Version Control System for Gis (VCSGis) Documentación de usuario

Borrador preliminar

(v1.0.4)

Sumario

Control de cambios

Versión	fecha	responsable	organización	estado	descripción
1.0.0	07/12/2020	Jose Olivas	Asociación gvSIG	Primera revisión	Borrador preliminar
1.0.1	21/01/2021	Jose Olivas	Asociación gvSIG		
1.0.2	17/02/2021	Jose Olivas	Asociación gvSIG		
1.0.3	23/02/2021	Jose Olivas	Asociación gvSIG		
1.0.4	23/03/2021	Jose Olivas	Asociación gvSIG		

Introducción

Este documento presenta una introducción resumida al control de versiones **VCSGis** o **Version Control System Gis**. En el se tratan los conceptos generales de un sistema de control de versiones genérico y estos mismos en el caso concreto de *VCSGis*. Además el documento pretende que tras su lectura el usuario no solo conozca las partes o
elementos que componen este tipo de sistemas sino que pueda utilizarlo o trabajar con *VCSGis*. Para llevar a cabo la segunda idea esta documentación presenta una guía de uso más ejemplos básicos.

Antes de empezar a describir las partes que componen un sistema de control de versiones hay que definir claramente que son estos y por tanto que es *VCSGis*. Un sistema de control de versiones es un programa o software basado en la centralización de informacion para compartir entre usuarios que a diferencia de un servidor normal, recuerda los cambios que hayan sido realizados en sus datos. Cabe destacar de nuevo que no es un servidor al uso, ya que no solo almacena información, sino que almacena la información así como las modificaciones que los usuarios realizan sobre ella, siendo igual de importante las gestión de esos cambios como la propia información.

También hay que destacar que los sistemas de control de versiones no se centran en ningún tipo de dato en concreto, pudiendo existir sistemas que controlan archivos fuente, imágenes… dependiendo de la necesidad de los usuarios. En el caso de VCSGis al estar orientado a un perfil técnico especialidado en informacion geografica. Este gestiona información sobre tablas y capas (tablas con información geométrica).

Sin mas dilación comenzamos a detallar las diferentes partes que componen un sistema de control de versiones tomando como ejemplo a *Version Control System Gis* (*VCSGis*).

Empezando

Instalación de VCSGis

TODO

Pendiente de realizar documentacion

Conceptos básicos del control de versiones

Antes de que entremos a ver como funciona *VCSGis* es importante tener una visión general de cómo funciona un *Sistema de Control de Version* (*VCS*) y los términos que se utilizan.

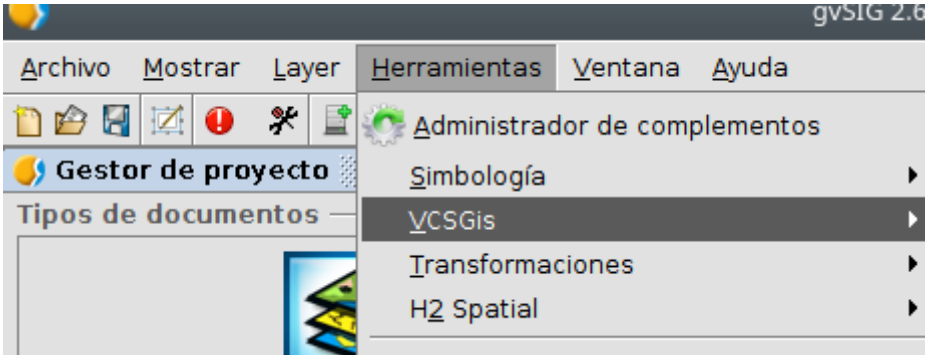
- El **repositorio** VCSGis usa una base de datos central que contiene todos los datos cuyas versiones se controlan y sus respectivas historias. Ésta base de datos se conoce como el **repositorio**. El repositorio normalmente esta en un servidor de base de datos, que provee a pedido el contenido a los clientes de *VCSGis*. Si solo puede hacer una copia de seguridad de una sola cosa, hágala del repositorio, ya que es la copia maestra de toda su información.

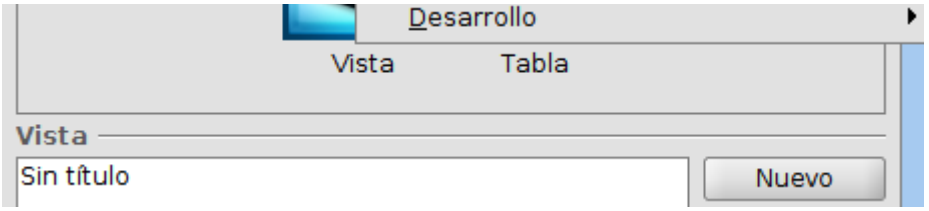
- **Copia de trabajo**

  Aquí es donde se realiza el trabajo en serio. Cada usuario tiene su propia copia de trabajo, comunemente conocida como *caja de arena* en su ordenador local. Usted puede obtener la última versión del *repositorio*, trabajar en ella localmente sin perjudicar a nadie, y cuando haya terminado con los cambios que ha realizado puede *confirmar* (commit) sus cambios en el *repositorio*.

  Una *copia de trabajo* de *VCSGis* no contiene la historia de los datos, pero sí contiene una copia de los datos que existían en el repositorio antes que comience a hacer cambios. Esto significa que es fácil verificar qué cambios ha realizado.

También necesita saber donde encontrar *VCSGis* dado que no hay mucho para ver en los menus y herramientas de *gvSIG desktop*. Esto se debe a que *VCSGis* es una complemento de *gvSIG desktop*, así que primero inicie el *gvSIG dektop*. Hága click en el menu *"Herramientas"* y debería ver una entrada nueva **VCSGis**:




Desarrollo
Vista
Tabla
Vista
Sin título
Nuevo

Repositorio

Como se dijo en el apartado anterior *VCSGis* es un sistema centralizado para compartir información, estando en su núcleo un **repositorio**, es decir un almacén central de datos, que salva la información en forma de tablas. Este almacén no es más que un sistema de base de datos que tiene tanto la información que modifican los usuarios del software, en el caso de *VCSGis* tablas o capas, como las tablas que almacenan los cambios producidos en los anteriores.

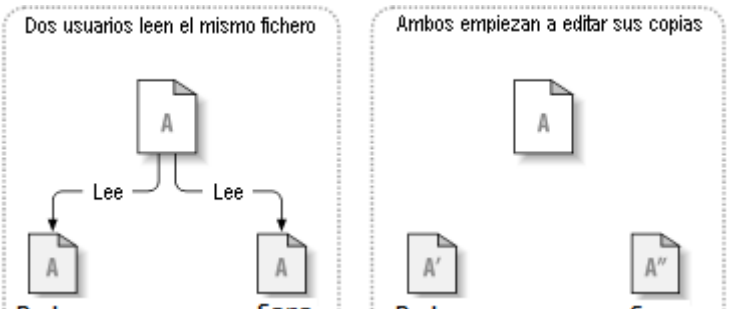
Cada tabla del usuario mantendra su propio control de versiones de forma independiente a las demas tablas del repositorio, llevando un control de los registros que se van insertando, modificando o borrando en ella. La unidad minima de informacion que sigue o controla *VCSGis* es el registro de una tabla, sin hacer seguimiento de los cambios en los campos de cada registro.

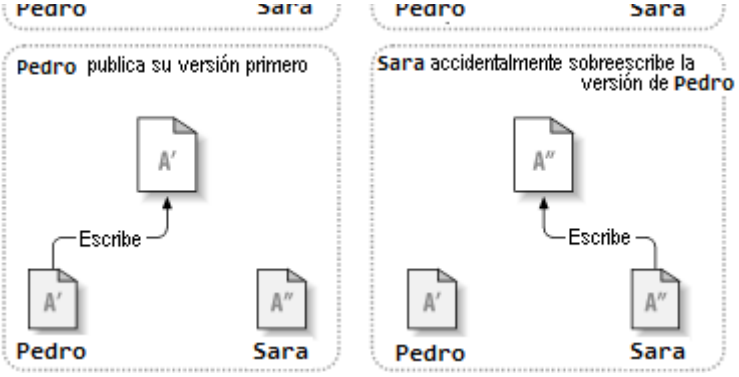
Cuando un cliente accede a los datos del repositorio en *VCSGis*, normalmente ve la última versión de este, es decir la informacion actual con las capas y tablas . Sin embargo el cliente también tiene la posibilidad de ver los estados previos del repositorio. Por ejemplo, un cliente podría preguntarse *¿Qué contenía esta tabla este miércoles?* o *¿Quién fue la última persona en editar esta capa y qué cambios realizó?* Estas son el tipo de posibilidades que diferencian a los sistemas de control de versiones: sistemas diseñados para guardar y registrar las modificaciones de los datos a lo largo del tiempo.

Modelos de versionado

Tras la definición de los sistemas de control de versiones y más concretamente *VCSGis*, así como el núcleo de estos o repositorio, puede intuirse la existencia de un punto crítico en este tipo de software. Este punto no es otro que el intercambio de información actualizada entre usuarios, evitando que los cambios de estos no se pisen unos a otros, ya que parece sencillo pensar que los usuarios de forma accidental pueden sobrescribir cambios de los demás al almacenar la información de los suyos en el repositorio.

Para entender mejor lo anterior consideramos el siguiente escenario. Suponga que tiene dos compañeros de trabajo, Pedro y Sara. Cada uno decide editar la misma tabla del repositorio a la vez. Si Pedro guarda sus cambios en el repositorio primero, es posible que, unos momentos después, Sara pueda accidentalmente sobreescribirlos con su propia versión. Mientras que la versión de la tabla de Pedro no se ha perdido para siempre, porque el sistema recuerda cada cambio, cualquier cambio que Pedro hizo no estará en la versión nueva de la tabla de Sara, porque ella nunca vio los cambios de Pedro. De lo anterior podemos decir que el trabajo de Pedro está efectivamente perdido, o al menos falta en la última versión de la tabla.



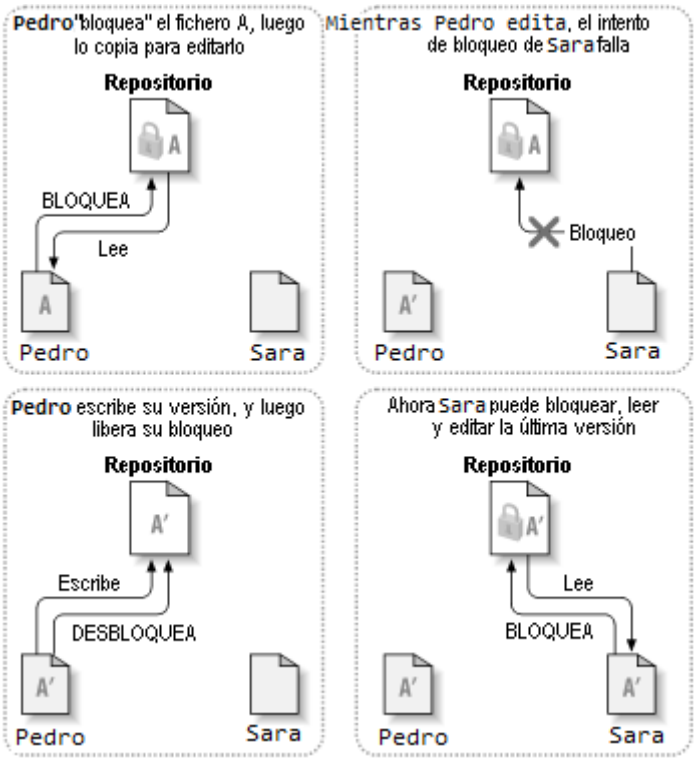


Para solucionar ese problema existen dos modelos;

- Bloquear-Modificar-Desbloquear
- Copiar-Modificar-Fusionar

Bloquear-Modificar-Bloquear

Muchos sistemas de control de versiones utilizan un modelo Bloquear-Modificar-Desbloquear para enfrentarse al problema anterior, lo cual es una solución muy simple. En estos sistemas, el repositorio sólo permite que una persona cambie un archivo. Pedro primero debe bloquear el archivo antes que pueda empezar a hacer cambios en él. Si Pedro ha bloqueado un archivo, entonces Sara no puede hacer ningún cambio en él. Si ella intenta bloquear el archivo, el repositorio le denegará la petición. Todo lo que ella puede hacer es leer el archivo, y esperar a que Pedro termine sus cambios y libere el bloqueo. Después que Pedro desbloquee el archivo es el turno de Sara para bloquear y editar.



El problema con el modelo bloquear-modificar-desbloquear es que es un poco restrictivo, y a menudo se convierte en una molestia para los usuarios:

- *El bloqueo causa muchos problemas administrativos*. A veces los usuarios bloquean archivos y se olvidan de desbloquearlos, dejando sin acceso a la

edición de estos por parte de los demás usuarios. Para solucionar esto se tiene que gestionar el desbloqueo por parte de un administrador lo cual hace que la situación cause un montón de retraso y pérdidas de tiempo innecesarias.

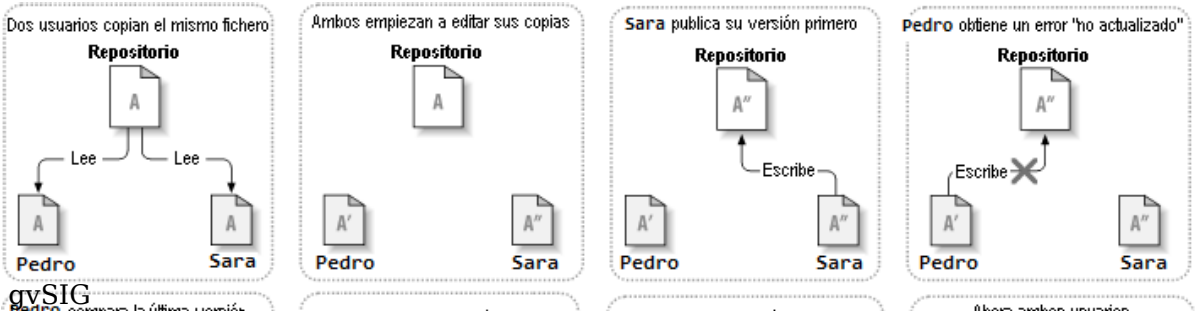
- *El bloqueo puede causar procesos en serie innecesarios*. ¿Qué ocurre si un usuario está editando el inicio de un archivo de texto, y otro simplemente quiere cambiar la parte final del mismo archivo? Esos cambios no se superponen en absoluto y por tanto se podría editar el archivo de forma simultánea, no existiendo la necesidad de tomar turnos en este tipo de situaciones.

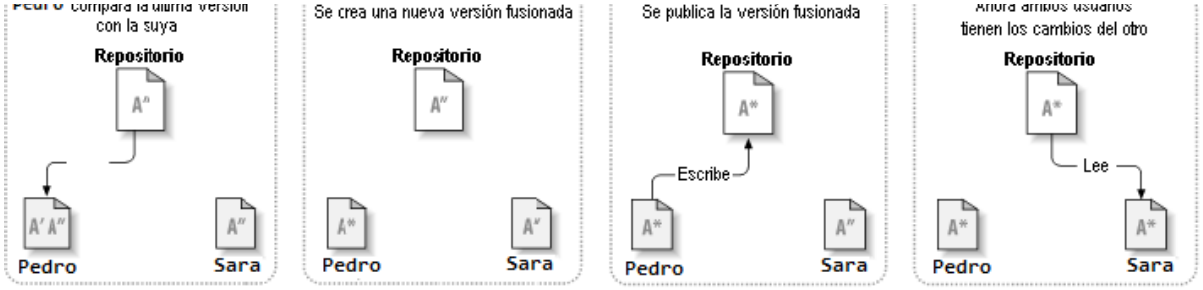
- *El bloqueo puede causar una falsa sensación de seguridad*. Imagine que un usuario bloquea y edita el archivo A, mientras otro usuario distinto simultáneamente bloquea y edita el archivo B. Pero supongamos también que los archivos A y B dependen uno del otro, y que los cambios hechos a cada uno son incompatibles. De repente A y B ya no funcionan juntos y el sistema de bloqueo no tiene forma de prevenir este problema, sin embargo, de alguna forma este sistema dio una sensación de falsa seguridad. Es fácil para los usuarios imaginar que al bloquear los archivos están realizando una tarea segura y aislada, y la realidad es que no, inhibiendoles de discutir si sus cambios son compatibles entre si.

Copiar-Modificar-Fusionar

VCSGis y otros sistemas de control de versiones usan el modelo Copia-Modificación-Fusión como alternativa al modelo de bloqueo. En este modelo, cada cliente de los usuarios lee el repositorio y crea una **copia de trabajo** personal de las capas o tablas de modo que los usuarios trabajan en paralelo, modificando sus copias privadas. Finalmente, las copias privadas son unificadas conjuntamente en una nueva versión final mediante el asesoramiento del sistema de control de versiones, pero es un humano en última instancia el responsable de hacer esta acción de manera correcta.

Para entender mejor lo anterior consideramos el siguiente escenario; Digamos que tanto Pedro como Sara crean copias de trabajo de la misma capa, copiadas del repositorio. Ellos trabajan simultáneamente, y hacen cambios al mismo archivo A dentro de sus copias. Sara es la primera en grabar sus cambios en el repositorio. Cuando Pedro intenta grabar sus cambios más tarde, el repositorio le informa que su archivo A está desactualizado. En otras palabras, que el archivo A en el repositorio ha cambiado de alguna forma desde la última vez que lo copió. Por lo que Pedro le pide a su cliente que fusione cualquier nuevo cambio del repositorio dentro de su copia de trabajo del archivo A. Lo más seguro es que los cambios de Sara no se superpongan a los suyos; por lo que una vez que ambos conjuntos de cambios se han integrado, él graba su copia de trabajo de nuevo en el repositorio.





¿Pero qué ocurre si los cambios de Sara se superponen a los cambios de Pedro? ¿Qué hacemos entonces? La situación se denomina un **conflicto**, y normalmente no es problema. Cuando Pedro le pide a su cliente que fusione los últimos cambios del repositorio en su copia de trabajo A, se da el caso anterior, la copia de trabajo de Pedro marca que está en un estado de conflicto. En ese estado de conflicto Pedro es capaz de ver ambos conjuntos de cambios conflictivos, y manualmente podrá elegir entre ellos cual es el más acertado. Tenga en cuenta que el software no puede resolver conflictos automáticamente, ya que sólo los humanos son capaces de entender y hacer las elecciones necesarias de forma correcta. Una vez que Pedro haya resuelto manualmente los cambios que se superponían, puede guardar de forma segura el archivo fusionado al repositorio.

El modelo Copiar-Modificar-Fusionar puede parecer un poco caótico, pero en la práctica, funciona extremadamente bien. Los usuarios pueden trabajar en paralelo, sin que tengan que esperar nunca uno por otro, y cuando trabajan en los mismos archivos, resulta que la mayoría de los cambios concurrentes no se superponen en absoluto, siendo los conflictos muy poco frecuentes. Además, el tiempo que lleva resolver conflictos es mucho menor que el tiempo perdido por un sistema que implementa el modelo de bloqueo.

Como curiosidad decir que hay una situación donde el modelo Bloquear-Modificar-Desbloquear resulta mejor que este, y es cuando se tiene archivos no-fusionables, por ejemplo imágenes. Si dos personas cambian una imagen a la vez, no hay forma de fusionar esos cambios, y uno de ellos perderá sus cambios.

Copias de trabajo

A modo de recordatorio decir que el sistema de control de versiones *VCSGis* utiliza el modelo Copia-Modificación-Fusión como modelo de versionado, por lo que utiliza **copias de trabajo**.

Una copia de trabajo en *VCSGis* esta formada por una base de datos en su sistema local, la cual contiene una colección de tablas o capas que pueden ser modificados sin ningún tipo de problema. Su copia de trabajo es su área de trabajo privada y *VCSGis* nunca incorporará los cambios de otra gente, ni hará que sus cambios estén disponibles para los demás, a menos que se lo pida expresamente. Básicamente la copia de trabajo es una captura/fotografía/snapshot del repositorio en un instante concreto.

Una copia de trabajo de *VCSGis* también contiene algunas tablas extra, creados y mantenidos por *VCSGis*, para ayudar a llevar a cabo la gestión de cambios. En estos se almacenan los numero de revisión, un sistema de numeración única que identifica de manera inequívoca los diferentes cambios registrados en el repositorio. Estos codigos de revision, únicos para cada capa o tabla, ayudan al software a reconocer qué capas o tablas contienen cambios no publicados o qué capas o tablas contienen contenidos desfasados respecto a la información del repositorio.

Flujo de trabajo (Modelo Copiar-Modificar-Fusionar)

Este apartado define la estructura o forma de trabajar teórica con un sistema de control de versiones que presenta el modelo Copiar-Modificar-Fusionar como podría ser *VCSGis*. Especificar que estos mismos procesos serán explicados desde un punto de vista práctico para *VCSGis* en el apartado siguiente.

Paso 1: Creación de repositorio. Se lleva a cabo por el administrador y con el se da inicio al proceso.

Paso 2: Creación de copia de trabajo. Proceso realizado por los diferentes usuarios o clientes del repositorio y como se dijo en el apartado anterior consiste en la captura del repositorio en un instante determinado. Esa captura completamente funcional permite cualquier acción de edición o modificación y los cambios realizados en esta serán propuestos por el usuario para formar parte del repositorio.

