsquemaIngreso]([id\_ingreso])

COMMIT TRANSACTION

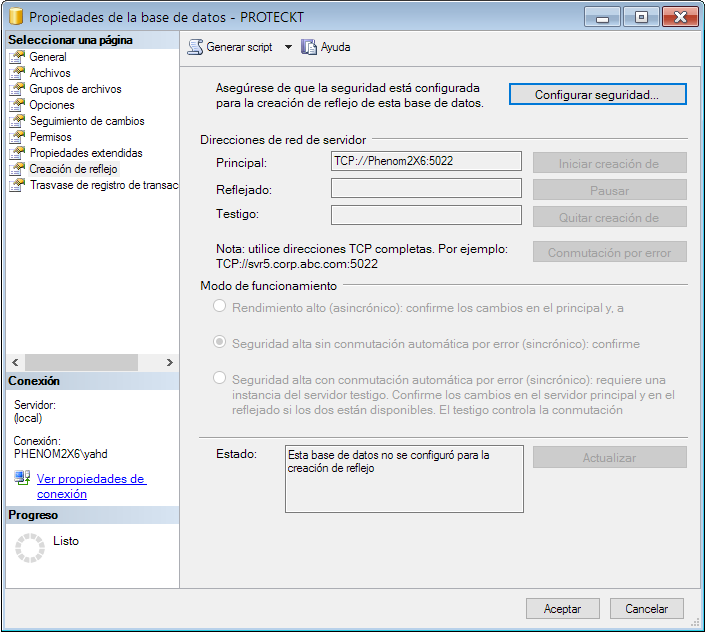
Proceso de respaldo

A continuación, el proceso de respaldo que utilizamos:

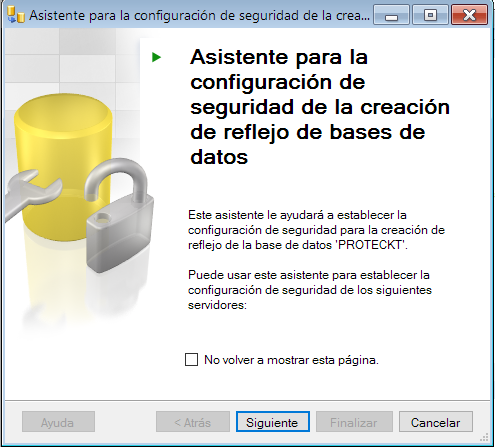
Generamos primeramente una copia de respaldo completa. Después seleccionamos Espejeo de la base hacia otra instancia de Sql Server llamada “REFLEJO”:



En seguida configuramos la seguridad para la creación del reflejo, haciendo clic en el botón correcpondiente:



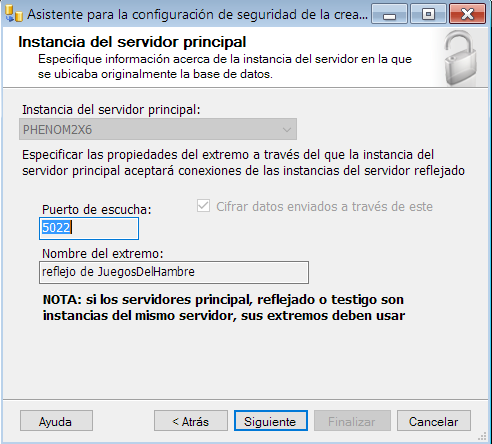
Nos aparece el asistente que nos guiará en todo el proceso de reflejo



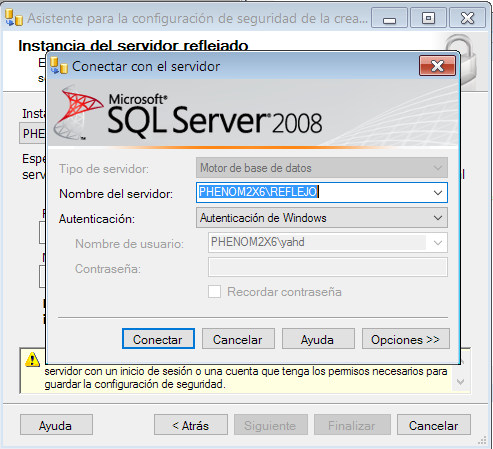
Como no contamos con una 3ra instancia, seleccionamos que no queremos un servidor testigo:



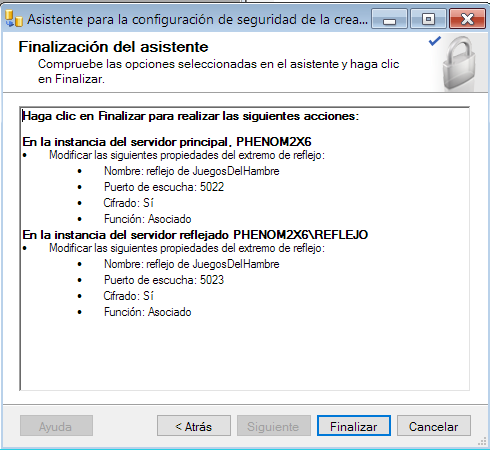
En seguida nos muestra el puerto de escucha que tendrá el reflejo para conectar “el otro exrtremo”



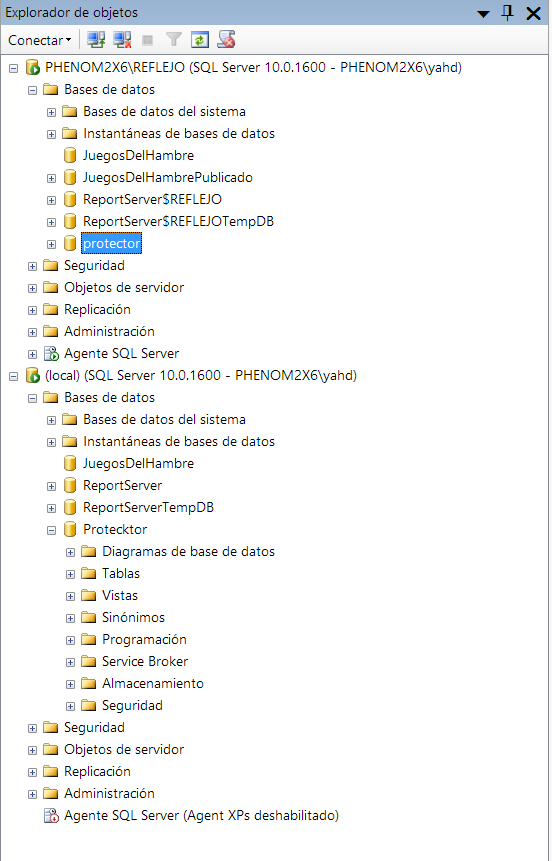
Debemos iniciar sesión en la instancia en donde se reflejará nuestra base de datos