Paso 3: Realización de cambios en la informacion de su copia de trabajo y verificación que funcionan correctamente

Paso 4: Publicación de cambios. El sistema de control de versiones *VCSGis* en este caso provee al usuario de comandos para publicar sus cambios en el repositorio por tanto estar estos disponibles para todos los usuarios registrado es en ese repositorio. En el caso de que los demás usuarios hayan publicado antes sus propios cambios, el software le provee de comandos para fusionar esos cambios dentro de su copia de trabajo tras leer el repositorio. La fusión puede presentar dos caso bien diferenciados:

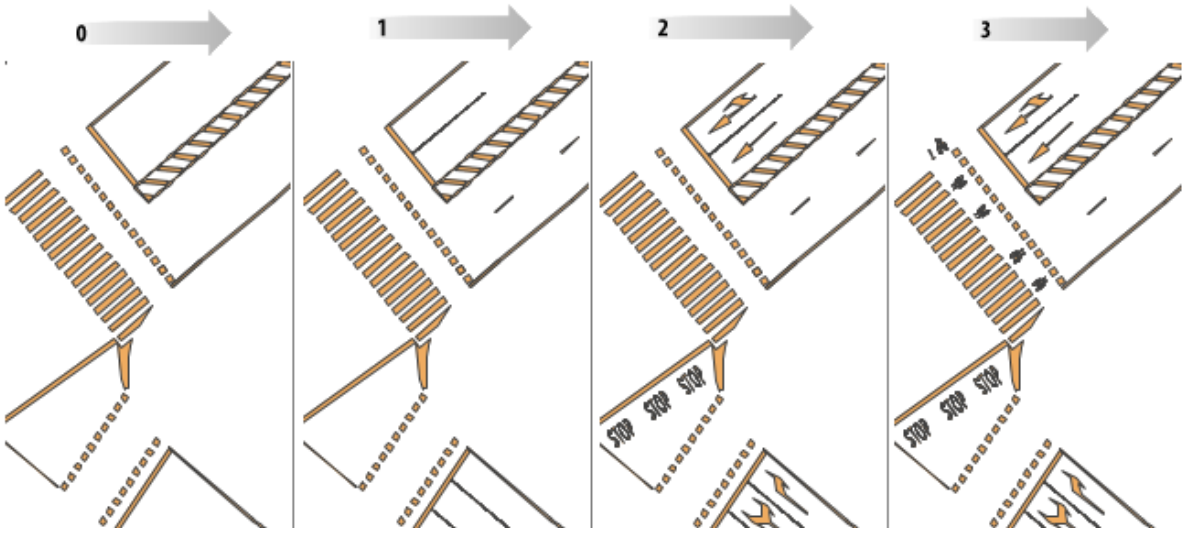
- Los cambios del usuario afectan a elementos no modificados anteriormente por los demás usuarios y por tanto se mezclan y fusionan las tablas.
- El usuario ha realizado cambios sobre elementos modificados anteriormente por otros usuarios y ya registrados en el repositorio, en ese caso se entra en conflicto y el usuario decide que cambios son los que se quedan finalmente en el repositorio los propios o los ya presentes.

Al acto de publicar los datos en el servidor se le denomina hacer **confirmar**, *commit*, y al proceso de actualizar la copia de trabajo con la información del repositorio se le denomina **actualizar**, *update*.

Revisiones

Volviendo al proceso de publicar o *confirmar* los datos en el servidor, hacer *commit*, este nos genera como resultado una **revisión** a la cual esta ligada cualquier cambio realizado en la tabla de la copia de trabajo modificada; cambiar el contenido de las tablas, crear, borrar, renombrar y copiar registros o geometrias... Siendo solo registrada en el repositorio si todos los cambios de esa tabla pueden realizarse (transacción atómica). En el caso de que algún cambio de una tabla no pueda hacerse, ninguno de los demás cambios de la tabla referentes a esa revisión se realizará. Las revisiones presentan un identificador único de la revisión y en el caso de VCSGis cada tabla de la copia de trabajo presentara un control de la revisión individual con identificadores únicos propios.

Para ilustrar lo anterior se presenta el siguiente esquema del funcionamiento de las revisiones;



En la primera revisión, revisión 0, se han creado las lineas base de la calzada, en la siguiente revisión, revisión 1, se añaden las separaciones entre carriles y medaneras, en la revisión 2 se añaden las feclas indicadores del sentido y señales de stop, y por ultimo, en la revisión número 3 se han añadido las señales del carril bici.

Tras la explicación anterior parece lógico pensar que las revisiones y su número o identificador único son el mecanismo que utilizan los sistemas de control de versiones y por tanto *VCSGis* para gestionar el estado de la copia de trabajo con respecto al repositorio y viceversa.

Estados de tablas y registros

Los estados que puede presentar la copia de trabajo y las diferentes capas y tablas que los componen son las siguientes;

- *Sin cambios y actualizado.* La tabla no se ha cambiado en la copia de trabajo y no se han confirmado cambios a esa tabla en el repositorio desde su revisión de trabajo. Una acción *commit* de esa tabla no hará nada, y una actualización de ella tampoco.

- *Cambiado localmente y actualizado.* La tabla ha sido cambiado en la copia de trabajo y no se ha confirmado ningún cambio a esa tabla en el repositorio desde su revisión base. Hay cambios locales que no se han confirmado en el repositorio, por lo que al hacer commit la tabla se confirmaran sus cambios actualizando el repositorio. Si se procede a hacer una actualización o update de la tabla no se realizarà ya que la copia de trabajo actual posee la versión mas moderna de esta presente en el repositorio.

- *Sin cambios y desactualizado.* La tabla no ha sido cambiado en la copia de trabajo, pero ha sido cambiado en el repositorio. La tabla deberá ser actualizado en algún momento para presentar el mismo contenido que la versión del repositorio. Un comando commit sobre la tabla no hará nada ya que no hay cambios en local y el comando actualizar o update traerá los últimos cambios del repositorio a su copia de trabajo.

- *Cambiado localmente y desactualizado.* La tabla se ha cambiado tanto en la copia de trabajo como en el repositorio. No podra ejecutar un *commit* sobre la tabla ya que necesita actualizar en primer instancia la copia de trabajo con el repositorio. Para ello necesita realizar una actualización o update de la copia de trabajo que intentará fusionar los cambios del repositorio con los cambios locales. Si *VCSGis* no puede completar la fusión de una forma plausible automáticamente, le dejará al usuario la tarea de resolver los conflictos. Por ultimo, una vez actualizado, se debera realizar una petición commit para registra los cambios locales en el repositorio y terminará el proceso.

Conexión con el repositorio

El sistema de control de versiones VCSGis dispone de dos opciones a la hora de realizar la conexión con el repositorio;

- Conexión a una BBDD de nuestra red local.
- Conexión a un servidor VCSGis.

La primera opción y más usual consiste realizar la conexión a un repositorio situado en un servidor local y se realiza identificando el fichero de la BBDD en la estructura de carpetas. La segunda opción al ser online permite la conexión al repositorio mediante URL, ampliando la operabilidad del software.

Notificaciones del repositorio

Un repositorio de VCSGis puede ser configurado para que cada vez que se den modificaciones en él este realice una notificación a un servicio externo. Un ejemplo de esto se da cuando un usuario realiza un “commit” contra el repositorio, VCSGis enviará una notificación indicando que el usuario en cuestión a realizado una operación de “commit” en una determinada tabla de repositorio, así como en que versión ha quedado dicha tabla. Normalmente las notificaciones se realizaran ha través de invocar a un servicio web mediante una llamada GET.

Para que se pueden llevar a cabo estas notificaciones hay que configurar en el repositorio un “hook”. Para configurarlo debemos crear una entrada nueva en la tabla VCSGISREPO_HOOK, indicando el modo de la notificación y la URL del servicio que ha de usarse para realizar dicha notificación.

Antes de enviar la notificación a la URL indicada, se personalizará esta con cuatro parámetros, todos ellos opcionales. Estos parámetros son:

- 1. Código del usuario que ha realizado la operación.
- 2. Operación realizada.
- 3. Nombre de tabla o entidad sobre la que se ha realizado la modificación.
- 4. Código de revisión en el que ha quedado la tabla implicada en el repositorio.

Un ejemplo de URL podría ser:

http://localhost:9810/hooktest?userCode=%1$s&request=%2$s&table=%3$s&revisionCode=%4$s

En esta URL se incorporaran los cuatro parámetros que se pueden establecer, estando referenciados como %1$s, %2$s, %3$s y %4$s. El mecanismo de envio de notificaciones sustituirá estos valores por el correspondiente parámetro. No es preciso indicar los cuatro parámetros en la URL, de hecho no es preciso indicar ninguno de ellos, ya que todos son opcionales tal y como se dijo con anterioridad. Así mismo, en el ejemplo, estos valores se asignan a los parámetros de la URL “userCode”, “request”, “table” y “revisionCode”, pero puede especificarse cualquier otro nombre para estos parámetros, adaptándose así a las necesidades del servicio.

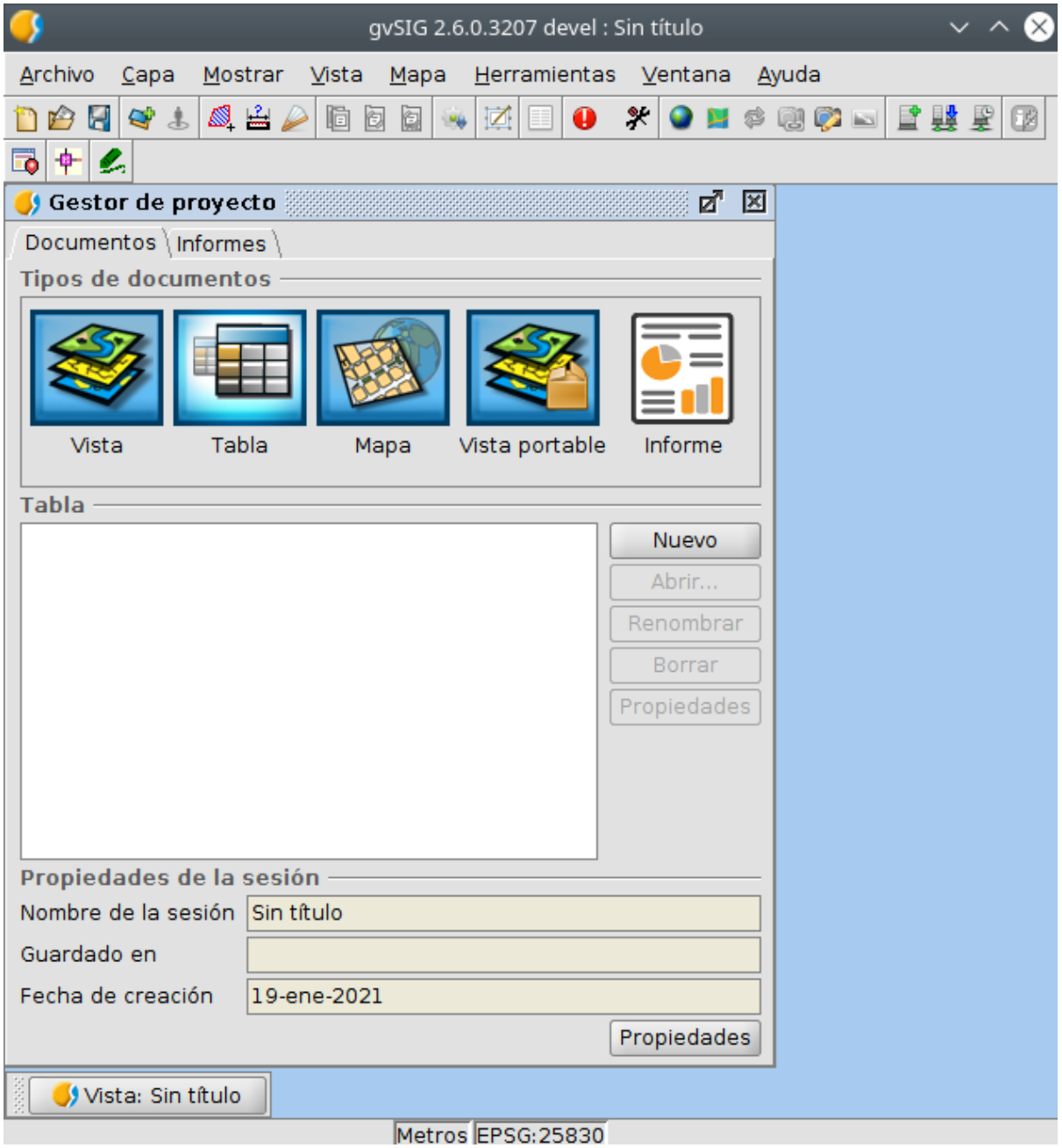
Cuando se realiza una operación contra el repositorio, se añade a la cola una petición de envio de notificación, resolviéndose dicha operación inmediatamente, sin esperar a que esta sea notificada a los servicios que estén registrados para ello. Los notificaciones serán enviadas a los distintos servicios configurados por el orden en el que se han realizado las operaciones.

Siempre que se configure el repositorio para enviar una notificación, deberemos tener en cuenta el rendimiento de este. El tiempo de envio de notificaciones a otros servicios puede verse afectado si existen servicios que reciben notificaciones caídas o que no responden en un tiempo razonable. A la hora de implementar un servicio de recepción de notificaciones tenemos que tener en cuenta que este no debe hacer operaciones “pesadas” durante el procesamiento de la notificación. Es recomendable que el servicio las añade a la cola y se vayan procesando en una tarea o hilo de ejecución independiente.

Un aspecto importante a destacar es que, en la notificación nunca se envían los registros involucrados en la operación que generó esta. En caso de que se requieran, sera responsabilidad del servicio que la reciba determinar en que revisión se encuentran los datos que tiene de esa tabla, y si no están en la correcta, solicitar al servidor VCSGis que le envie los datos de esa tabla que han cambiado desde la revisión en la que están los suyos hasta el HEAD o la revisión indicada en la notificación.

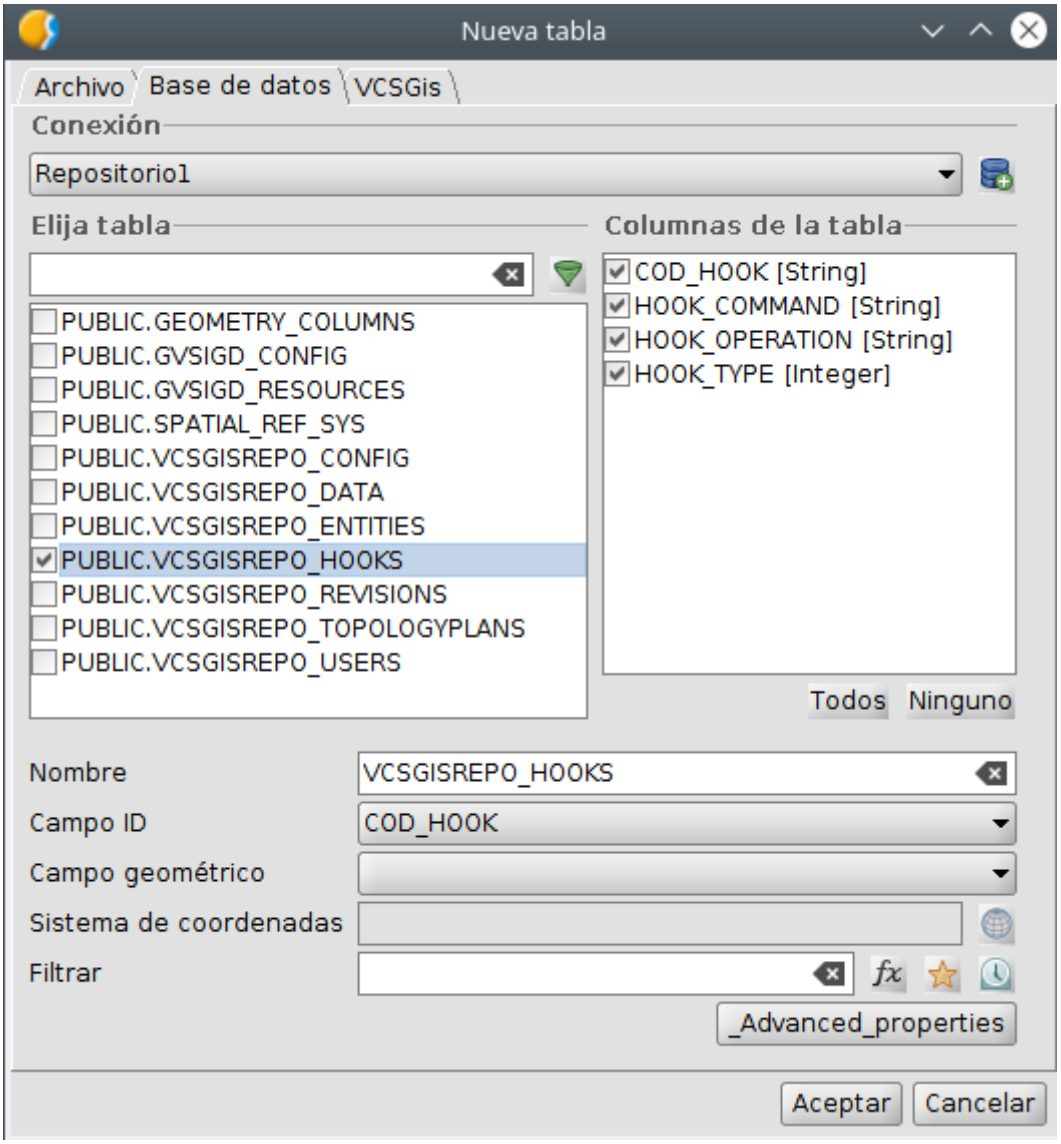
Y... **¿ como registramos en el servidor VCSGis que queremos que nos notifique cuando se realizan operacion contra el ?**

Para registrar una nueva notificación, deberemos tener acceso a la base de datos en la que se encuentra el repositorio. Si estamos usando *gvSIG Desktop*, crearemos una conexión a esa base de datos en caso de no tenerla y cargaremos la tabla **VCSGISREPO_HOOKS** como una tabla del proyecto gracias al gestor de proyectos.

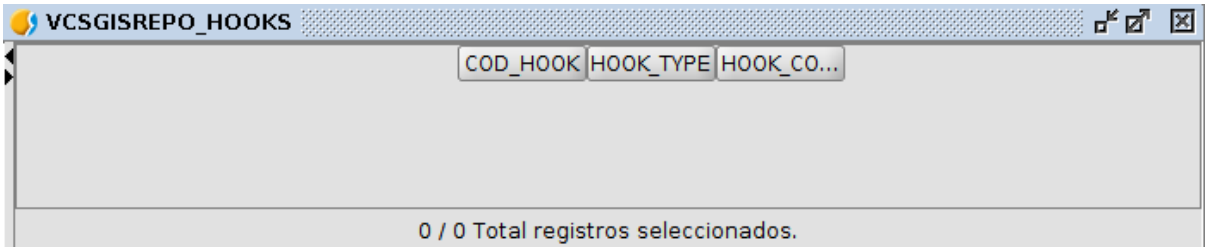


La carga de la capa se realiza seleccionando la opción *Tabla* y posteriormente el

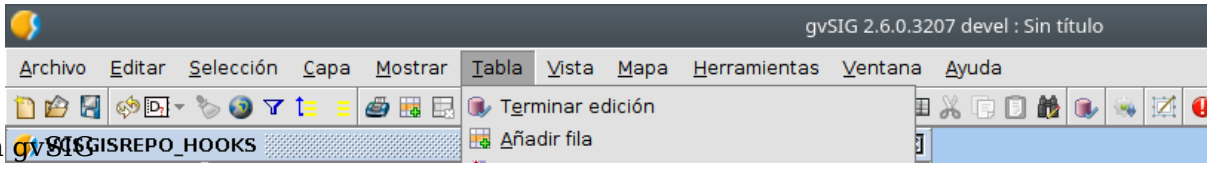
boton *Nuevo*.Tras eso se selecciona la opción *Base de datos* en las pestañas de selección de fuente de datos situados en la zona superior de la ventana. Una vez allí, se selecciona la conexión con el repositorio y se carga la tabla **VCSGISREPO_HOOKS**.

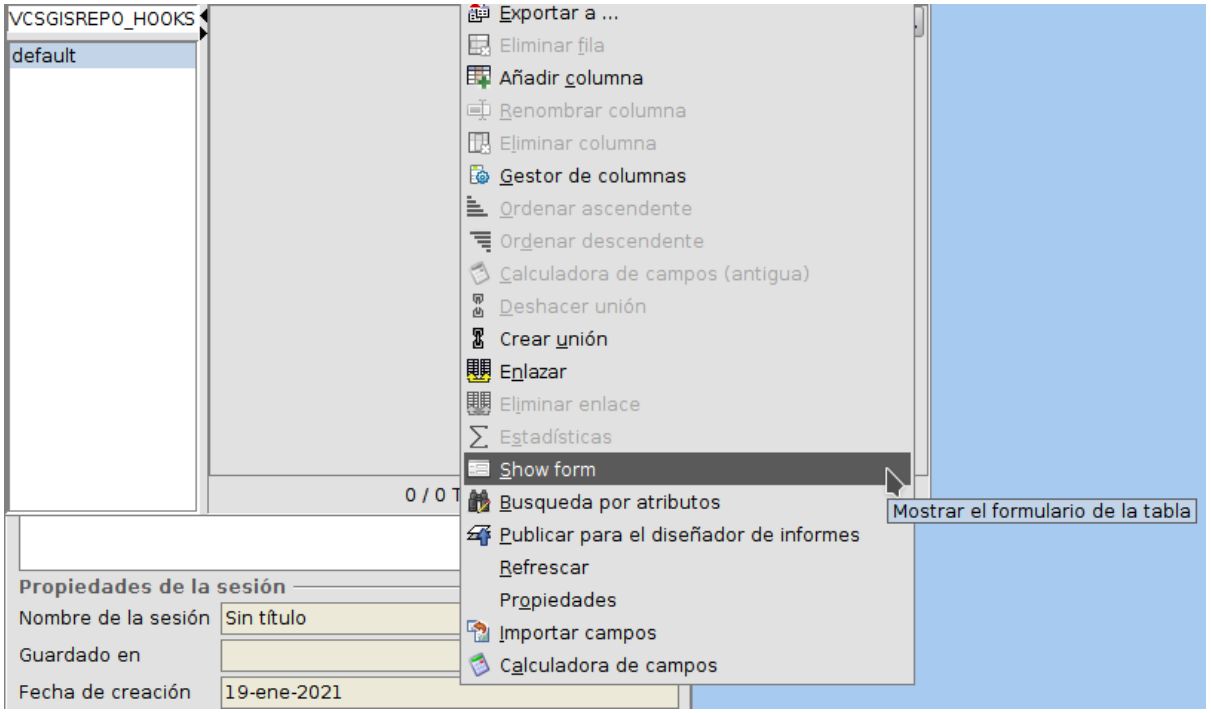


Como resultado se muestra la tabla **VCSGISREPO_HOOKS**, la cual no presenta ningún registro.

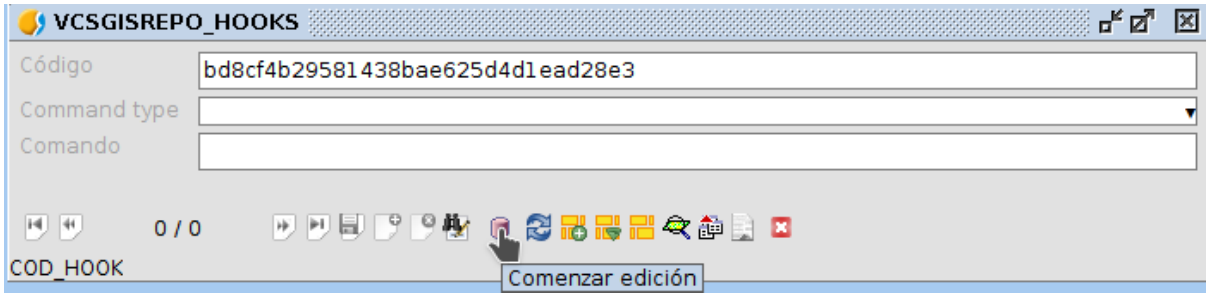


Una vez visualizada la tabla se procede a mostrar el formulado asociado a esta. Una de las múltiples formas de obtener esta herramienta se realiza al ejecutar el comando *Show Form* situado en la pestaña *Tabla* de gvSIG Desktop.

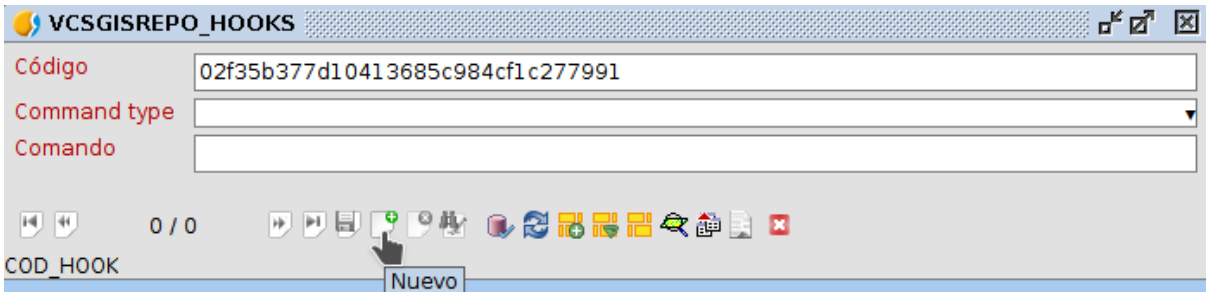




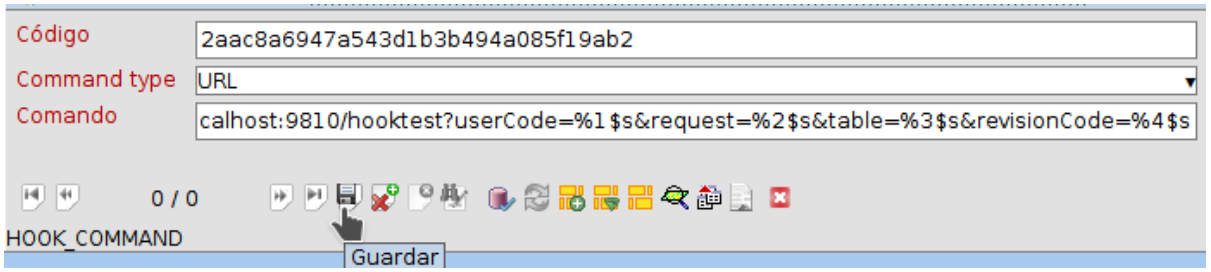
Una vez diponemos del formulario de la tabla hay que poner poner en edición esta para agregar un nuevo registro. Este proceso se puede realizar desde el mismo desplegable que se mencionó anteriormente para obtener el formulario, o desde el mismo formulario utilizando el botón *Comenzar edición*.



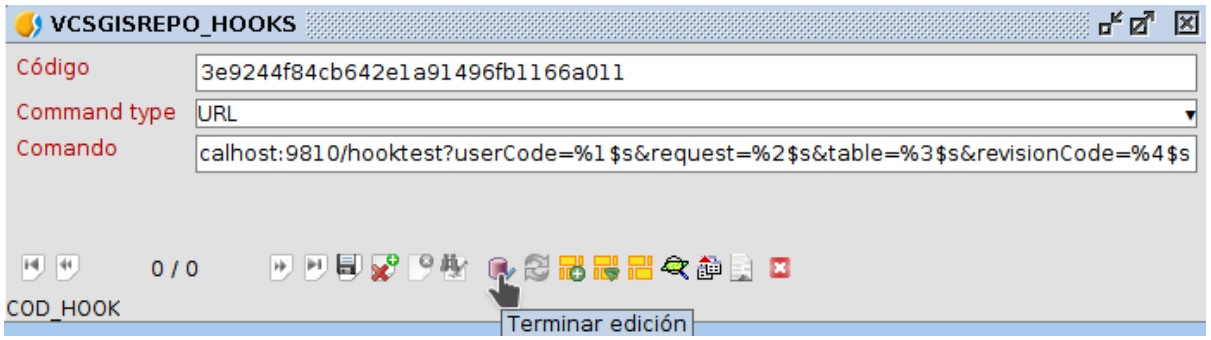
Una vez visualizado el formulario y con la tabla en edición, ver siguiente ilustración, se procede a crear un nuevo registro pulsando el icono *Nuevo* situado en la zona inferior del formulario e indicado en la imagen siguiente.



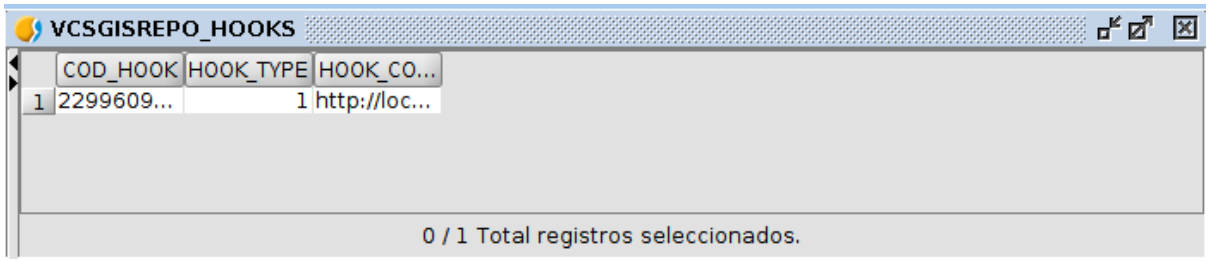
Tras lo anterior se rellena los campos del formulario. El primer campo, *Código*, se rellena de manera automática con un identificador único. El segundo campo, *Command Type* dispone de las opciones *URL* y *Shell*, seleccionando la primera. El tercer campo, *Comando*, permite almacenar el objeto en concreto especificado en el campo anterior, en nuestro caso la URL del ejemplo. Tras terminar de completar e formulario solo queda guardar los cambios con el icono *Guardar* presente en la zona inferior de la ventana y especificado en la próxima imagen.



Una vez realizado lo anterior solo queda terminar la edición de la tabla. Para ello hay que ir a la pestaña *Tabla* de *gvSIG Desktop* y ejecutar el comando *Terminar edición* o terminar la edición desde el mismo formulario.



Realizado lo anterior ya podemos ver que la tabla **VCSGISREPO_HOOKS** presenta un registro de notificación.



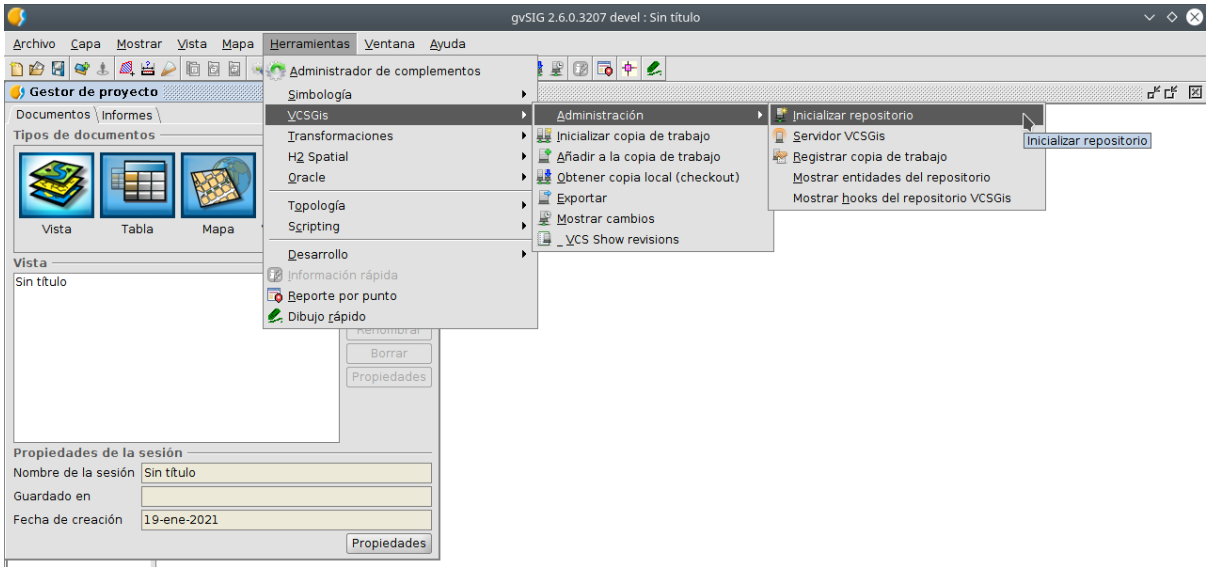
Utilización básica

A continuación, como se mencionó en el Apartado *Flujo de trabajo (Modelo Copiar-Modificar-Fusionar)* se realiza un aproximación más práctica a la manera o forma de trabajar que tiene el software VCSGis. Para llevar a cabo lo anterior se seguir el flujo de trabajo indicado en el apartado antes mencionado, adaptando los puntos a nuestro sistema de control de versiones.

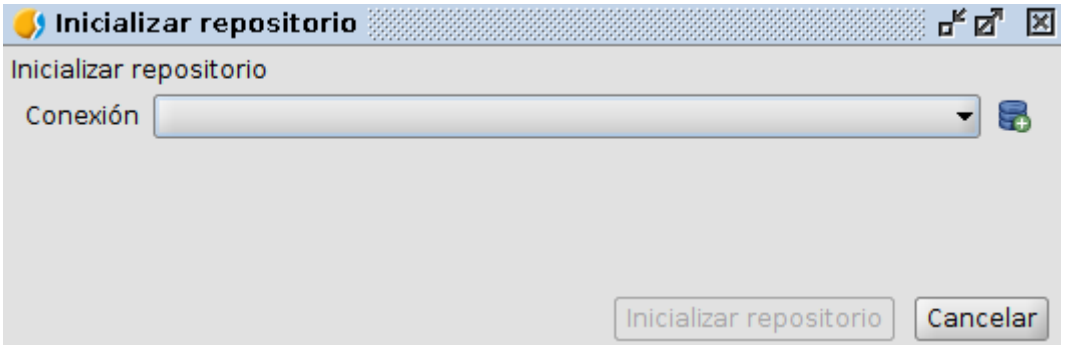
Creación de un repositorio

Para la creación del repositorio tal y como se destaca anteriormente en el documento disponemos de dos opciones, conexión local a un BBDD como repositorio o conexión a un servidor o repositorio remoto (online). En esta guía nos inclinaremos por la primera opción o conexión local.

Para inicial la conexión al repositorio hay que dirigirse a la pestaña *Herramientas*, desplegarla y buscar la opción *VCSGis*. Una vez dentro seleccionaremos la opción *Administración* y dentro de esta *Inicializar repositorio*. Puede ver gráficamente lo mencionado anteriormente en la siguiente ilustración.

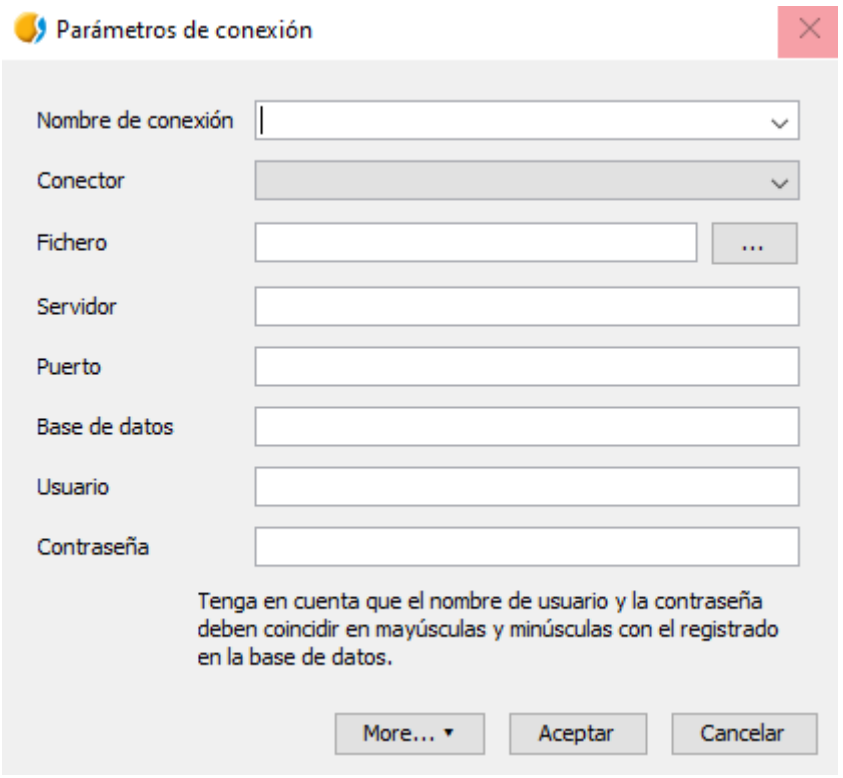


Tras pulsar se obtiene la siguiente ventana la cual nos permite seleccionar una conexión a un a BBDD ya existente o generar una nueva.



En el caso de que se busque crear una nueva conexión hay que hacer clic sobre el botón situado a la derecha del desplegable en la ventana anterior, acción que iniciará el proceso de creación de una nueva conexión a una BBDD. Esta nueva

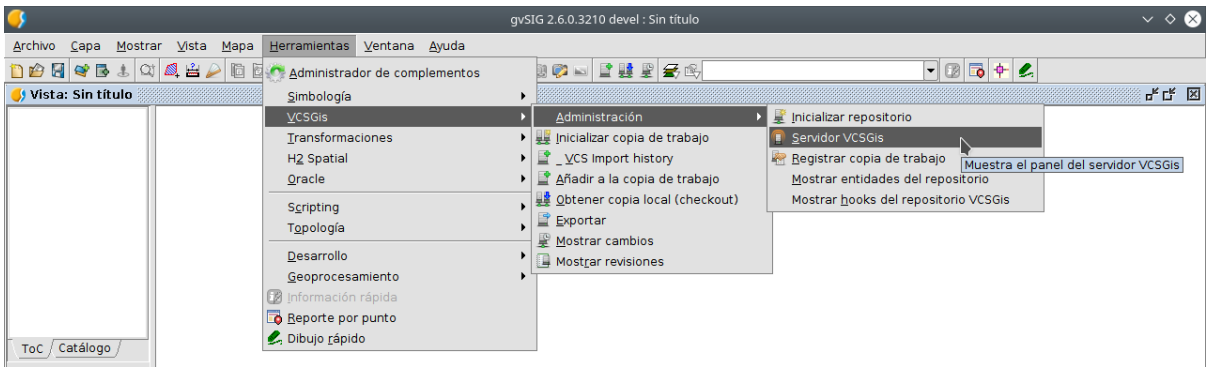
conexión se realiza mediante la ventana genérica de conexiones a este tipo de datos de gvSIG Desktop, ver la figura siguiente.



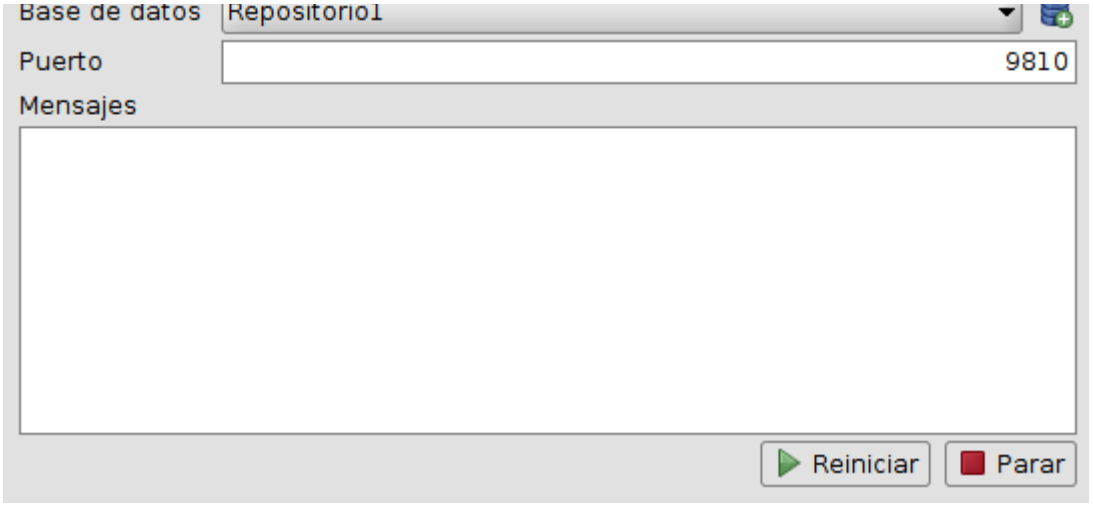
Como se detallo en la explicación teórica, la creación es un proceso destinado a ser ejecutado por el administrador, un usuario regular no debería llevar a cabo este proceso.

Iniciar servidor VCSGis remoto

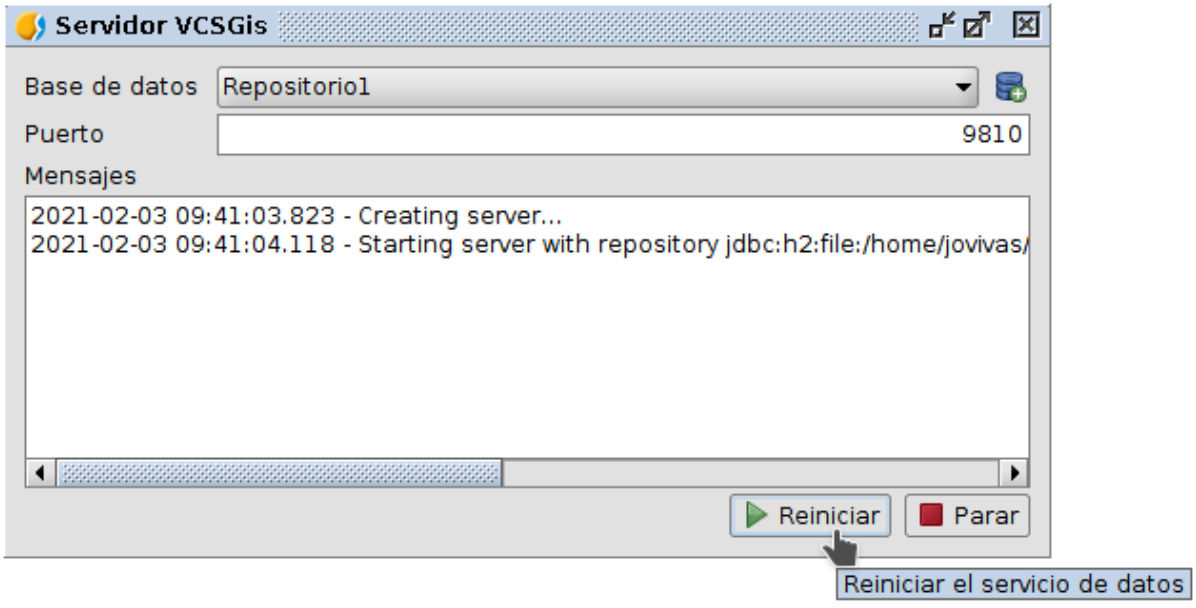
Como se dijo en el apartado anterior, la herramienta de control de versiones VCSGis dispone de la posibilidad de suministrar un repositorio local a otras máquinas mediante la inicialización de este en un servidor propio en gvSIG desktop. En otras palabras VCSGis permite que un único equipo almacene el repositorio y este suministre la información a los demás equipos mediante una conexión (URL) al repositorio remoto. Esta opción se puede configurar ejecutando el comando *Servidor VCSGis* situado dentro de *Herramientas*, desplegable *VCSGis*, despleglable *Administración*.