Si la conexión es exitosa, nos muestra un resumen de los procesos que se generarán para la creación del reflejo



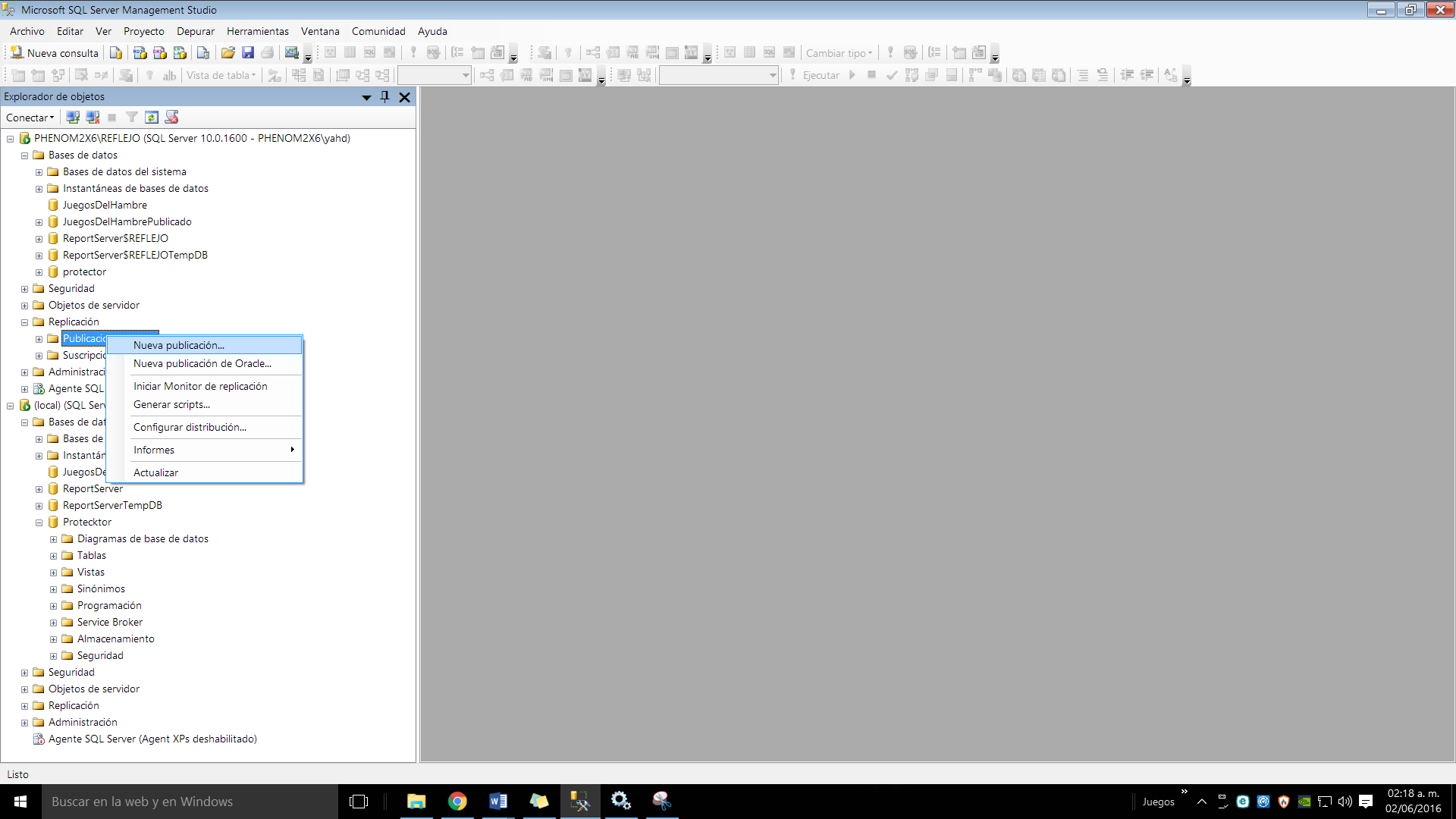
Esta es la muestra de la base de datos reflejada en la instancia “REFLEJO”, cuyo nombre le hemos puesto “protector”



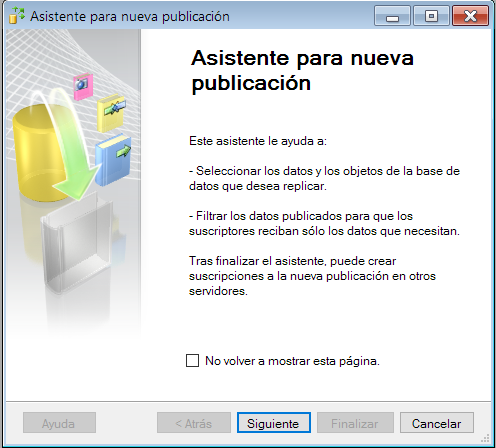
Replicación de datos

A continuación, explicaremos el proceso de la replicación de la base de datos:

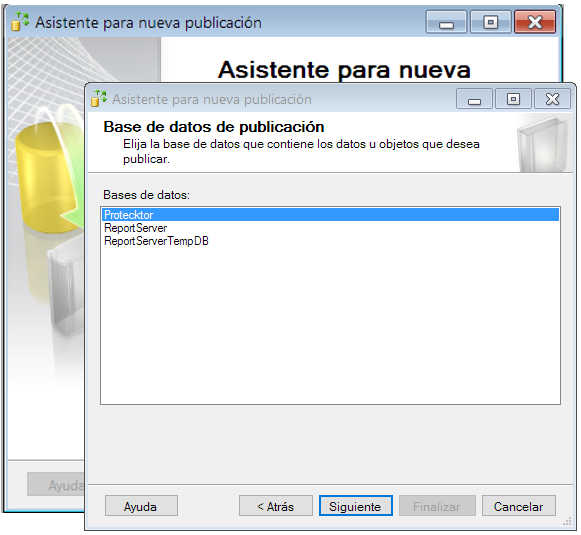
Primero seleccionamos en el servidor que será el publicador, crear “nueva publicación”



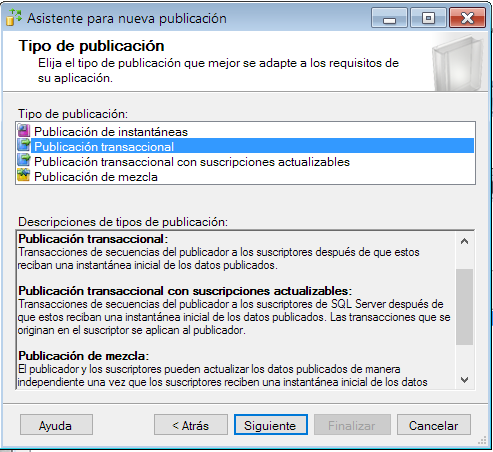
Nos aparee el asistente que nos guiará paso a paso en todo el proceso de publicacion