Tras la ejecución del comando anterior se obtiene el siguiente interfaz.

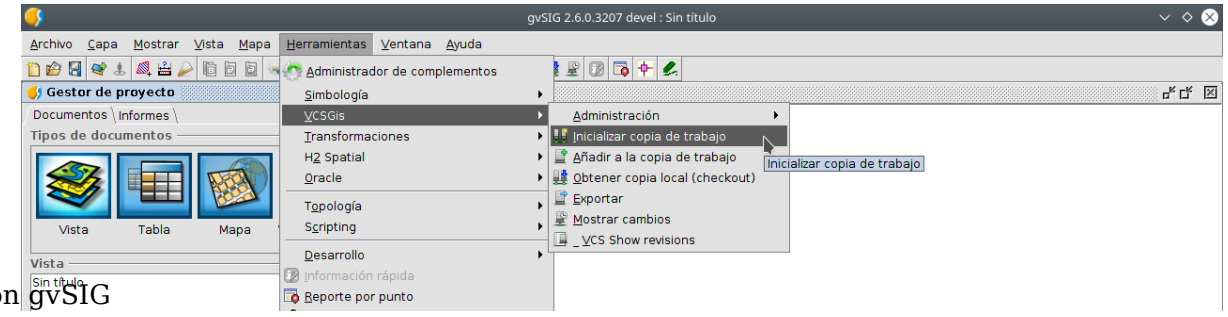


En esta ventana hay que especificar la base de datos o repositorio que va a ser suministrado mediante conexión remota así como el puerto por el cual se va a enviar la información. Una vez especificado lo anterior solo queda iniciar el proceso pulsando el botón *Reiniciar* situado en la zona inferior de la ventana. Puede también pararse el suministro de dicha información con el botón *Parar* adyacente al anterior. Tras iniciar el proceso la ventana en la zona central muestra las peticiones realizadas al servidor.

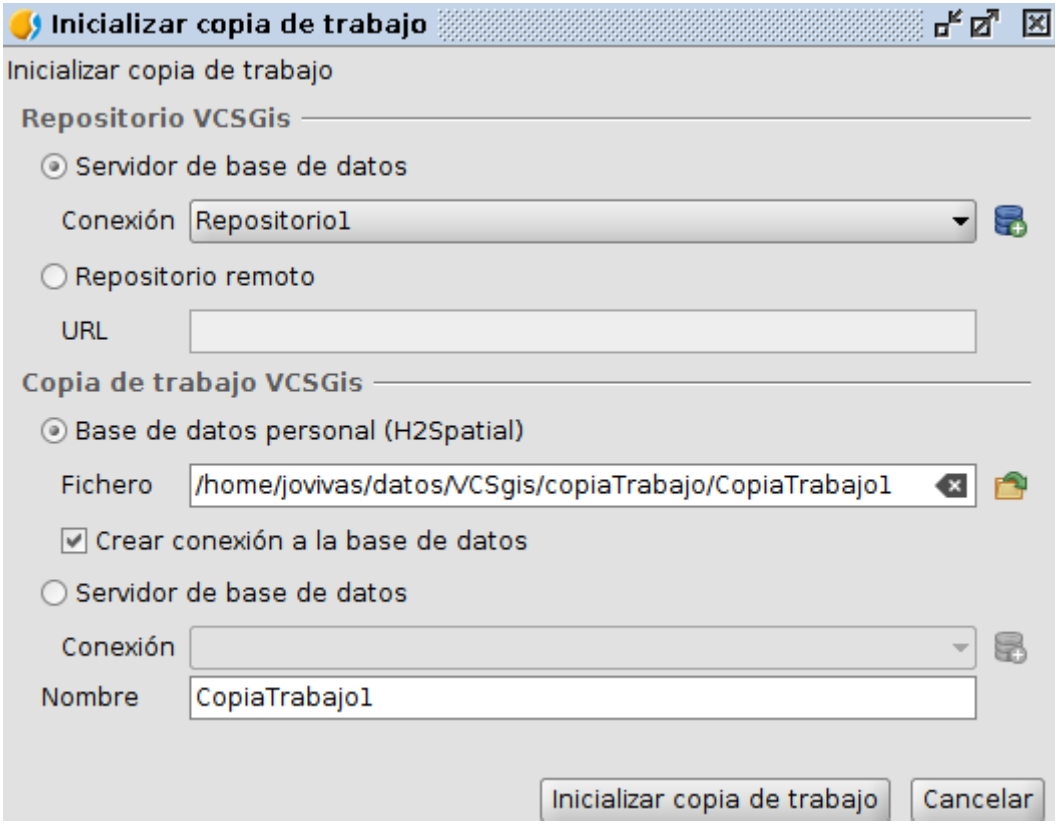


Creación de una copia de trabajo

El siguiente paso es crear la copia de trabajo, proceso realizad al ejecutar el comando *Inicializar copia de trabajo* dentro de las pestaña *VCSGis* situada en la opción *Herramientas* de gvSIG Desktop.



Una vez ejecutado lo anterior se obtiene la siguiente venta cuyos componentes se dividen en dos apartados, *Repositorio de VCS* y *Copia de trabajo de VCS*.

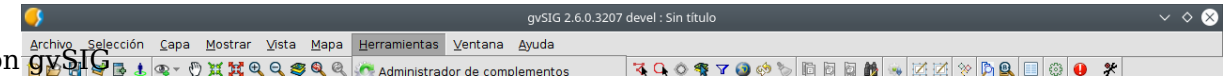


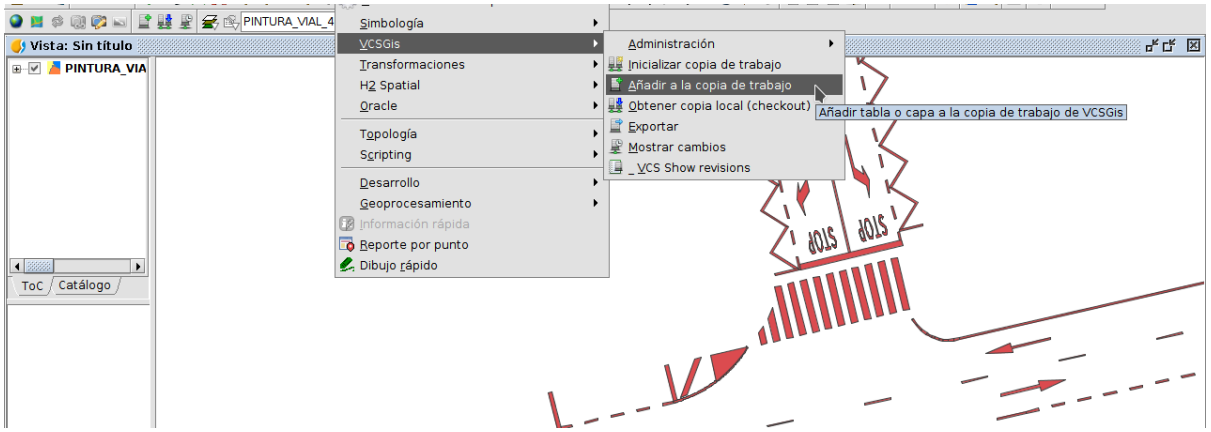
En el apartado *Repositorio del VCS* se selecciona la conexión al repositorio, pudiendo ser local u online, mientras que el apartado *Copia del trabajo VCS* se introducen los parámetros para generar la cueva copia de trabajo. Los parámetros necesarios son el nombre de la copia y la situación de esta, la cual puede ser local, genera un BBDD personal en formato H2Spatial, u online. Si se selecciona la opción local, marcar la opción *Crear conexión a la base de datos* como recomendación.

Añadir una capa al repositorio

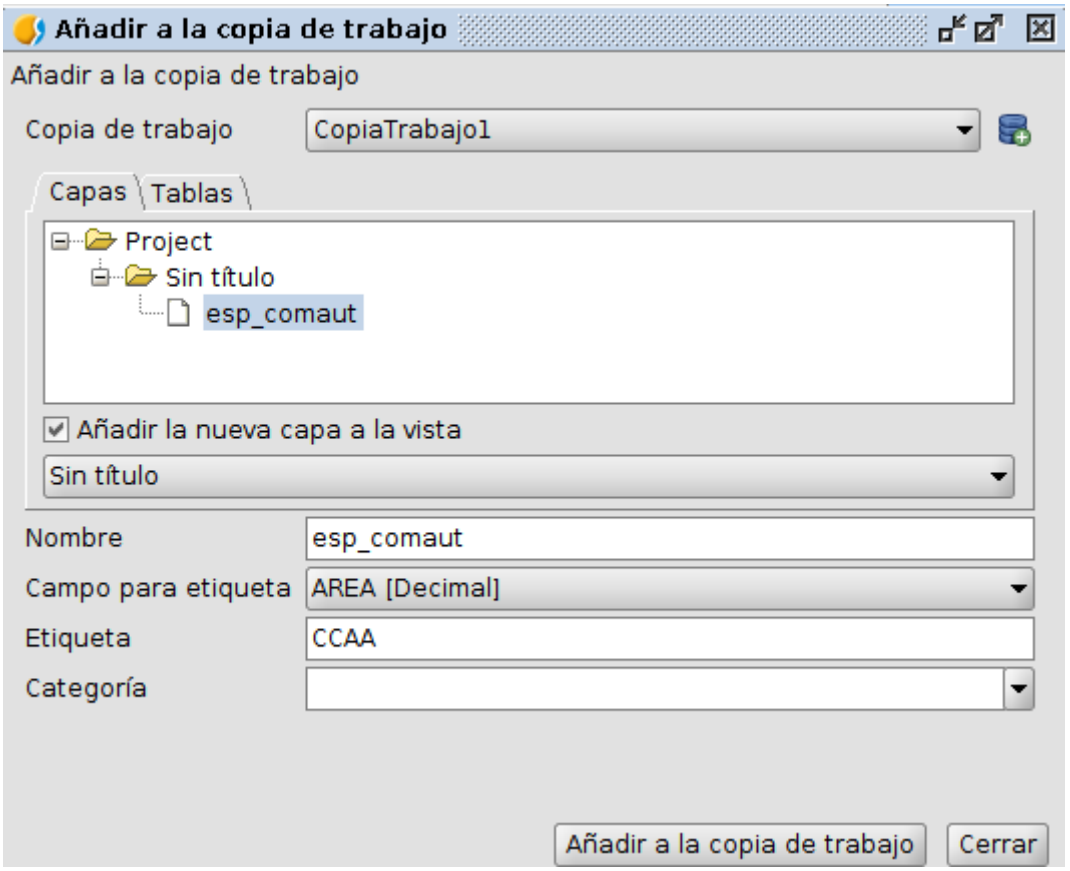
Tras cerciorarnos de la existencia de un repositorio y de una copia de trabajo se procede a actualizar este con nueva información. Para realizarlo hay que cumplir dos pasos.

El paso 1 consiste en cargar dicha nueva información o capa en la vista. El segundo paso se basa en ir a la opción *Herramientas* del menú de gvSIG Desktop, pestaña *VCSGgis* y pestaña *Añadir a la copia de trabajo*.

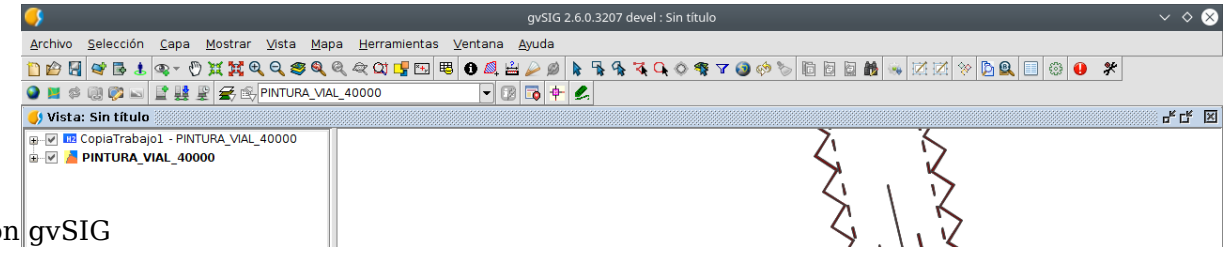


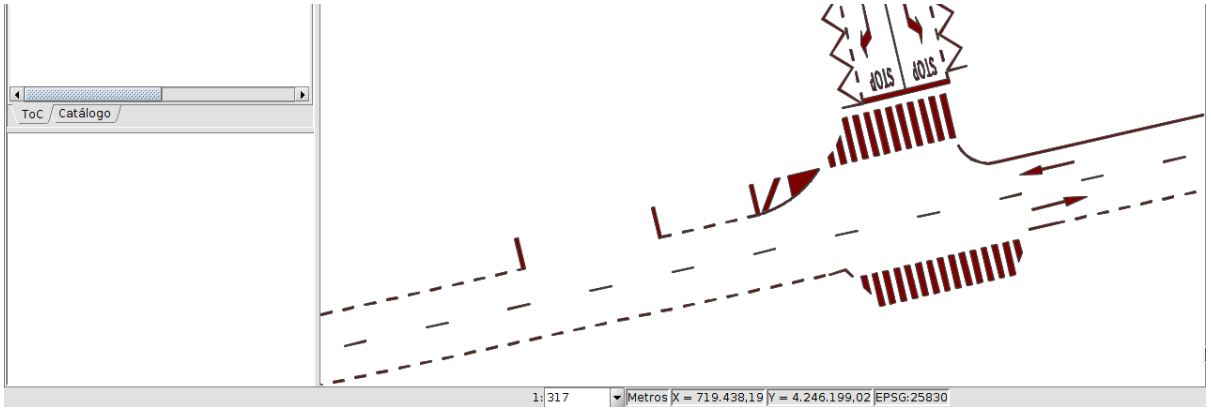


Tras pulsar el comando *Añadir a la copia de trabajo* se desplegá la ventana siguiente donde se selecciona la copia de trabajo donde queremos añadir la capa, la estructura de carpetas actual de gvSIG donde se selecciona la información a añadir y una serie de opciones.



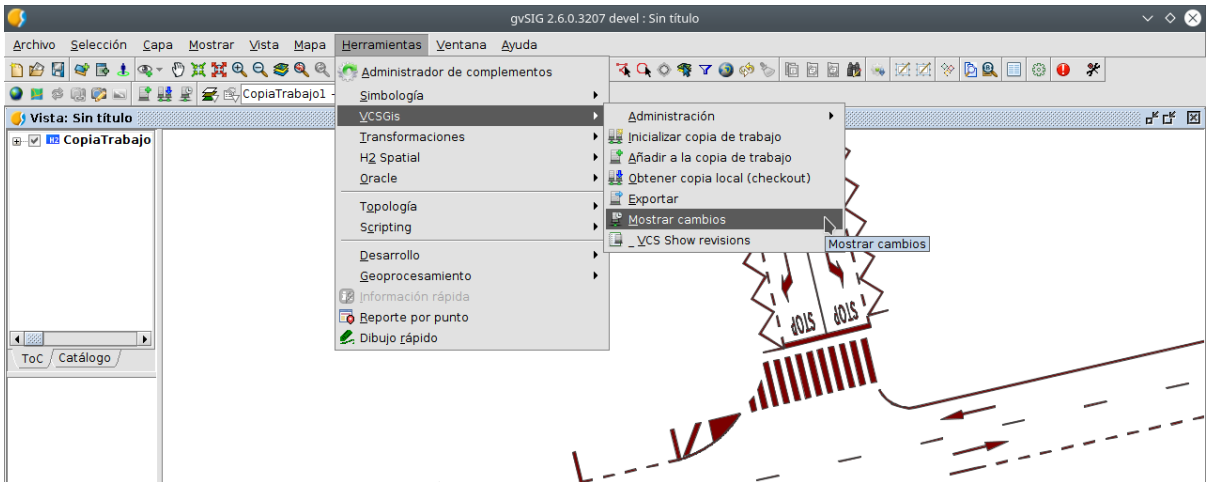
Entre las opciones destaca *Añadir esa capa a la vista*, opción recomendada para iniciar el proceso de edición utilizando el control de versiones. Hay que hacer incapie en la idea anterior, la capa añadida recientemente es sobre la que hay que trabajar ya que es la que tiene un control de versiones asociado, la capa inicial usada para introducir el dato puede eliminarse de la vista ya que los cambios sobre ella no se registan en VCSGis.



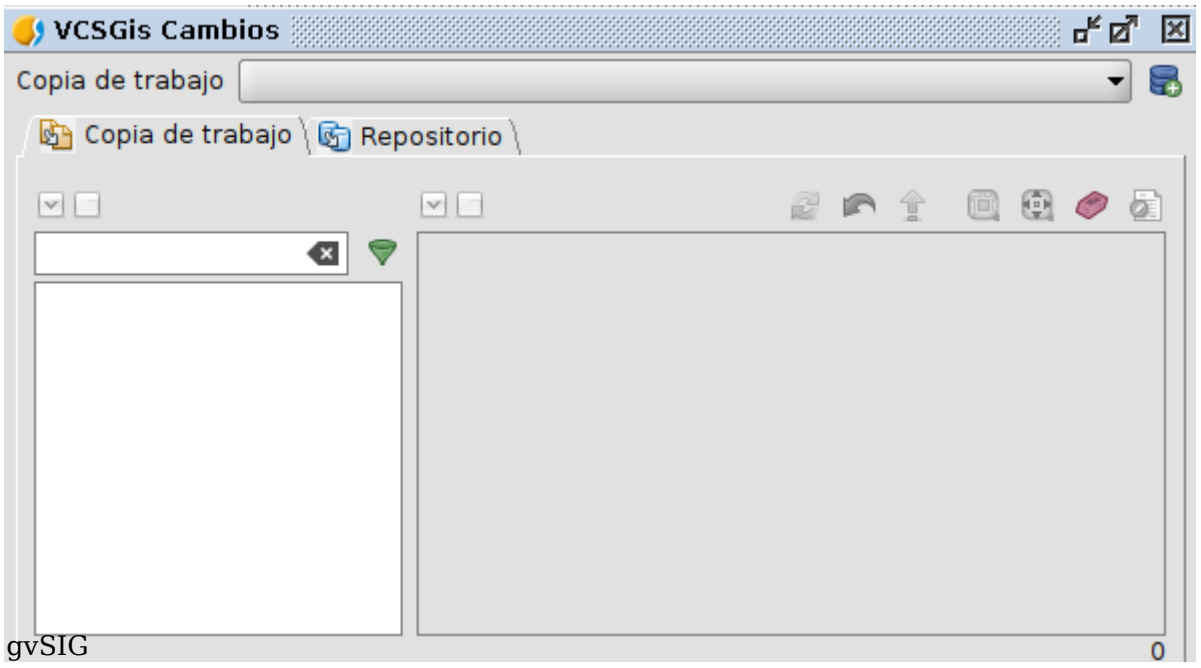


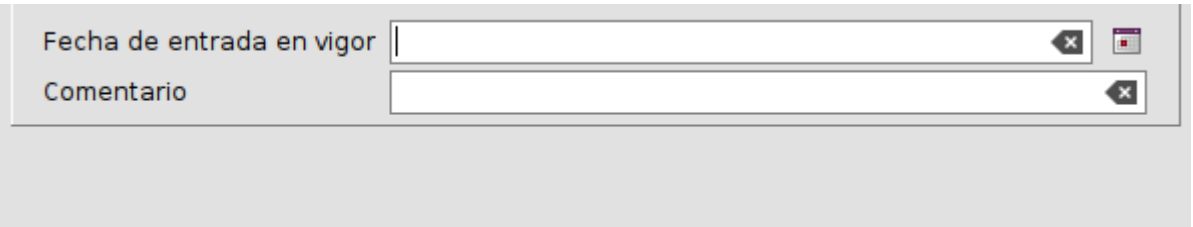
Las opciones restantes son nombre, campo para etiquetas, etiqueta y categoría. Estas opciones permiten respectivamente renombrar la capa en el árbol de la vista, seleccionar que campo queremos que sea representado como campo de etiquetas respectivamente, crear una etiqueta referida a esa capa y asignar la capa a una categoría definida por el usuario.

El proceso de añadir la nueva capa al repositorio termina cuando tras realizar la carga de información anterior se ejecuta la el comando *Mostrar cambios* situado en la pestaña *VCSGis* dentro de la opción *Herramientas* de del software.



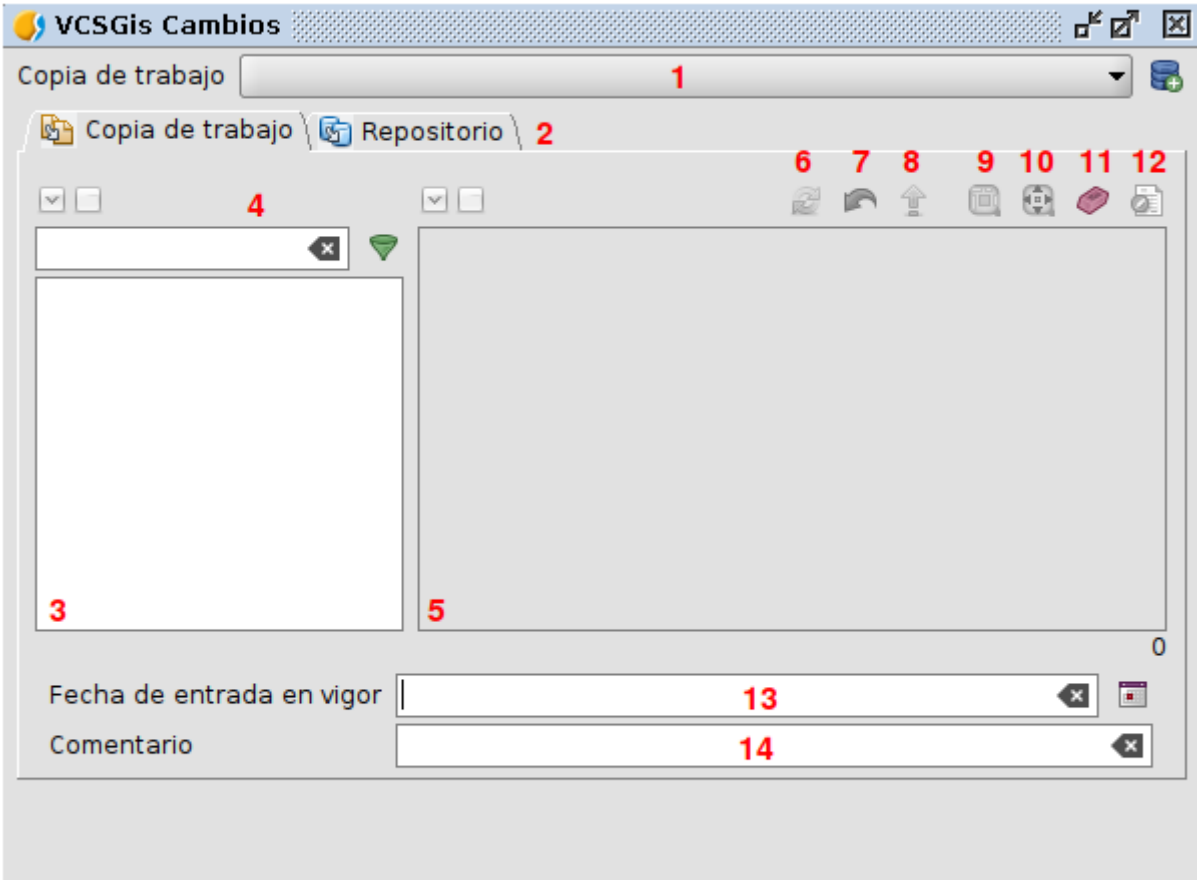
La ejecución de lo anterior genera la siguiente ventana.





La ventana anterior o ventana *Mostrar cambios* es una de las ventanas más importante de VCSGis y es la encargada como su propio nombre indica de mostrar los cambios entre la copia de trabajo actual y el repositorio. Los cambios entre información se detectan seleccionando la copia de trabajo en cuestión y mediante la gestión de las pestañas *Local* y *Remoto*. Al seleccionar la pestaña ***Local*** se muestran los ***cambios existentes en la copia de trabajo frente al repositorio***. Si por el contrario se selecciona la opción ***Remoto*** se muestran los ***cambios del repositorio frente a la copia de trabajo***. Además de las pestañas, la ventana permite identificar la *Fecha de entrada en vigor* así como un apartado *Comentarios* asociados a los cambios que se van a enviar al repositorio.

Los componentes seleccionada la opción *Local* de esta se listan a continuación:



1. Desplegable para selección de la *Copia de trabajo* sobre la que ver los cambios.
2. Pestañas Local/Remoto.
3. Área de visualización de capas presentes en la copia local. Dependiendo del color que presente en esta zona, las capas presentarán cambios locales o no.
4. Elementos de filtrado para realizar búsquedas de capas.
5. Área de visualización de cambios.
6. Botón de refrescar el área de visualización.
7. Botón revert. Esta función permite deshacer los cambios locales presentes en el área de visualización de cambios.

8. Botón para enviar o *confirmar* los cambios locales al repositorio (commit).
9. Botón que marca la geometría con cambios locales.
10. Botón que centra la vista en la geometría con cambios locales.
11. Botón que elimina la marca de la geometría con cambios locales, generada con el componente 6.
12. Botón que muestra un folmulario con los datos del registro seleccionado.
13. Campo para indicar la fecha de entrada en vigor de los cambios que van a ser enviados al repositorio.
14. Campo para introducir un comentario a los cambios que van a ser enviados al repositorio.

Los componentes seleccionada la opción *Remoto* de esta se listan a continuación:

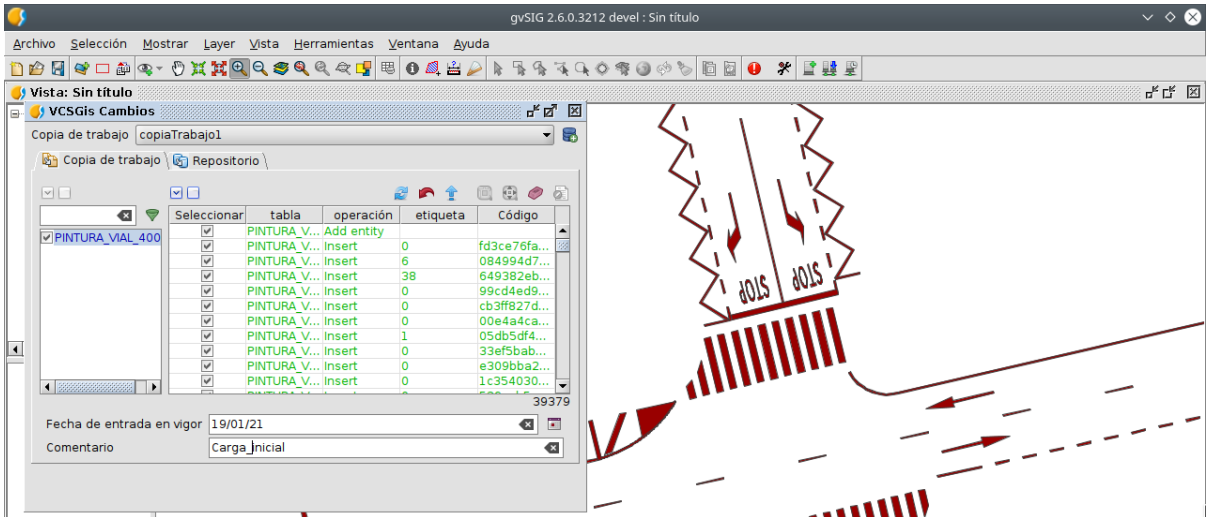
1. Desplegable para selección de la *Copia de trabajo* sobre la que ver los cambios.
2. Pestañas Local/Remoto.
3. Área con lista de capas de la *Copia de trabajo*.
4. Botón de refrescar la lista de capas de la *Copia de trabajo*.
5. Botón que permite hacer de manera directa una copia local (checkout) de la capa seleccionada.
6. Botón para descargar los cambios que han habido en la capa seleccionada en el repositorio desde la ultima vez que se actualizó esta en la *Copia de trabajo*. Esta operacion puede ser pesada dependiendo de la cantidad de cambios que hayan en el repositorio desde la ultima actualización de la *Copia de trabajo*.
7. Área de visualización de cambios. Muestra los cambios que han habido en el repositorio en relacion a las capas de la *Copia de trabajo*.
8. Actualiza el área de visualización de cambios, releyendo estos de la informacion almacenada en la *Copia de trabajo*. No accede al repositorio para actualizarlos.
9. Elimina la lista de cambios de la *Copia de trabajo*. El usuario debera descargarlos de nuevo del repositorio en caso de que desee verlos.
10. Botón *Update*. Se actualizaran las tablas locales,*Copia de trabajo*, con los cambios del repositorio. Si hay cambios locales se perderán. Esta opción no

esta disponible por defecto si existen conflictos entre los cambios del repositorio y los cambios realizados en la *Copia de trabajo*. En el caso que existiendo conflictos se busque prevalecer la información remota frente a la local, se tendria que marcar todos los cambios del repositorio, proceso que habilita dicho botón en la ventana *Mostrar cambios*.

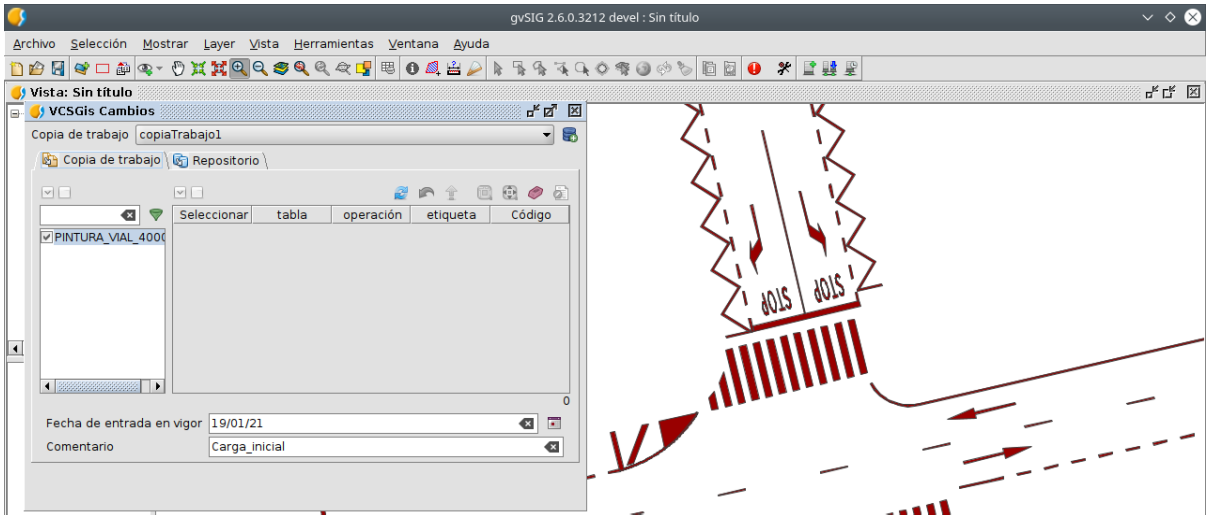
11. Botón *Merge*. Se mezclan o fusionan los cambios de la capa del usuario de la *Copia de trabajo* con los cambios del repositorio. El proceso es simple, los cambios remotos que no tengan conflicto, cambios de color diferente a rojo, se actualizan en la *Copia de trabajo*, al igual que los cambios con conflictos, cambios de color rojo, con el check seleccionado. Marcar el check de los cambios en conflicto implica que prevalecerán los cambios del repositorio para esos determinados elementos frente a los existentes en la *Copia de trabajo* o capa local. Si lo que se busca es mezclar la capa local y remota asegurando que los cambios locales sean los finales, no hay que marcar ningún cambio en conflicto y realizar la operación *Merge*. Esta operación solo estará activa si se han detectado conflictos entre la *Copia de trabajo* y el repositorio.
12. Botón que marca la geometría con cambios en el repositorio.
13. Botón que centra la vista en la geometría con cambios en el repositorio.
14. Botón que elimina la marca de la geometría con cambios en el repositorio, generada con el componente 12.
15. Botón que muestra un folmulario con los datos del registro seleccionado.

Esta opción de la ventana se explicará con mayor detalle en los apartados siguientes mediante ejemplos.

Volviendo al caso en cuestión tras ejecutar el comando *Mostrar cambios* y al pulsar la opción *Local* en el área destinada a estos aparecen registros correspondientes a los elementos de la capa a añadir, ver siguiente imagen. Esto se debe a que hay diferencias entre la copia de trabajo y el repositorio, presentando la copia de trabajo una serie de nuevos elementos, una capa, que el repositorio carece. El proceso de añadir una nueva capa al repositorio finaliza si seleccionamos esos registros y pulsamos el botón que realiza un *commit*, componente 8 de la ventana. Con esa acción el repositorio se actualiza con la información de la copia de trabajo y tendría por tanto la nueva capa a su disposición.



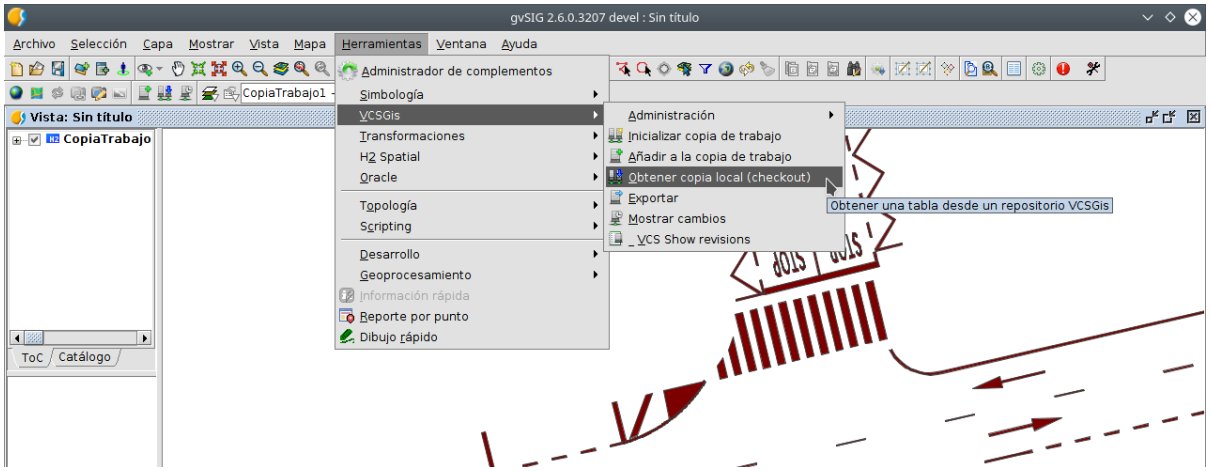
A modo de comprobación tras lo anterior si se selecciona la pestaña *Local* y se pulsa el botón de actualizar o refrescar área de visualización (componente 6 de la ventana), esta aparecerá vacía.



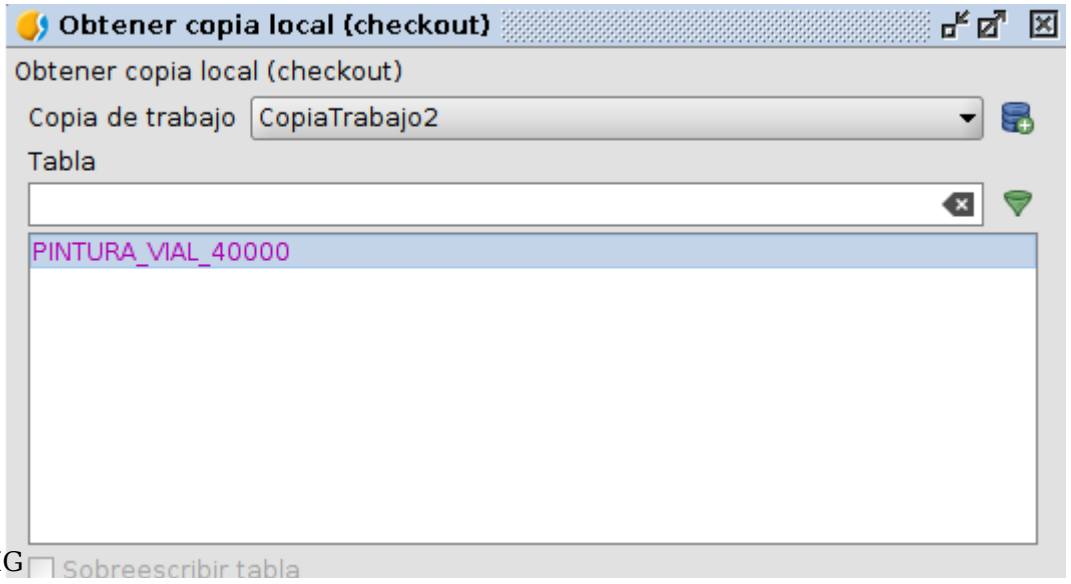
Añadir una capa del repositorio

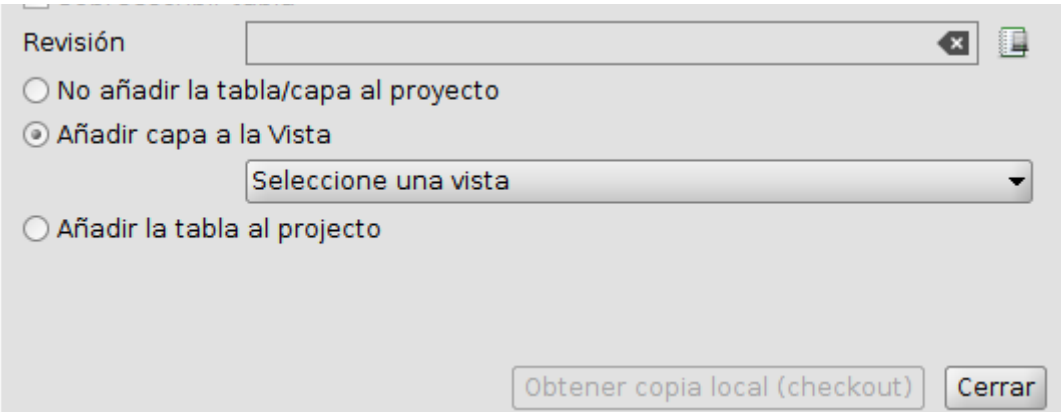
El procedimiento de añadir una capa del repositorio a la copia de trabajo puede llevarse a cabo de dos manera diferentes.

La primera forma, muy similar a *Añadir capa al repositorio*, se realiza ejecutando el comando *Obtener copia local* (checkout) situado en la opción Herramientas del menú de *gvSIG Desktop*, pestaña *VCSGgis*.



Como resultado de la ejecución anterior se obtiene la siguiente ventana:





En la ventana hay que indicar la copia de trabajo a la que se quiere añadir la capa procedente del repositorio y la capa en cuestión, seleccionando esta de la lista presente. Además de lo anterior, la ventana presenta varias opciones en función si la tabla/capa ya se encuentra en la *Copia de Trabajo*.

Si la Tabla/capa se encuentra ya en la copia en cuestión, se puede utilizar la opción *Sobreescribir tabla*. dicha opción borra la información local de la tabla y vuelve a descarga la tabla desde el repositorio. En el caso de que la tabla nunca haya existido en la copia la opción sobreescribir aparecerá deshabilitada y se podrá cargar esta cumplimentando el resto de parámetros; *No añadir la tabla/capa al proyecto*, *Añadir capa a la vista* y *Añadir la tabla al proyecto*.

Otra cosa a destacar es la posibilidad de cargar la capa en la copia para una determinada revisión, para ello se dispone de un parámetro específico.

Hacer hincapié también en una idea importante, si en el proceso de obtener una copia local, es decir una capa del repositorio no se marca la opción *Añadir capa a la vista*, seleccionando la vista deseada, la capa se almacena en la copia de trabajo pero no se representará en la vista. Llegado a este punto se tendría que cargar la capa mediante mediante el dialogó estandard de *Añadir capa* de gvSIG Desktop.

La segunda manera de realizar el proceso se realiza con el diálogo genérico de *Añadir capa* de gvSIG Desktop. Esta forma de proceder puede verse en el apartado **Añadir capa VCSGis usando el diálogo “Añadir Capa” de gvSIG Desktop**.