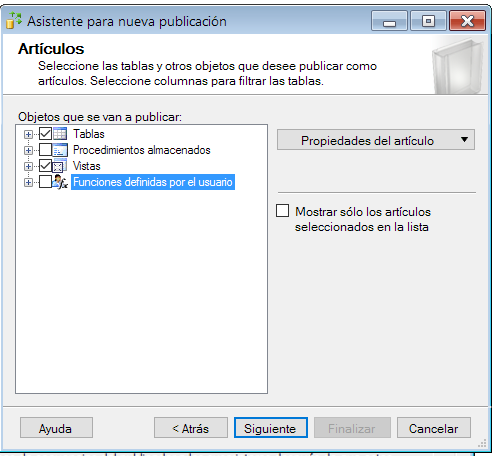
Seleccionamos la base de datos que queremos publicar, en este caso la base de datos “protecktor”, que es una copia de seguridad cargada de la base de datos “PROTECTKC”



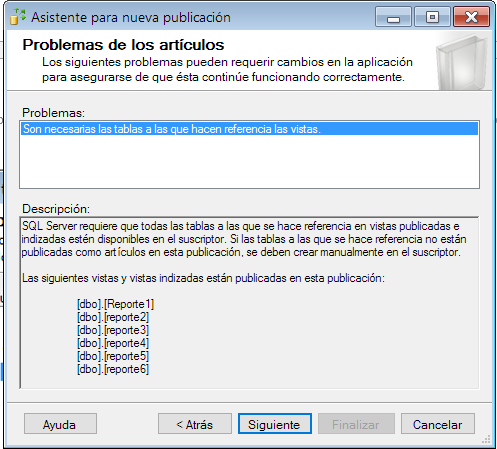
Seleccionamos el tipo de publicación. En esta ocasión, será “publicación transaccional”



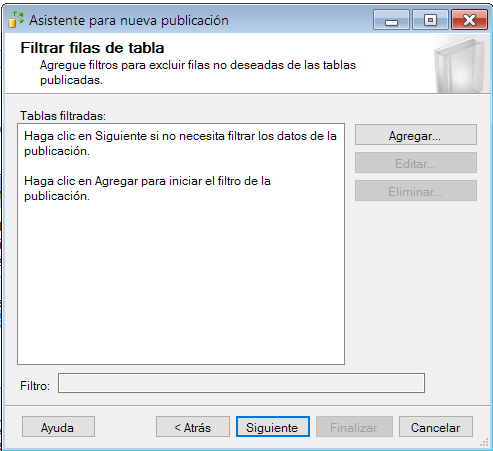
Para esta publicación, solo “publicaremos” las tablas y las vistas a los futuros “suscriptores”



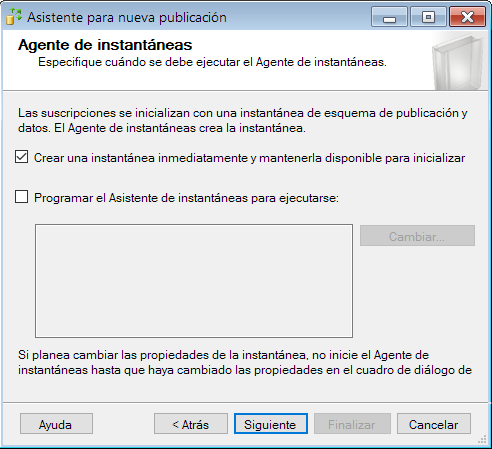
En un rápido analizis se nos informa de las dependencias que algunos elementos puedan tener, como en este caso, las vistas “dependen” de las tablas



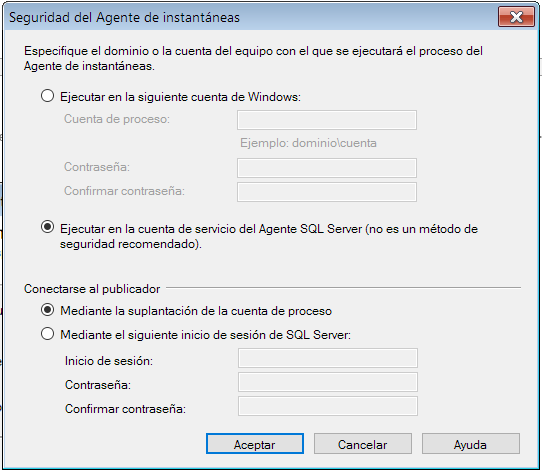
En caso de querer agregar filtros para publicar solo ciertos registros, se daría clic en el boton “agregar”. En esta ocasión, no agregaremos ningún filtro



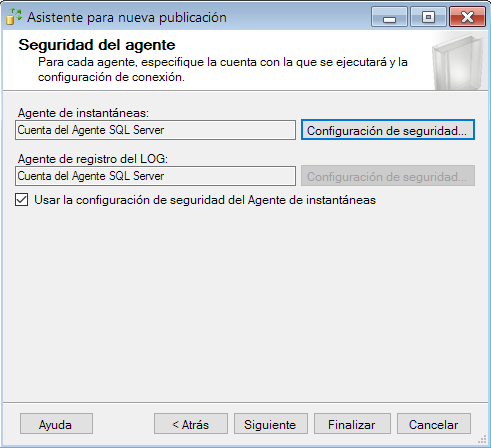
Después nos pregunta si queremos generar una instantánea de nuestra publicación y mantenerla disponible para futuros suscriptores. En esta ocasión, seleccionamos dicha casilla



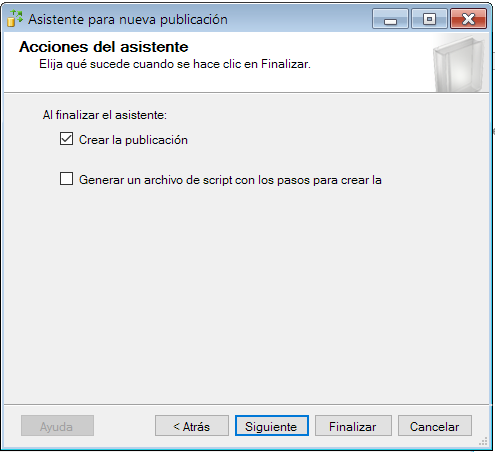
Ahora, nos pregunta sobre el como “conectarse” a la cuenta del equipo con el que se ejecutara el proceso del agente de instantáneas. Seleccionamos las casillas que se muestran en la imagen, aunque no es un método de seguridad recomendado. Solo por motivos didácticos, lo permitiremos