Ciclo de trabajo básico

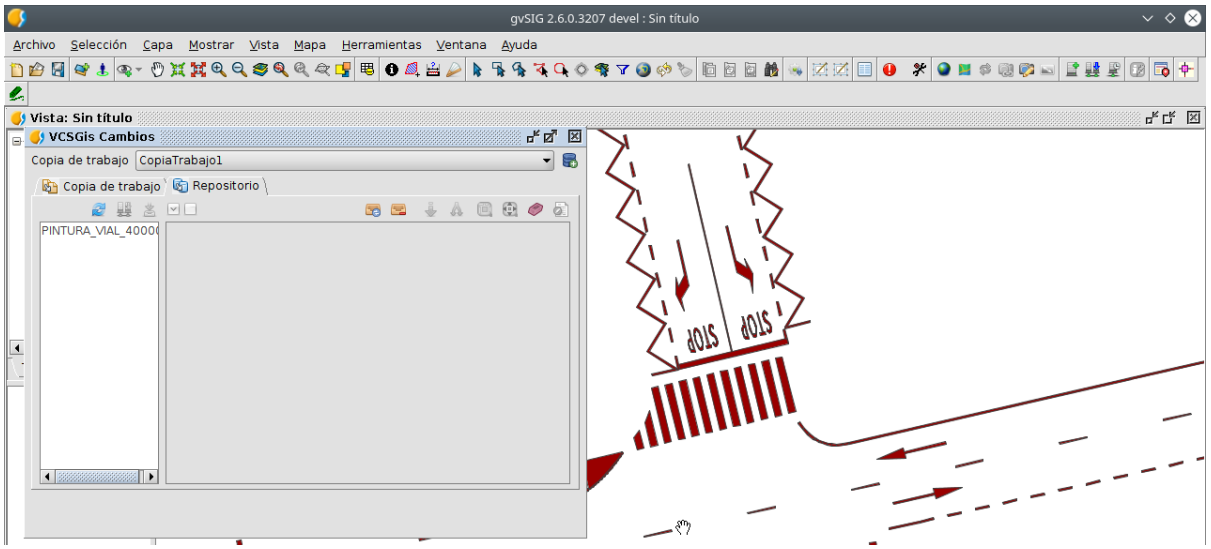
Una vez que tenemos una copia de trabajo con las tablas del repositorio que necesitemos ya podemos comenzar a trabajar. Los pasos típicos que se siguen cuando se trabaja el bajo control de versiones son:

- Actualizar la copia de trabajo (update).
- Realizar cambios. Es decir, trabajar con las capas normalmente.
- Revisar los cambios.
- Arreglar tus errores.
- Mezclar las capas con los cambios que se hayan realizado mientras trabajabas. Posiblemente haya que arreglar conflictos.
- Enviar los cambios al repositorio.

Y cada vez que se vuelve al trabajo, se repite este ciclo.

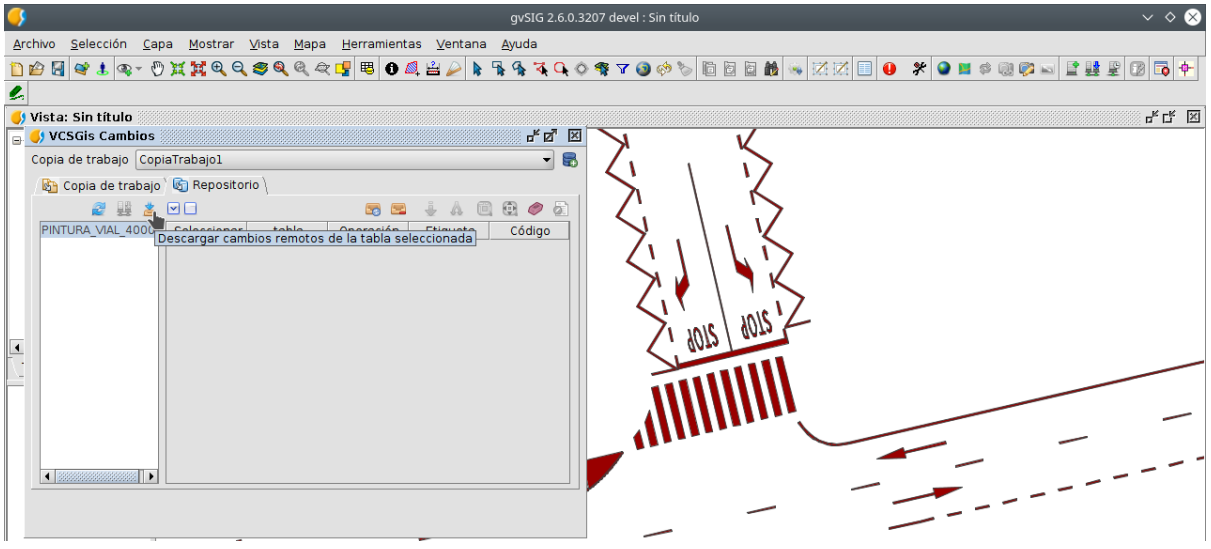
Actualizar la copia de trabajo

Para actualizar la copia de trabajo hay que ejecutar el comando *Mostrar cambios* y una vez en la ventana con el mismo nombre seleccionar la pestaña *Remoto*; quedando la ventana en cuestión como en la siguiente figura.



Si no existen cambios en el repositorio con respecto a la copia de trabajo, las capas de la lista de capas de la *Copia de trabajo* apareceran con una tipografía sin negrita en el componente 3 de la ventana (área con lista de capas de la *Copia de trabajo*) indicando que la *Copia de trabajo* está actualizada; pero si existen cambios, estas aparecerán en negrita estando la copia desactualizada.

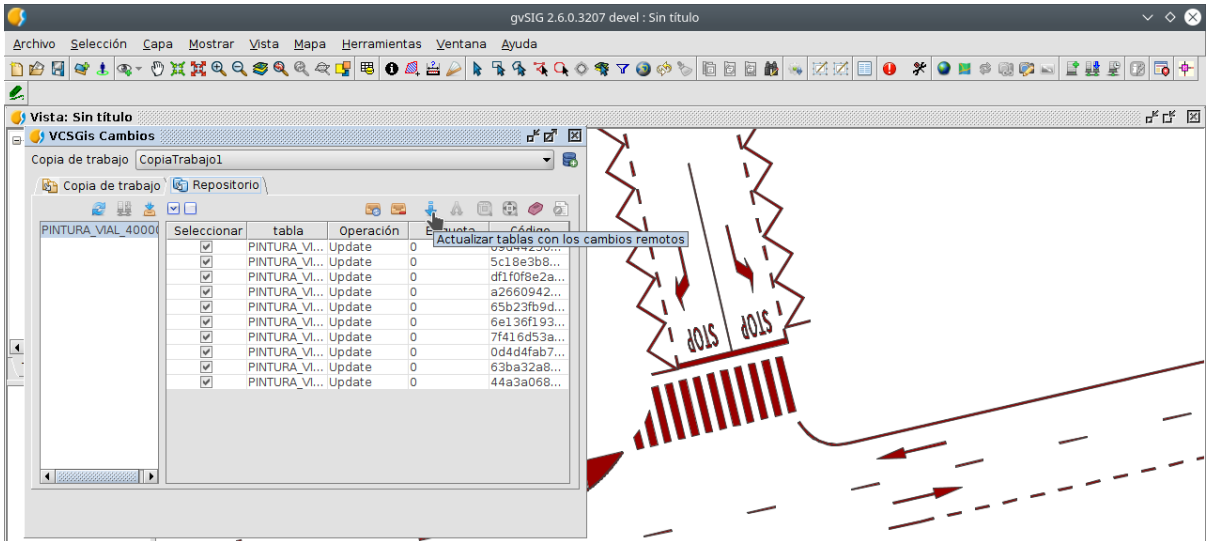
Para actualizarla hay que seleccionar la capa en negrita y pulsar el botón para descargar los cambios o diferencias del repositorio frente a la *Copia de trabajo*, componente 5 de la ventana.



La acción anterior provocará que los cambios se descarguen del repositorio y se muestren en el área de visualización de cambios, componente 6. Estos cambios aun no se habrán aplicado sobre las tablas del usuario. Para realizar esto deberán usarse los botones de **actualizar** (Update) o **mezclar** (Merge) cambios en la *Copia de trabajo*, componentes 10 y 11 de la ventana.

Note

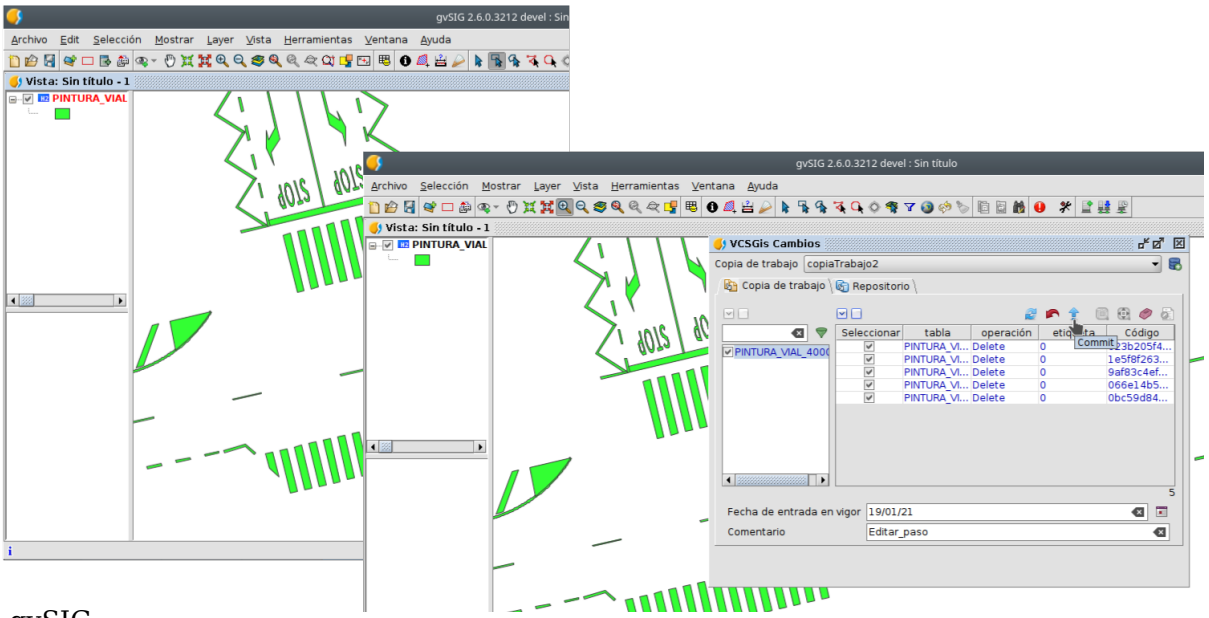
Actualizar (Update); Se actualizaran las tablas locales con los cambios del repositorio. Si hay cambios locales se perderán. Mezclar (Merge); Se mantendrán los cambios remotos para los registros seleccionados,y los cambios locales para los registros no seleccionados.



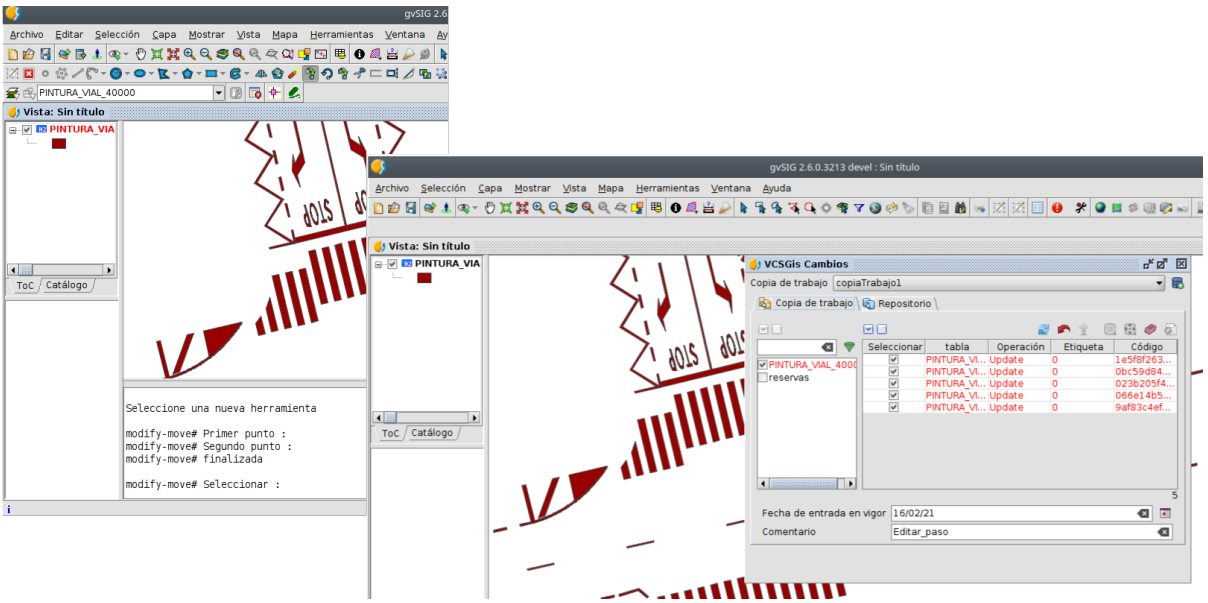
Revisando y enviando los cambios realizados

Para explicar mejor este comportamiento se va a usar el siguiente ejemplo:

Un usuario (Pedro) modifica una serie de elementos de una capa de su copia de trabajo (CopiaTrabajo2) llamada PINTURA_VIAL, tras terminar edición (eliminar elementos) ejecuta el comando *Mostrar cambios*. Como resultado al seleccionar la opción Local de la ventana Mostrar cambios, se muestran una serie de registros correspondientes a los citados cambios y solo tendría que pulsar el botón confirmación (Commit) para actualizar esos cambios de su copia de trabajo en el repositorio. Proceso que termina realizando.

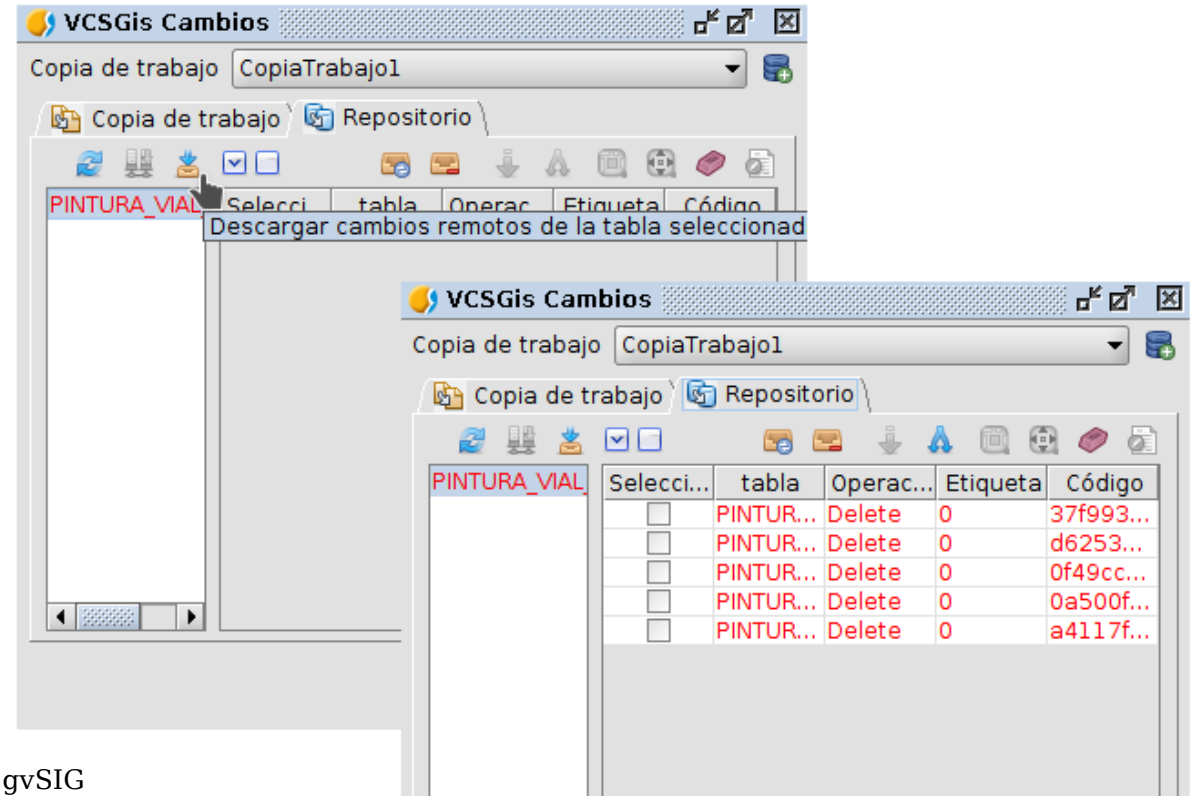


Paralelo al trabajo del usuario Pedro, otro usuario (Sara) trabaja en su copia de trabajo (CopiaTrabajo1) en la misma capa que Pedro y justamente en los mismos elementos que este modificando su posición. Tras terminar la edición procede a ejecutar el comando *Mostrar cambios* para subir sus cambios al repositorio, pero Pedro ya ha actualizado el repositorio con sus cambios previamente. De modo que el resultado que arroja la ventana de *Mostrar cambios* es el siguiente;

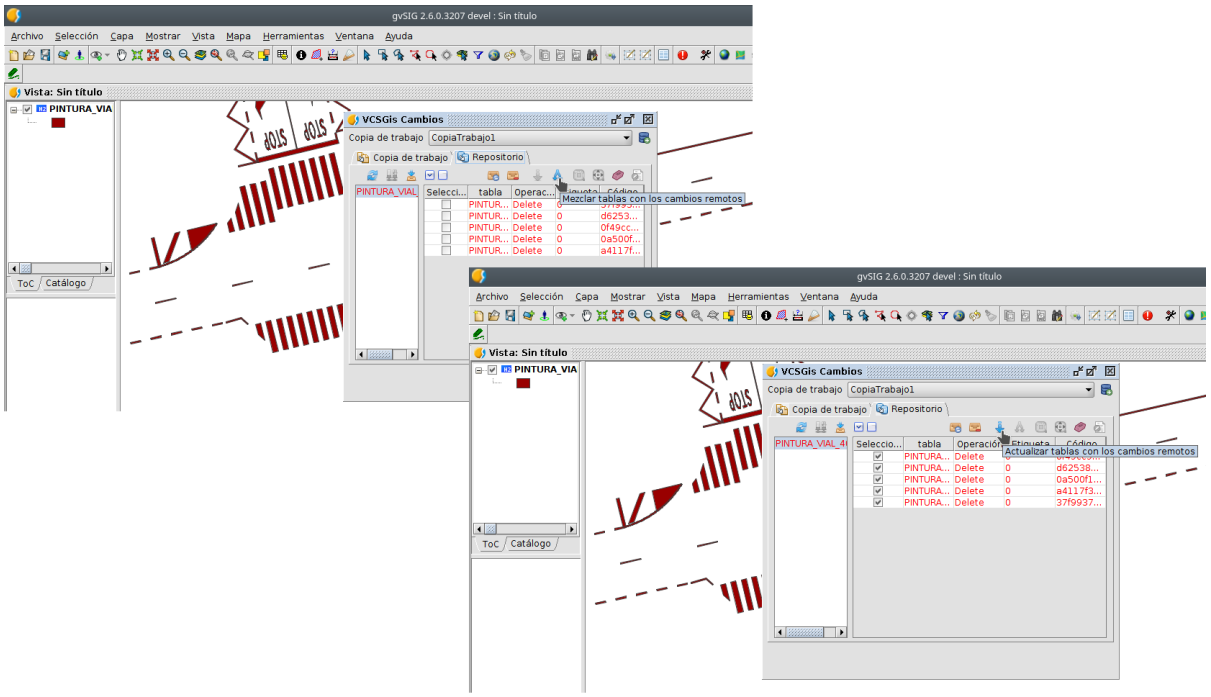


En primer lugar Sara tiene que actualizar la copia de trabajo con respecto al repositorio tal y como se indica en el apartado anterior y tras eso hacer un *Commit*, pero existe un problema sus cambios entran en conflicto con los de Pedro, de ahí el color rojo de estos. Si no entrara en conflicto tras la actualización de la copia de trabajo podría realizar el *Commit* sin problema.

Para solucionar el problema Sara tiene que hacer clic sobre la pestaña *Repositorio*, descargar los cambios del repositorio.

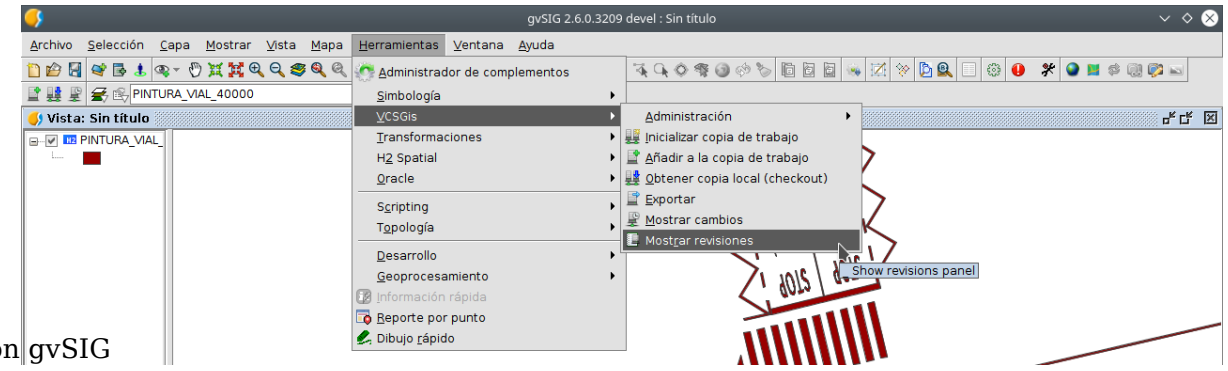


La ventana *Mostrar cambios* contiene ahora los cambios del repositorio, es decir los de Pedro y por tanto Sara tiene que decidir que cambios de este aceptar como buenos y cuales no, marcando con el check, mediante la opción mezclar, componente 11. Es importante destacar que también puede descartar los cambios de Pedro totalmente si realiza un *Merge* sin marcar ninguno de los cambios de la lista de cambios del repositorio o los suyos si marca todo los cambios del repositorio y realiza un *Merge* o si realiza simplemente un *Update*.

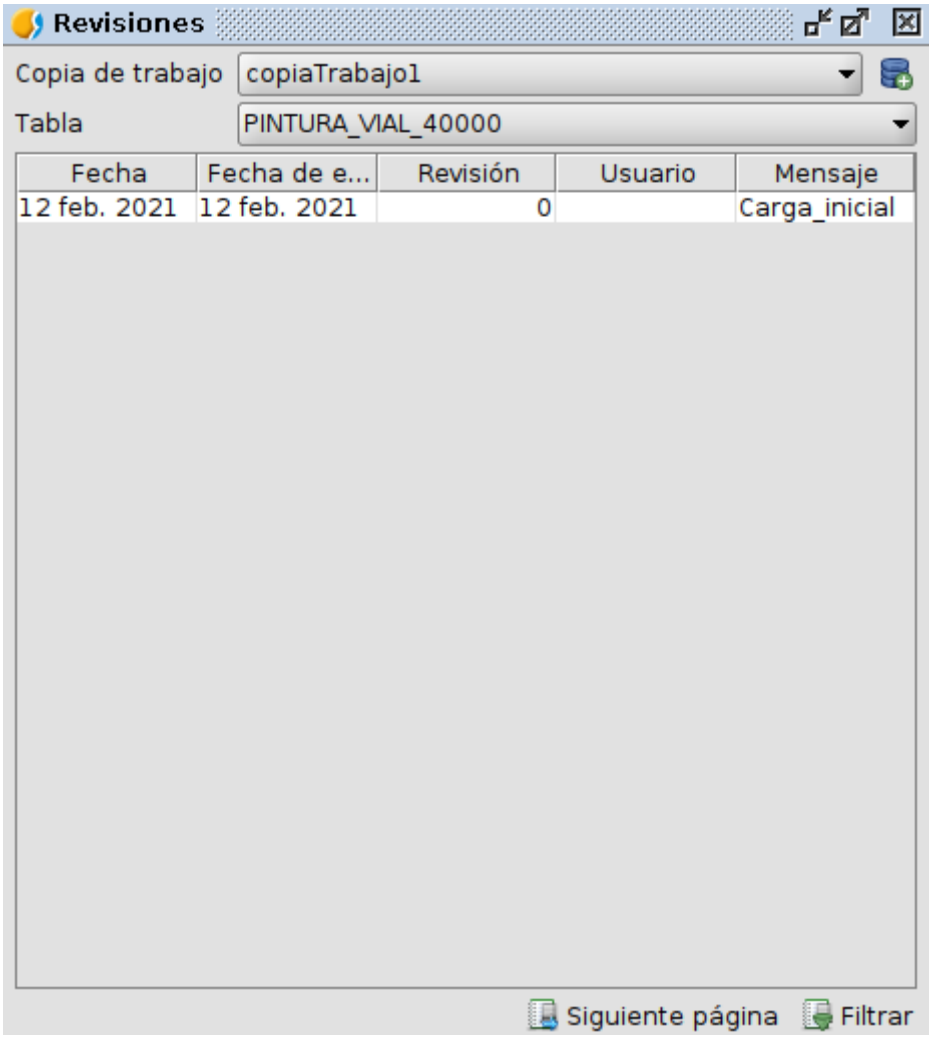


Gestión de las Revisiones

Tal y como se menciona al inicio del documento, las revisiones o cambios que sufren los datos en un software de control de versiones presentan tanta o más importancia que los propios datos. Haciendo hincapié en la idea anterior, la herramienta software VCSGis para el control de versiones dispone de un modulo para la correcta gestión de dicha información. Para la gestión de las revisiones ir a la opción *Herramientas* del menú de gvSIG Desktop, pestaña *VCSGgis* y ejecutar *Mostrar revisiones*.

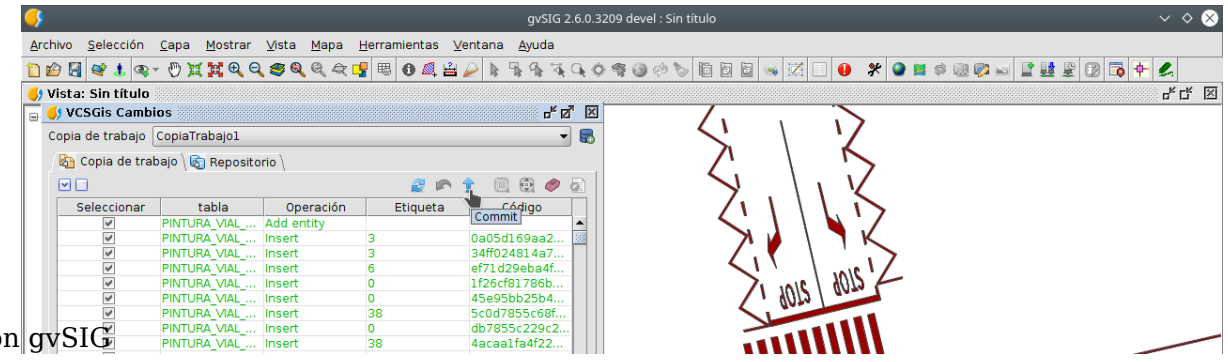


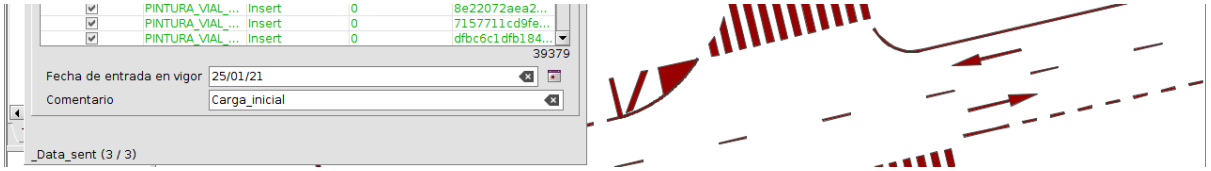
Como resultado de realizar la anterior acción se muestra la siguiente ilustración.



La ventana presenta dos elementos donde hay que seleccionar la *Copia de Trabajo* y la Tabla/Capa en cuestión. Una vez seleccionadas en la zona central de la ventana se muestran las revisiones que presenta el elemento seleccionado anteriormente en forma de tabla de datos. Para la gestión de las revisiones en dicha ventana se dispone en su zona inferior de dos iconos que permiten aumentar las revisiones presentes en la zona de visualización de esta o filtrarlas para realizar una búsqueda sobre estas.

A continuación se muestra en la siguiente composición un ejemplo de como se crea una revisión y como aparece esta en la ventana anteriormente definida.

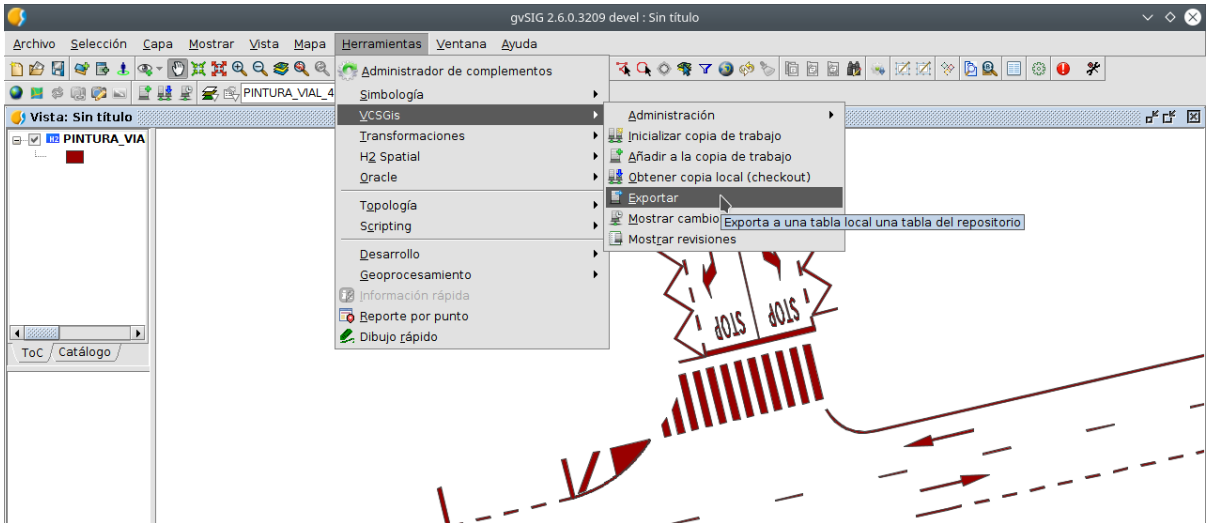




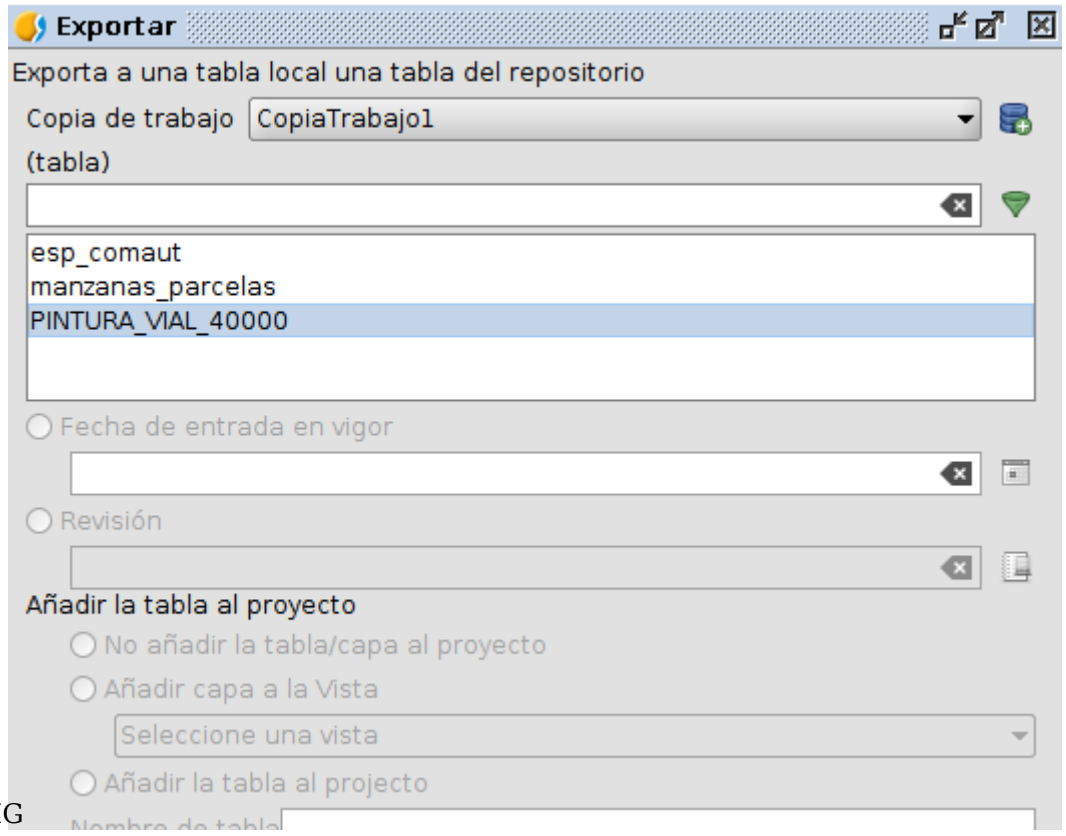
Exportar datos desde el repositorio

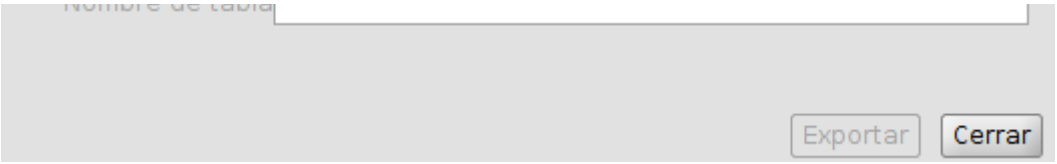
La herramienta de control de versiones VCSGis dispone de la posibilidad de exportar una capa/tabla de la copia de trabajo como un archivo. Este nuevo fichero posee la característica de ser un conjunto de datos normal, es decir, sin control alguno de sus cambios y por tanto de sus revisiones.

Para realizar el proceso anterior se tiene que ejecutar el comando *Exportar* situado en la opción *Herramientas* del menú de gvSIG Desktop, pestaña *VCSGgis*.



Tras la ejecución de comando se presenta la siguiente ventana.





El funcionamiento de esta es simple, en primer lugar hay que seleccionar la *Copia de Trabajo* de la cual obtener la capa que se busca exportar. Tras esto aparecerán en la ventana de capas la lista de capas de la copia en cuestión, debiendo seleccionar la deseada.

Concretado lo anterior solo queda indicar a que fecha se quiere la capa exportada. Para ello disponemos de dos opciones:

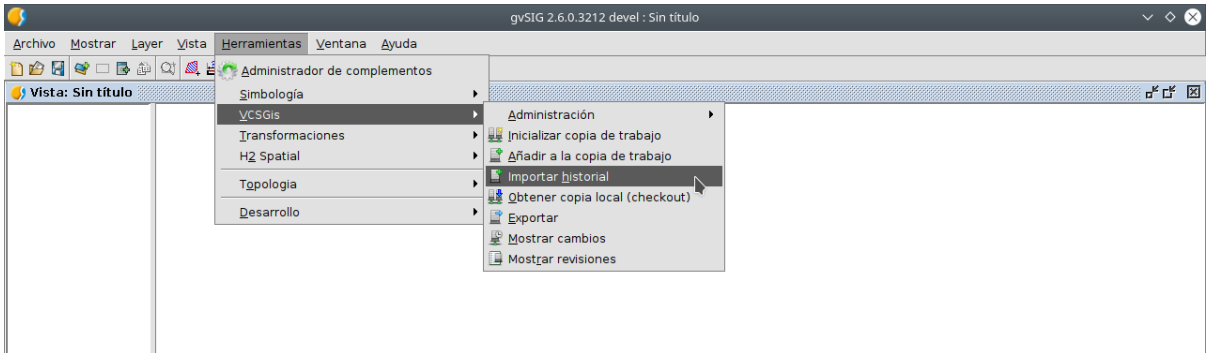
- Fecha de entrada en vigor. Esta opción proporciona un imagen de la cartografía a una determinada fecha indicada por el usuario.
- Revisión. Con esta opción se descarga la capa/tabla con los cambios presentes hasta la citada revisión. La gestión para la selección de revisión se realiza de igual manera que en apartado anterior.

Tras selección de la capa y la “fecha” de la información que queremos exportar de esta solo queda en la ventana una serie de parámetros que permiten la visualización y gestión de esta en las vistas.

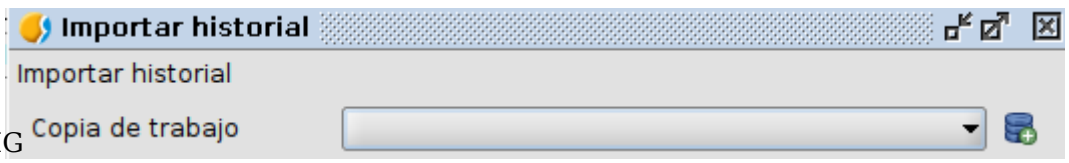
Importación o carga de historial de datos

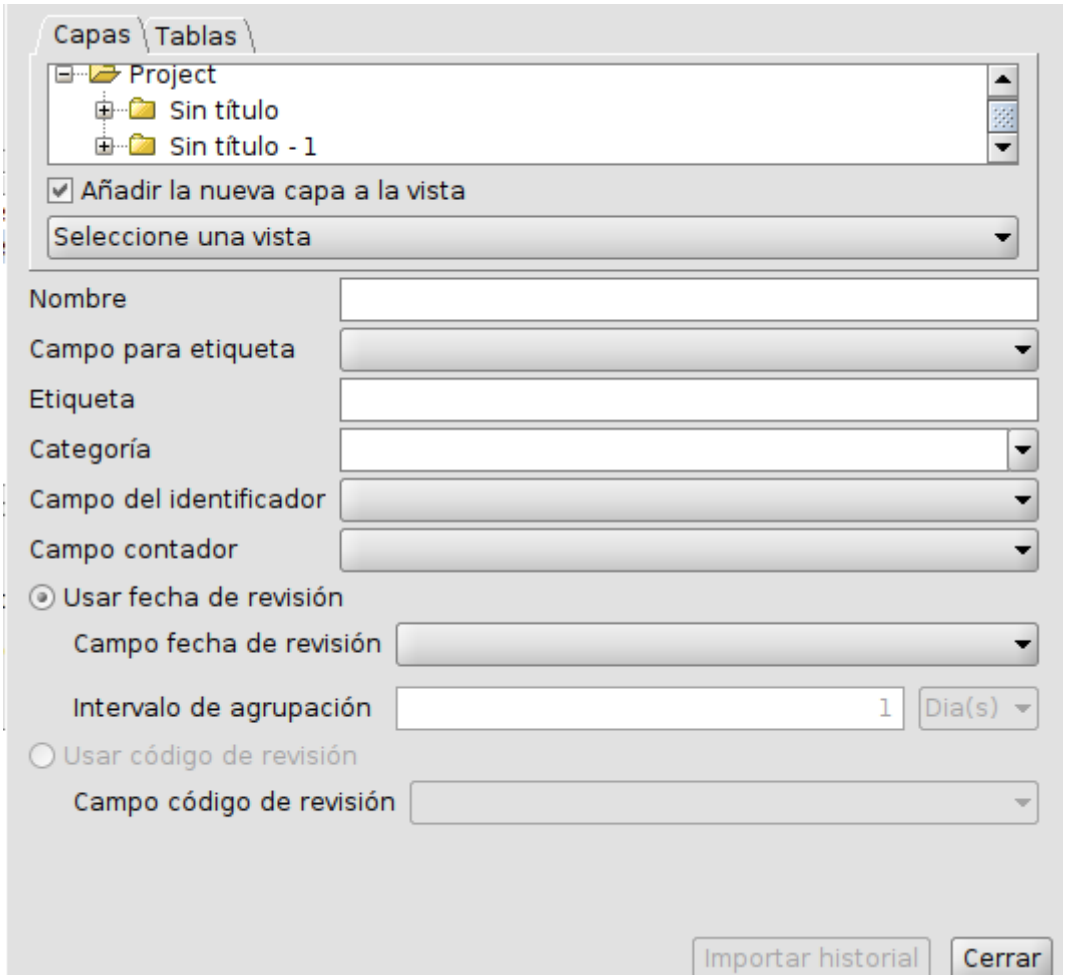
Otra funcionalidad presente en el control de versiones *VCSGis* de *gvSIG Desktop* es la de realizar el proceso de carga o importación de datos históricos siempre y cuando estos presenten un campo de carácter temporal que indique cualquier modificación de estado del elemento. Un ejemplo de esos estados sería su fecha de implantación, modificación , actualización, eliminación…

La opción *Importar historial* se encuentra en en la opción *Herramientas* del menú de gvSIG Desktop, pestaña *VCSGis*.



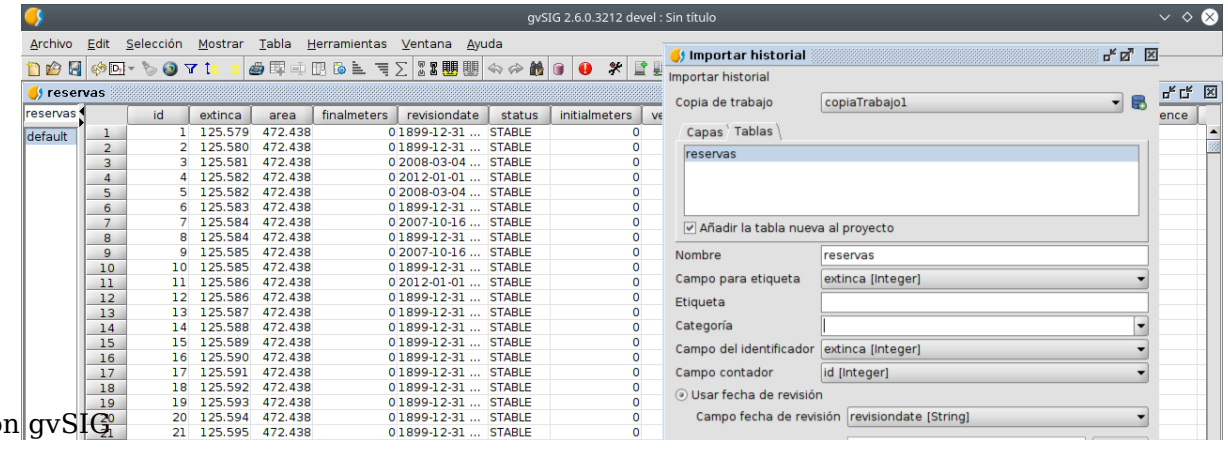
La ventana o interfaz resultante de ejecutar dicho comando se muestra en la siguiente ilustración.