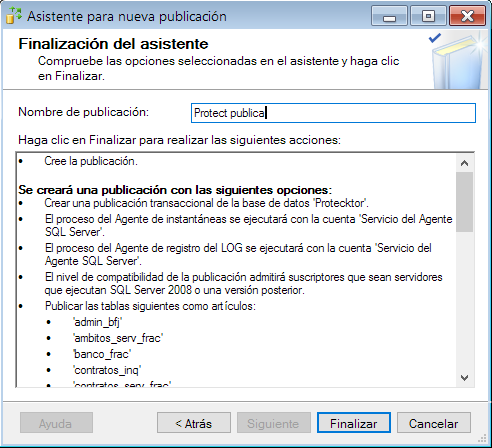
Una vez que tenemos el cómo se conectará tanto el agente de instantáneas como el registro de LOG, podremos continuar



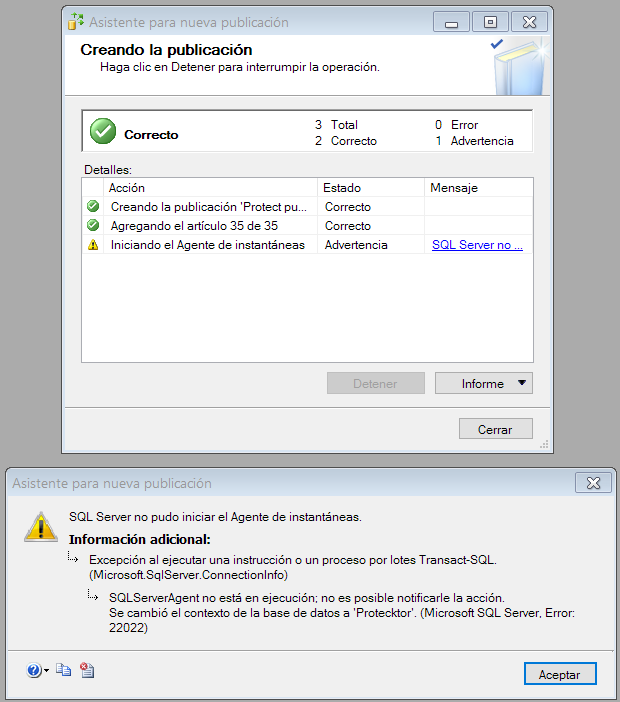
Seleccionamos “crear la publicación”:



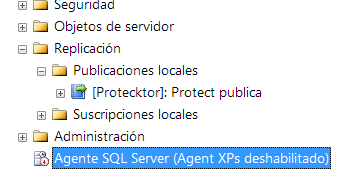
Se nos muestra un resumen de lo que será la publicación

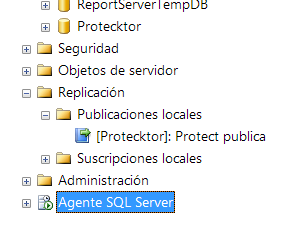


Si todo sale bien, terminará el proceso. En este caso, es necesario iniciar manualmente el “agente de instantáneas”

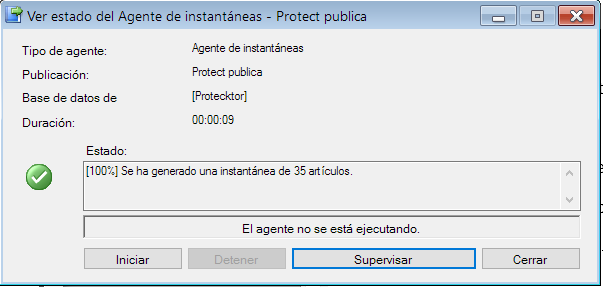
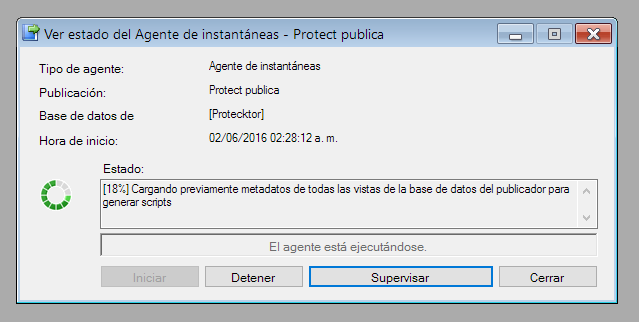


Podemos ver que está inactivo:



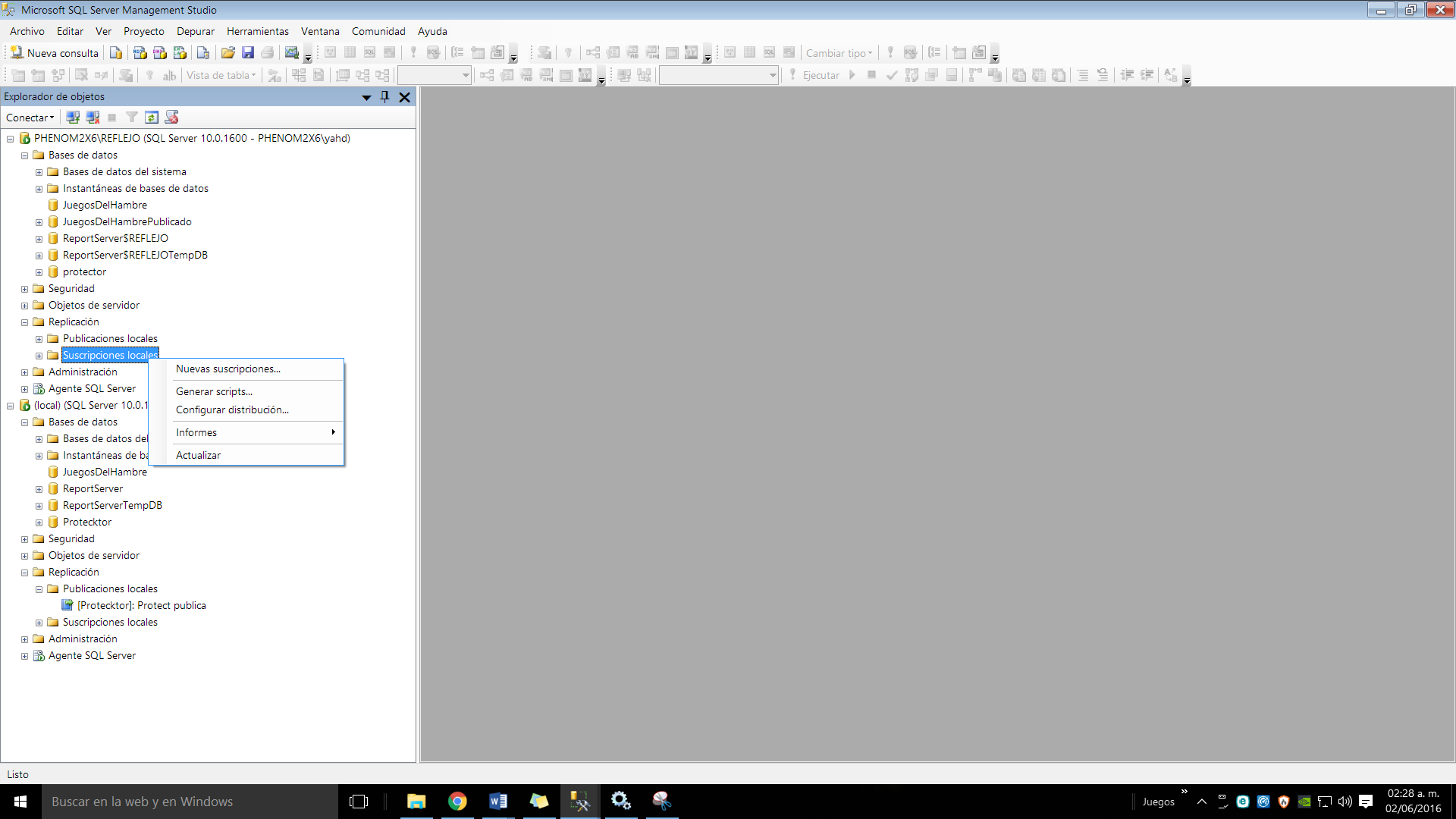


Pasamos a iniciarlo y comenzará a generar la instantanea

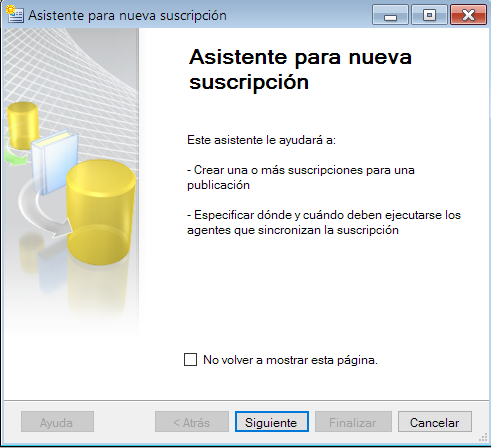


Servidor donde se replicaron los datos

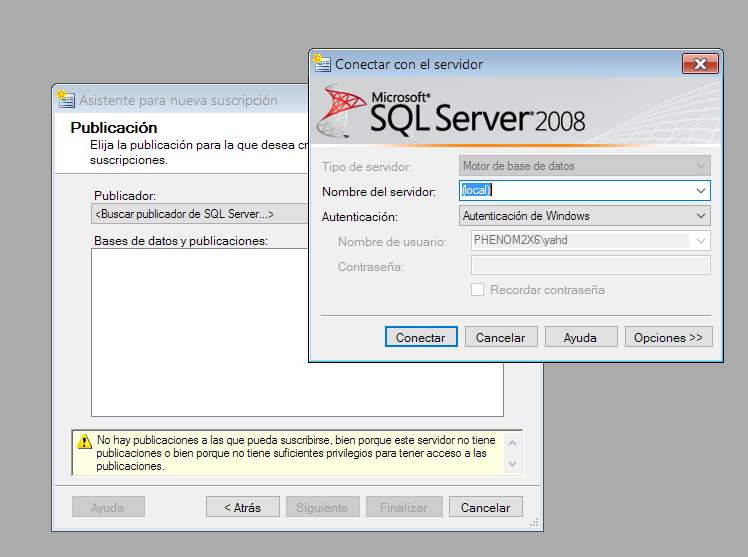
En el servidor donde queramos la replicación de estos datos, por medio de una publicación desde otra instancia, damos a “Nueva suscripción”



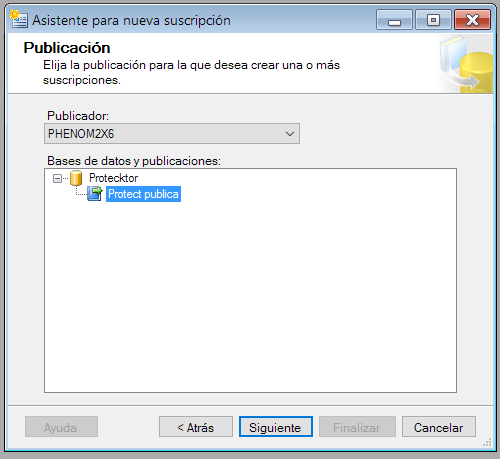
Nos aparece otro asistente que nos guiará en todo el proceso



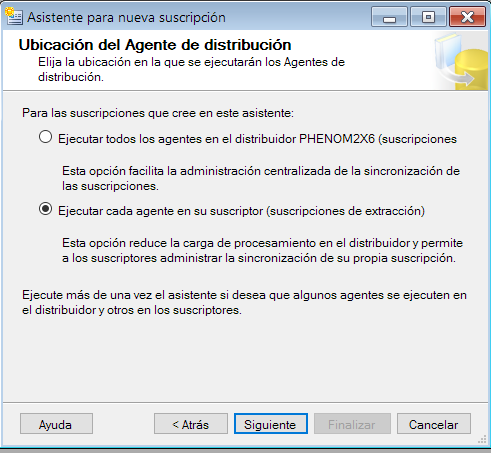
Ahora, buscamos conectarnos a la instancia que tiene la publicación que deseamos



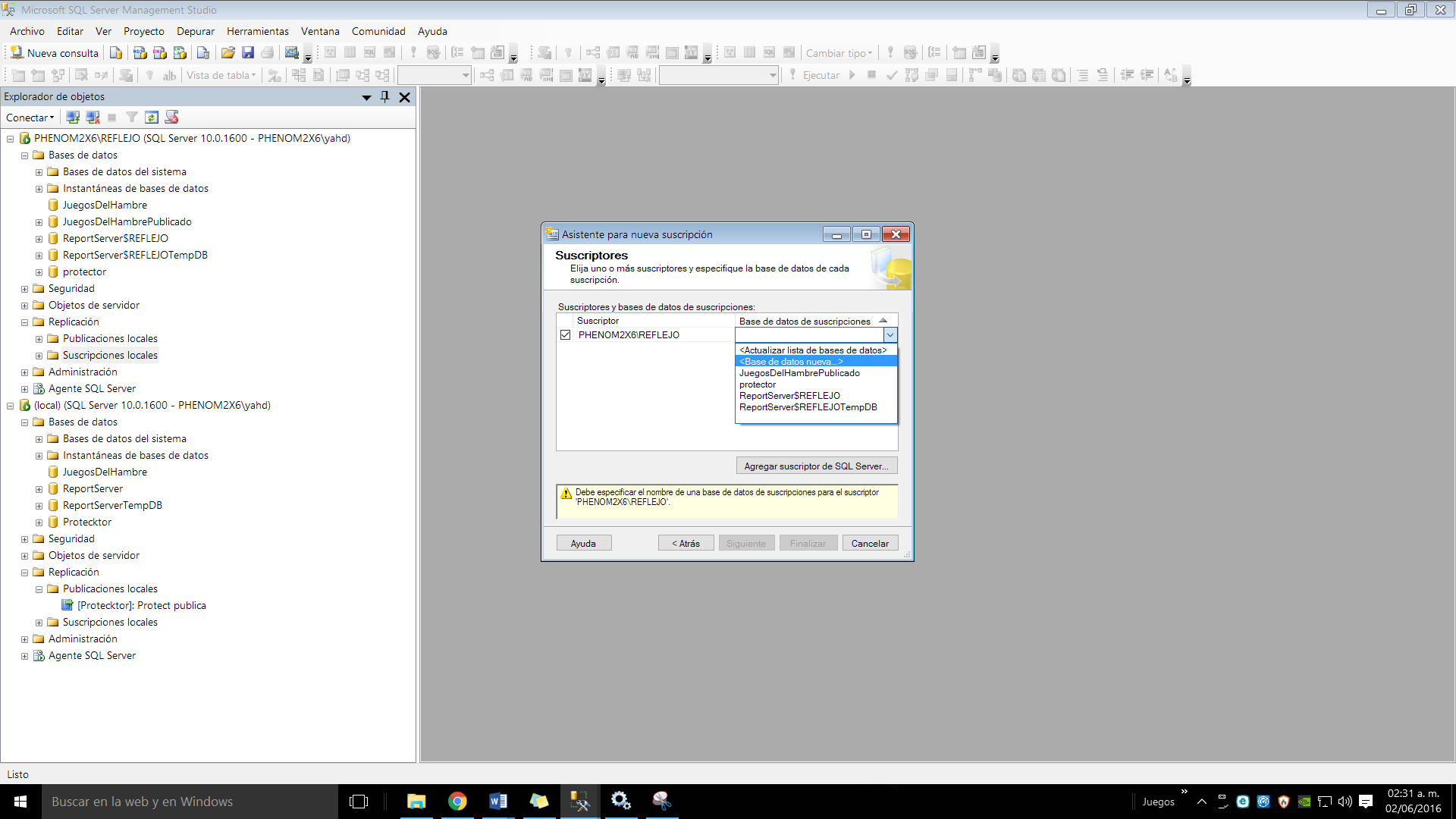
Una vez conectados, nos aparecerá la “publicación” a la cual buscamos “suscribirnos”



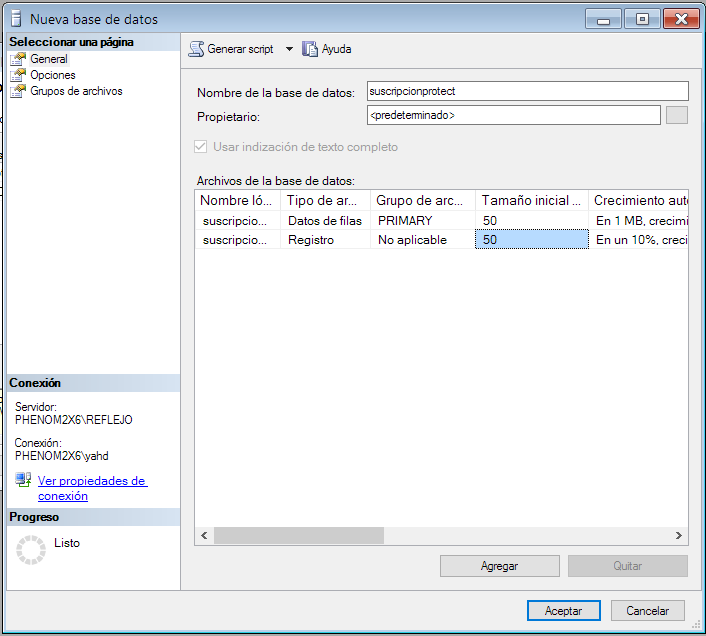
Al seleccionar la publicacion, nos aprecera estas opciones:



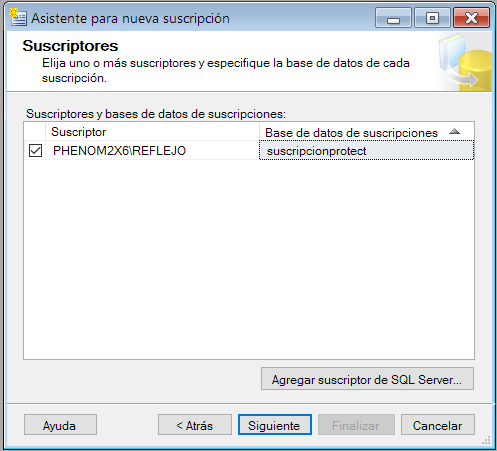
Ahora toca decidir en donde se “cargarán” los elementos que contiene dicha “publicacion” a la que nos estamos suscribiendo



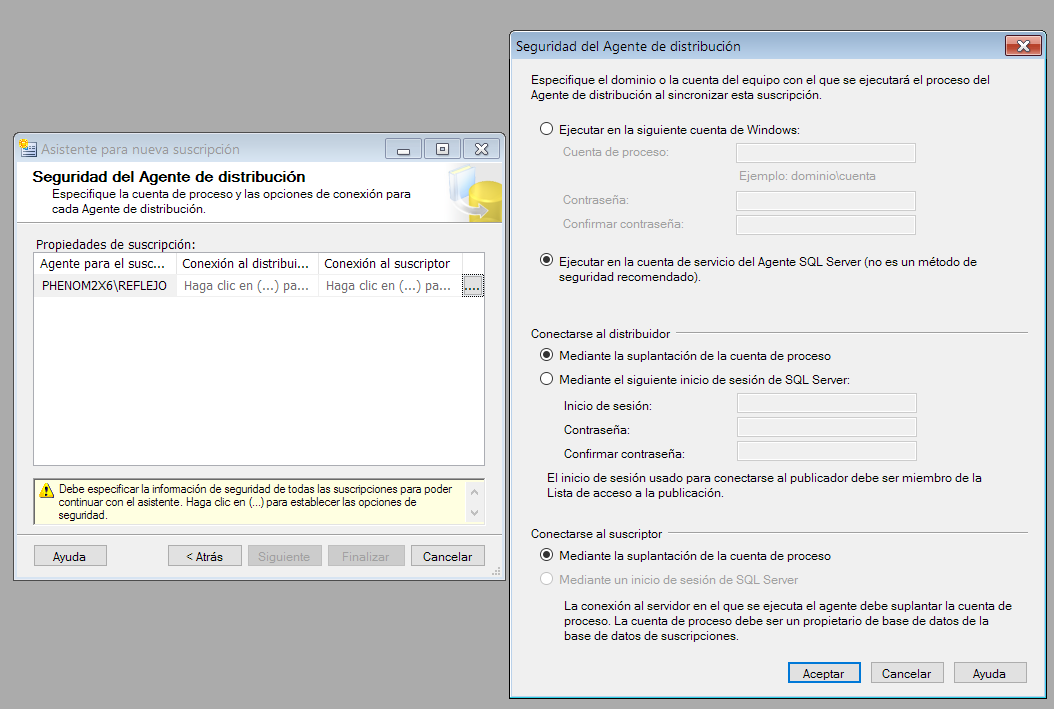
En esta ocasión, optamos por crear una nueva base de datos para contener los elementos que se cargaran al suscribirnos a la publicación que seleccionamos



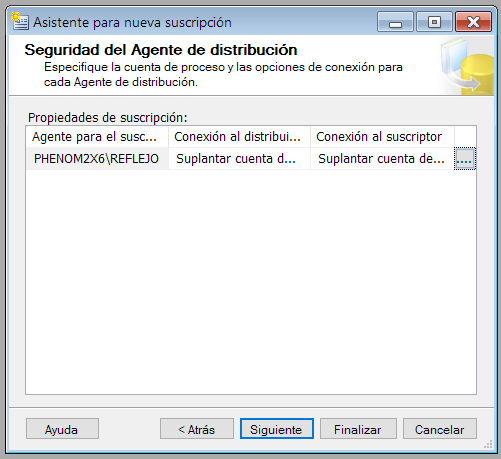
Si todo sale bien, podremos seguir suscribiéndonos a otras publicaciones disponibles. En esta ocasión, solo tenemos una



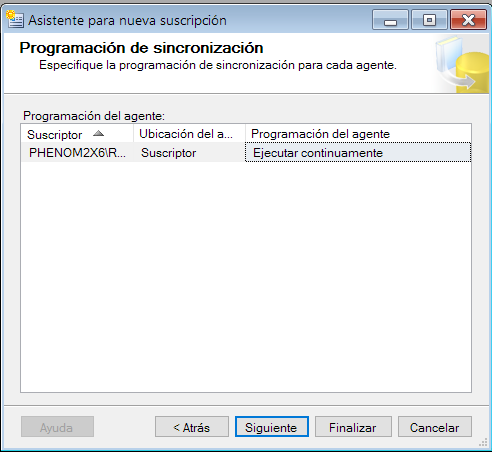
Ahora, configuramos la forma de conectarse a la publicación por medio de una suscripción.



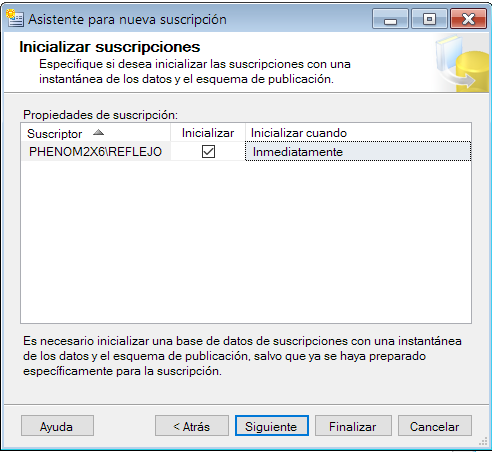
Ya teniendo la forma de conexión que se usará para el “agente de suscripción”, podemos continuar



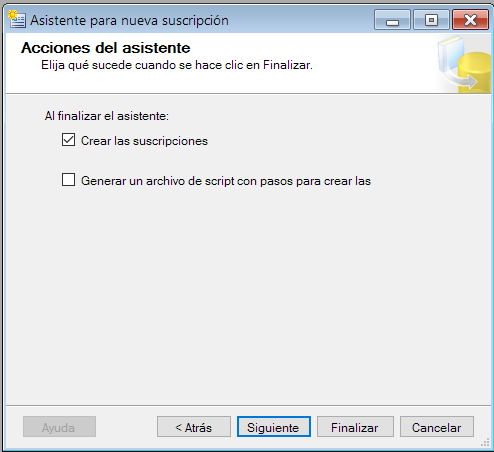
Elegimos el “como” queremos programar la suscripción, ósea, con qué periodo se actualizará



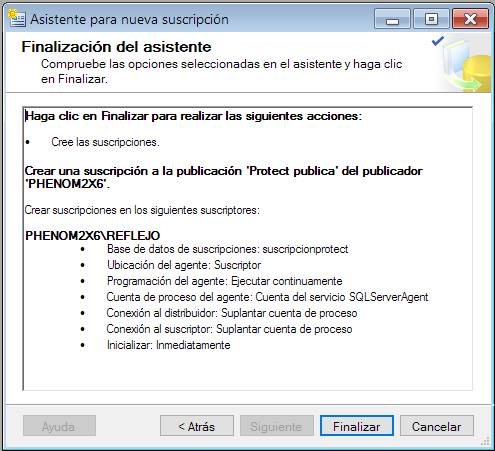
Ahora podemos decidir si la suscripción comenzará inmediatamente



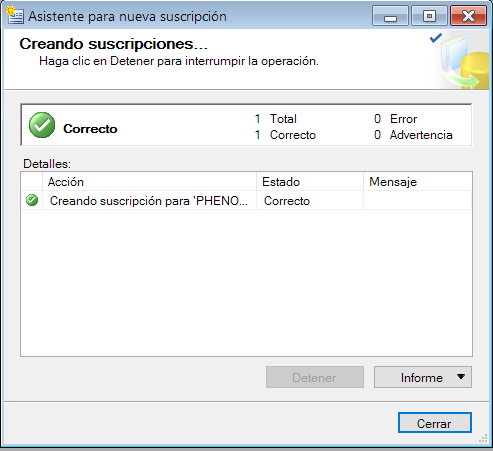
Creamos la suscripción y nos permite también crear un script con todos los pasos



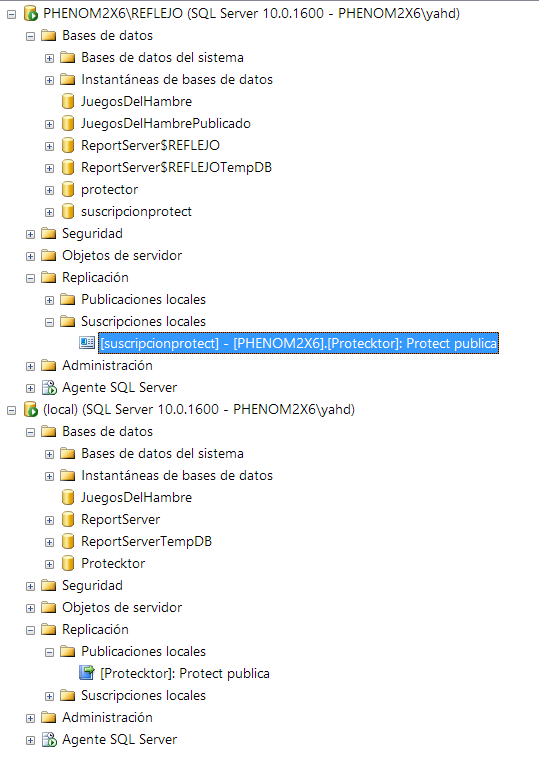
Nos muestra un resumen de los procesos que se llevaran a cabo y listo.



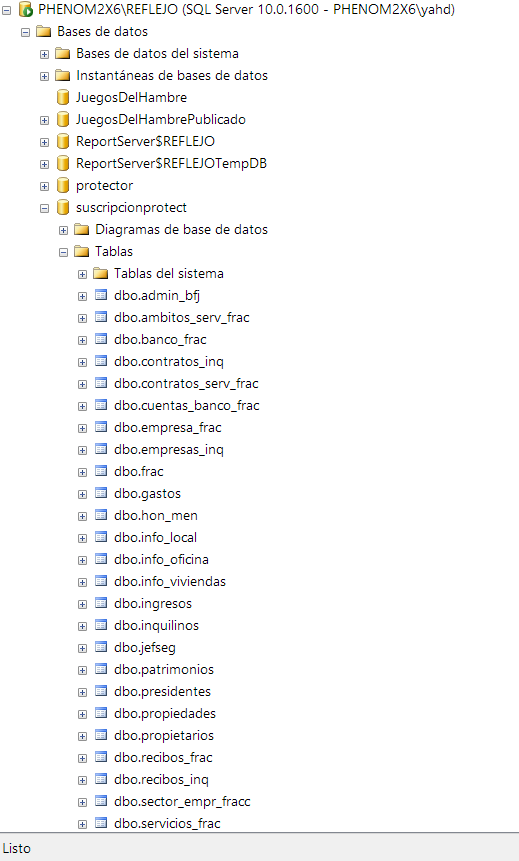
Si todo está correcto, pasará a “suscribirse” a la publicación



Podemos ver que ahora ya témenos una suscripción.



Y podemos ver como ya se han cargado los elementos que representan dicha “suscripción”



Conclusiones

**Yair**: He podido ver la gran importancia de tener un buen manejador, robusto y con todas las capacidades de replicación y espejeo, cosa que me ha gustado mucho y encuentro tanto útil como poco usado en muchísimos lugares que tiene datos críticos solo respaldados en una sola unidad de almacenamiento. Perder datos importantes por desconocer todo esto, es un error que no tiene medida. Ahora me quedo con ciertos retos que me motivan a aprender más de todo este ámbito.

**Aziel**: El SQL Server me ha gustado mucho por su facilidad de configuración. Sin embargo esta versión 2008 al parecer ya presenta ciertas anomalías e incompatibilidades en sistemas operativos nuevos, inclusive con sistemas de seguridad y detección recientes, como antivirus y barreras de fuego. Ahora toca la importancia de manejar sistemas gestores de bases de datos más actualizados, los principios son los mismos, tengo mucha curiosidad por probar el SQL server más reciente.

**Martin**: La verdad me he quedado impresionado con las capacidades que tiene un buen manejador de base de datos. Fue muy grato poder ver en forma práctica todos los puntos que el programa de la materia exigía. A la vez me ha interesado mucho aquellos manejadores “no sql”, ósea los “por objetos”. Deben tener grandes ventajas a costa de ser muy difíciles de configurar. Por otra parte me ha alegrado ver que SQL en verdad tiene un potencial enorme y es bastante similar tanto en manejadores antiguos como en los más nuevos y poderosos.

**Mario**: La importancia del administrador de base de datos, lo pudimos vivir al desarrollar este proyecto. En unos puntos del proyecto, cada quien “normalizo” ciertas entidades y vimos muchísimas circunstancias que nos obligaron a buscar un “lenguaje en común” para ponernos de acuerdo y que pudiéramos juntar todos los diagramas en uno solo. Fue una experiencia bastante tediosa pero muy productiva. Podríamos decir que trabajamos como si fuéramos primero “varios administradores de base de datos”, hasta que pudimos llegar a pensar como uno solo.

Observaciones durante el desarrollo del proyecto

1.- El sql server 2008 no soporta ser instalado en carpetas con el atributo avanzado “carpeta lista para archivarse” activado, debido a que este atributo es una forma de “compresión” de datos. Para corregir esto, se debe de quitar ese atributo en la carpeta especificada de instalación.

2.- Se tuvo que deshabilitar temporalmente la siguiente opción: *impedir guardar cambios que requieren la recreación de la tabla* para el alta y la edición de las tablas en el momento de la creación del esquema de la base de datos:

Para cambiar la opción *“impedir guardar cambios que requieren la creación de la tabla”,* seguimos estos pasos:

*Abra SQL Server Management Studio (SSMS).*

*En el menú Herramientas, haga clic en Opciones.*

*En el panel de navegación de la ventana de Opciones, haga clic en los diseñadores.*

*Seleccione o desactive la casilla de verificación impedir guardar cambios que requieren la recreación de la tabla y, a continuación, haga clic en Aceptar.*

Al final de la creación y modificaciones de las tablas en el proceso de diseño, volvimos a activar la opción por seguridad.

3.- al dar el siguiente error al restaurar una copia de la base de datos:

*“El sistema operativo devolvió el error '32(failed to retrieve text for this error. Reason: 15105)* *' al intentar RestoreContainer::ValidateTargetForCreation' en 'C:\Program Files\Microsoft SQL Server\MSSQL10.MSSQLSERVER\MSSQL\DATA\JuegosDelHambre.mdf'. (Microsoft.SqlServer.Smo)”*

Al parecer el error aparece por que el archivo que se intenta restaurar (la copia de seguridad .bak) está en uso por otro programa o en este caso, por otra instancia de sql server. Se solucionó al “detener” el servidor “local” del que se generó la copia de la base de datos.

Para nosotros, este detalle de necesitar “detener” el servidor “local” (que es nuestro servidor principal), para que se pueda recuperar un archivo de “copia de respaldo” .bak en otro servidor, es una especie de error porque no tiene sentido de uso eficiente. Buscaremos que es exactamente lo que estaba usando el archivo .bak en el servidor local.