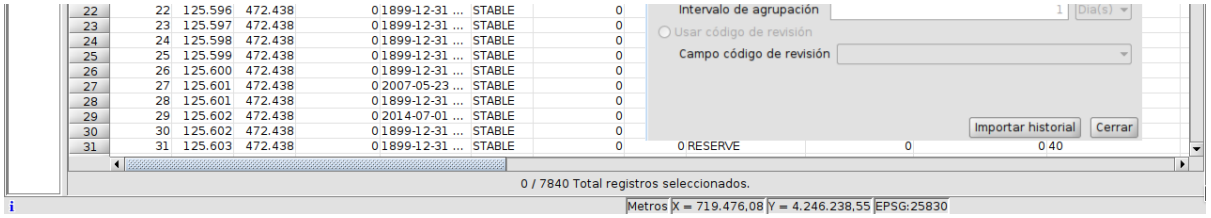




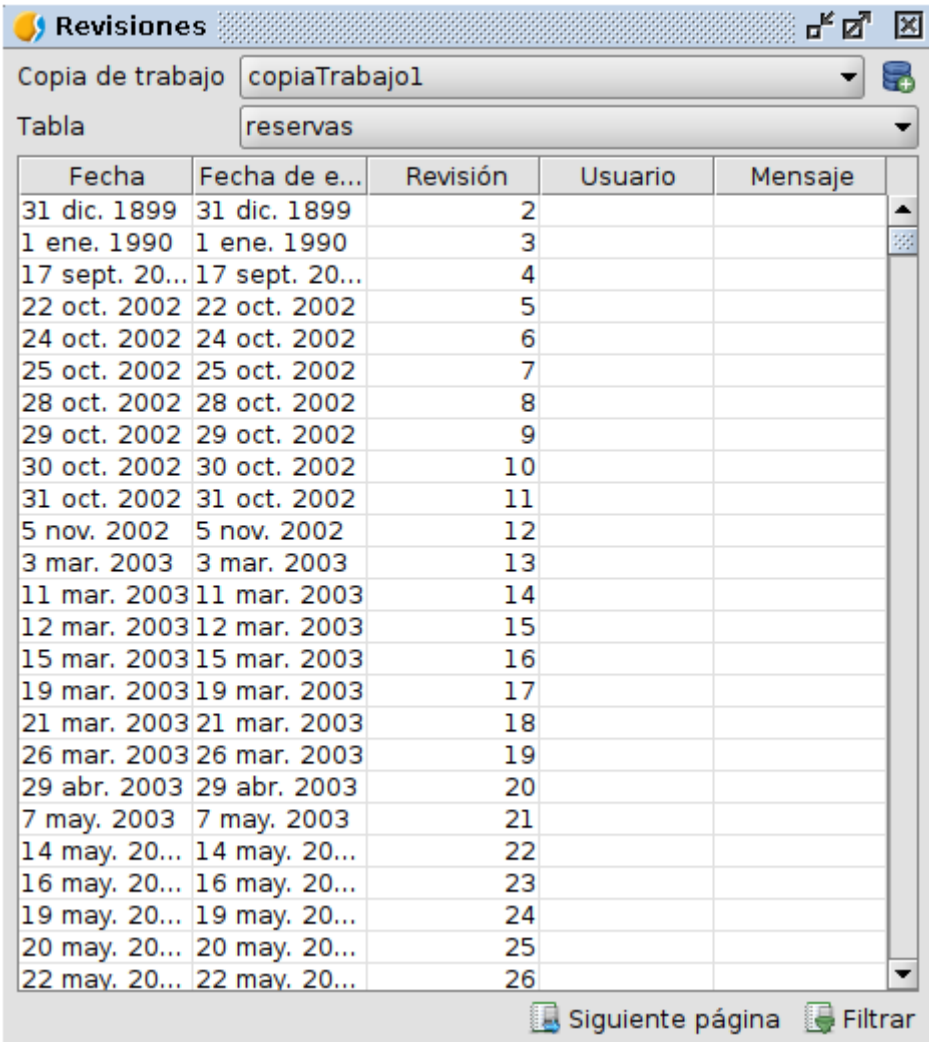
En dicha ventana hay que especificar la copia de trabajo donde se almacenará el histórico en cuestión mediante un desplegable. Una vez seleccionada la copia de trabajo en la ventana hay que identificar el archivo donde se encuentra el historial, el cual debe estar cargado de antemano en *gvSIG Desktop* ya sea como tabla o capa. Tras lo anterior, la ventana dispone de una serie de campos y desplegables a completar, de los cuales hay que destacar el campo identificador el cual hace referencia a la clave primaria de la tabla y los campos que referentes a la inclusión de los datos según fecha de revisión o código de revisión. Referente a estos campos últimos, existe la posibilidad de realizar las diferentes revisiones según fecha de revisión especificando un intervalo de agrupación o usando un código de revisión, previa identificación de este campo.

Como ejemplo para ilustrar el proceso anterior se muestra como la tabla *Reservas* de una base de datos ajena a *VCSGis* pasa a formar parte del control de versiones tras ejecutar el comando *Importar historial*.





Tras finalizar el proceso se puede comprobar que el proceso de carga fu correcto visualizando las diferentes revisiones creadas ejecutando el comando *Mostrar revisiones* dentro de la opción *Herramientas* del menú de *gvSIG Desktop*, pestaña *VCSGis*.



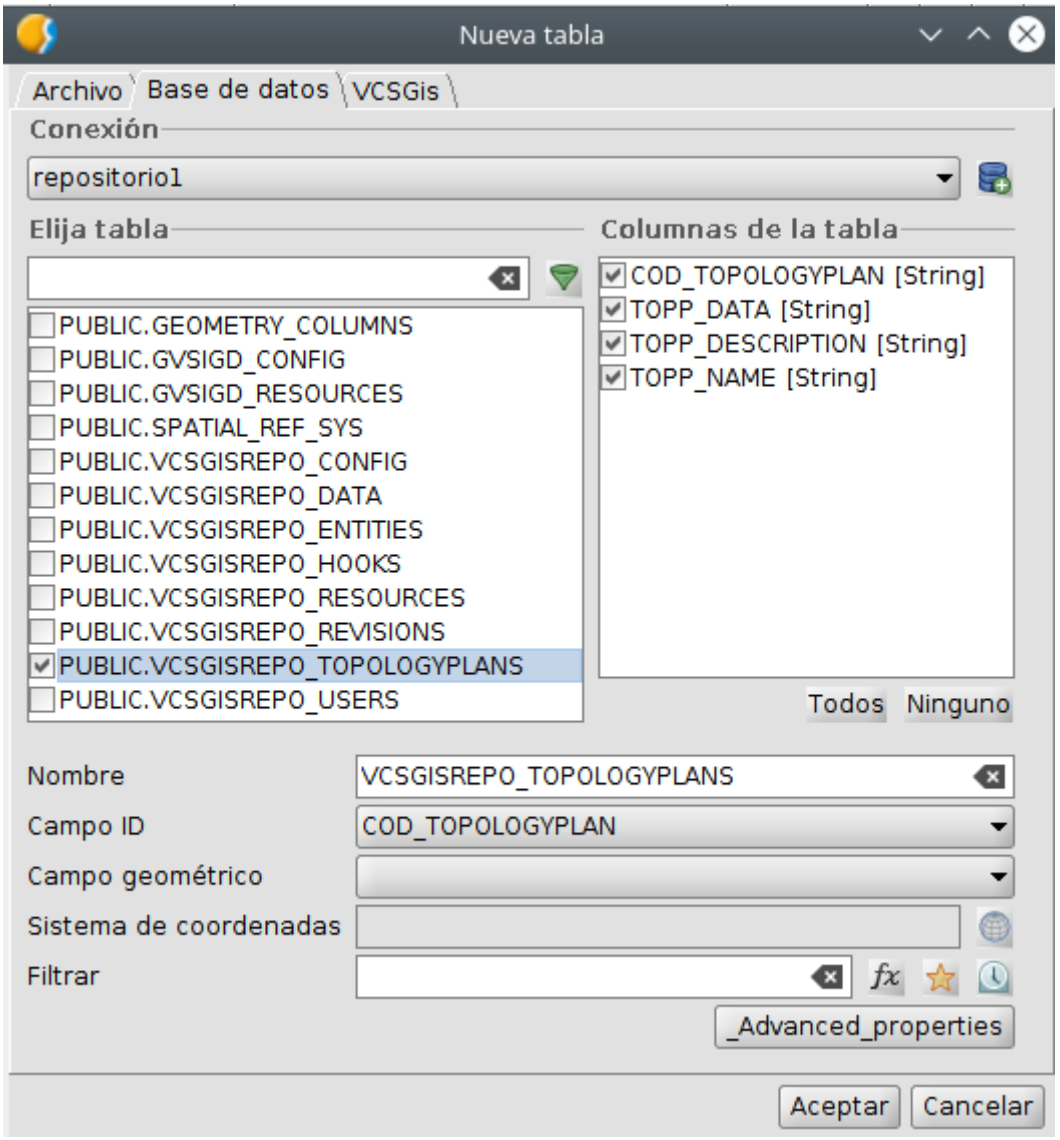
Fecha	Fecha de e...	Revisión	Usuario	Mensaje
31 dic. 1899	31 dic. 1899	2		
1 ene. 1990	1 ene. 1990	3		
17 sept. 20...	17 sept. 20...	4		
22 oct. 2002	22 oct. 2002	5		
24 oct. 2002	24 oct. 2002	6		
25 oct. 2002	25 oct. 2002	7		
28 oct. 2002	28 oct. 2002	8		
29 oct. 2002	29 oct. 2002	9		
30 oct. 2002	30 oct. 2002	10		
31 oct. 2002	31 oct. 2002	11		
5 nov. 2002	5 nov. 2002	12		
3 mar. 2003	3 mar. 2003	13		
11 mar. 2003	11 mar. 2003	14		
12 mar. 2003	12 mar. 2003	15		
15 mar. 2003	15 mar. 2003	16		
19 mar. 2003	19 mar. 2003	17		
21 mar. 2003	21 mar. 2003	18		
26 mar. 2003	26 mar. 2003	19		
29 abr. 2003	29 abr. 2003	20		
7 may. 2003	7 may. 2003	21		
14 may. 20...	14 may. 20...	22		
16 may. 20...	16 may. 20...	23		
19 may. 20...	19 may. 20...	24		
20 may. 20...	20 may. 20...	25		
22 may. 20...	22 may. 20...	26		

Topología en VCSGis

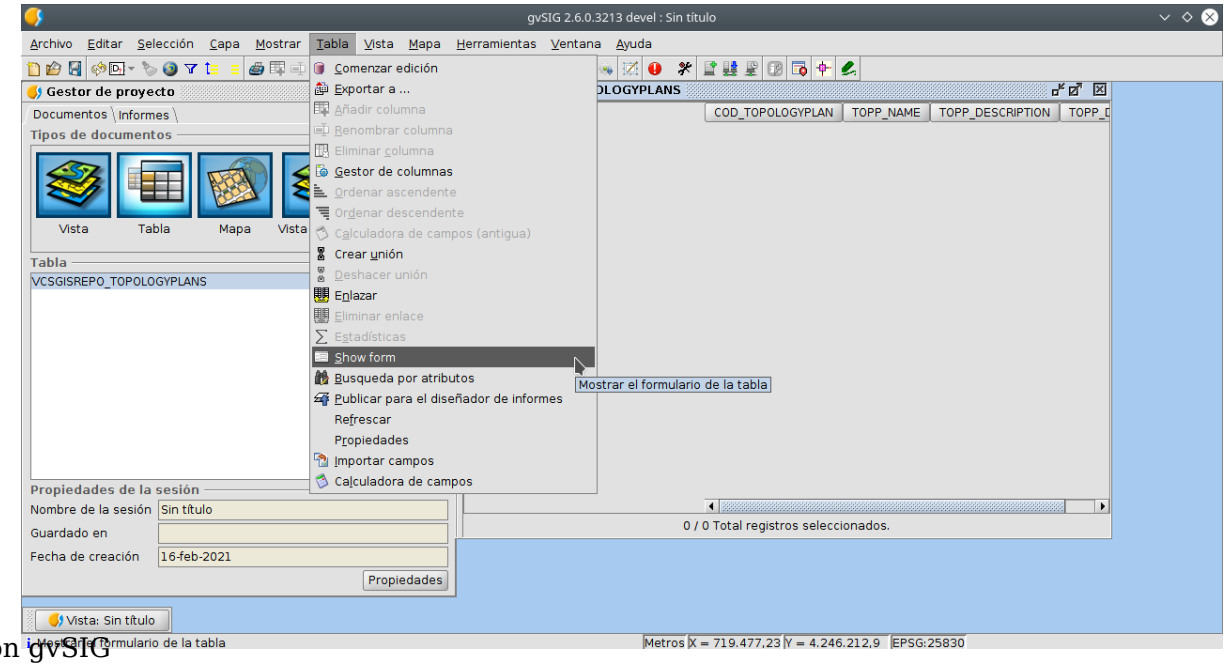
El sistema de control de versiones *VCSGis* de *gvSIG Desktop* permite incluir revisiones topológicas a aplicar a la información que este maneja si el usuario lo desea. En otras palabras, garantiza que la información gráfica gestionada por el control de versiones cumpla las reglas topológicas especificadas por el usuario. Para realizar lo anterior hay que configurar varios aspectos del repositorio como son la tabla de planes topológicos y la tabla de entidades.

Como se cita anteriormente la configuración topológica se realiza modificando varias tablas. En primer lugar hay que crear en la tabla de planes topológicos, **VCSGISREPO_TOPOLOGYPLANS**, una nueva entrada. La tabla originalmente

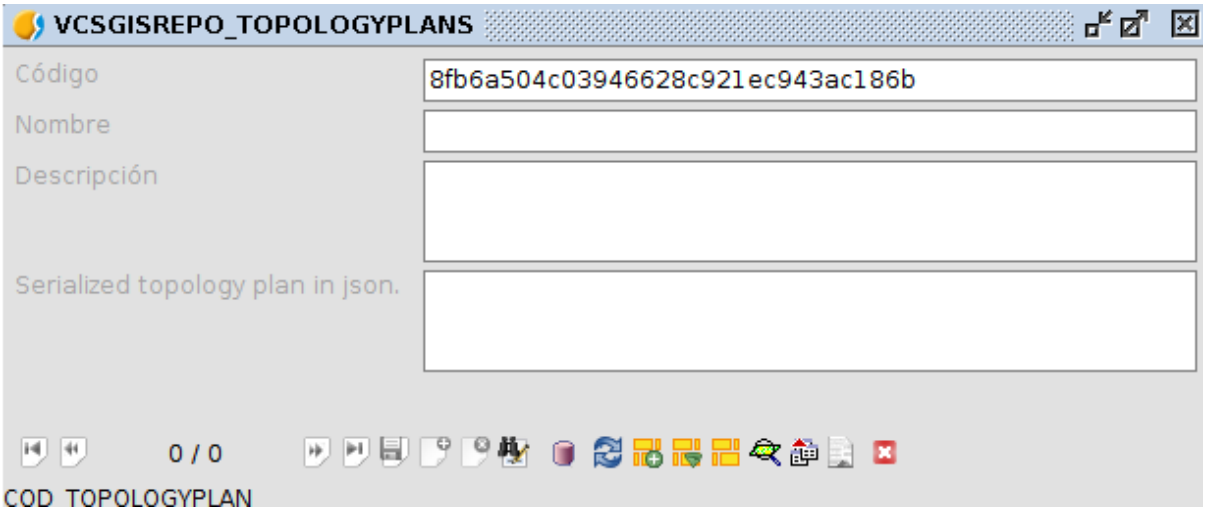
aparece vacía y sin ningún elemento. Para acceder a ella hay que abrir la tabla desde el gestor de proyectos.



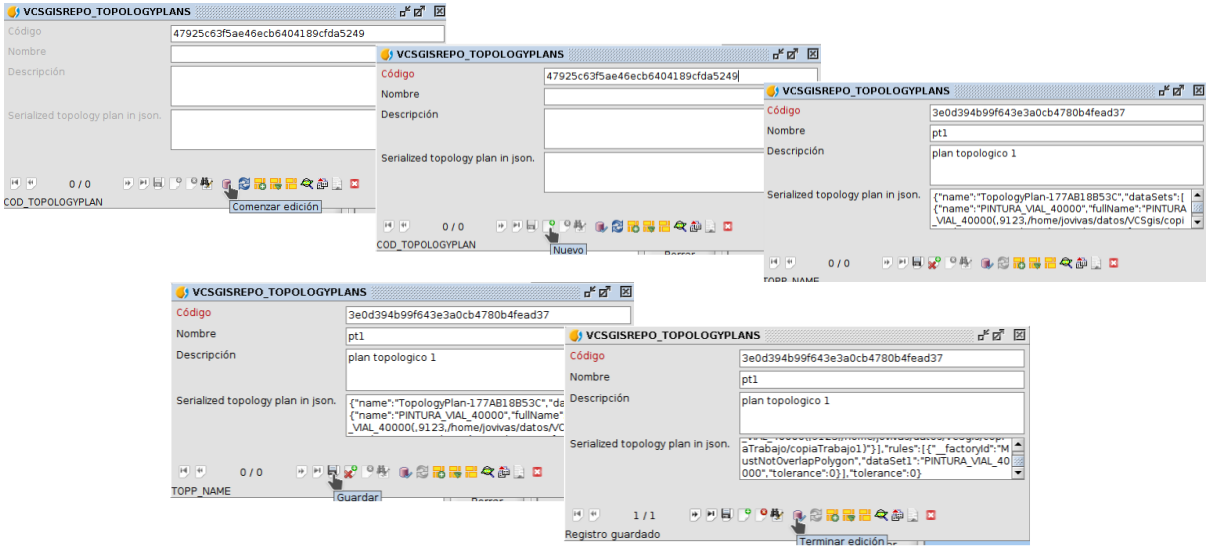
Tras haber accedido a ella hay que ejecutar el comando Show form, el cual nos permite de forma cómoda crear un nuevo plan.



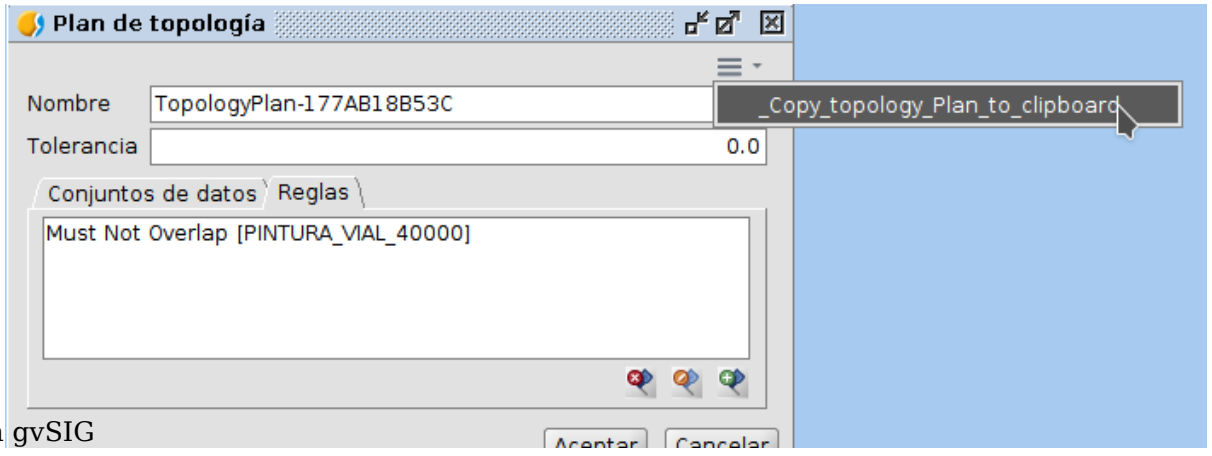
Como resultado de ejecutar el comando anterior se obtiene el siguiente formulario.



Para crear un nuevo plan solo tendremos que poner la tabla en edición, crear un nuevo plan, rellenar la información del plan (Código, Nombre, Descripción y plan topológico en JSON), guardar dicho plan y terminar la edición de la tabla. Dicho proceso se detalla en la siguiente ilustración.

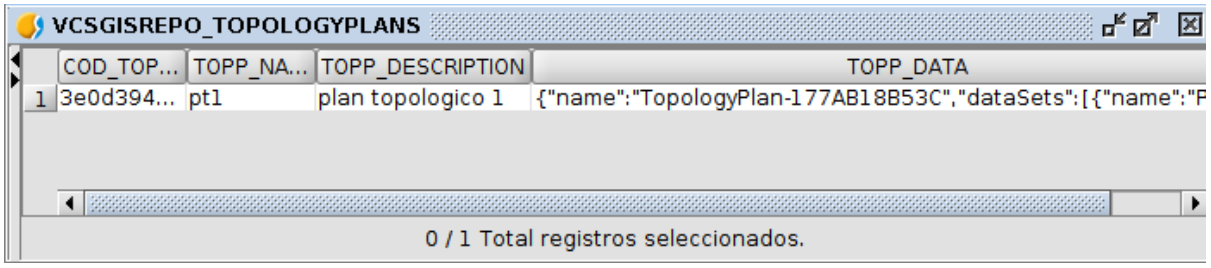


Para obtener la información sobre el plan topológico en formato JSON hay que crear este previamente mediante el plugin de *Topologia* de *gvSIG Desktop*. Este se puede obtener en ese formato solo al pulsar el botón de opciones situado en la esquina superior derecha y seleccionar la opción *Copiar plan topológico al portapapeles*.

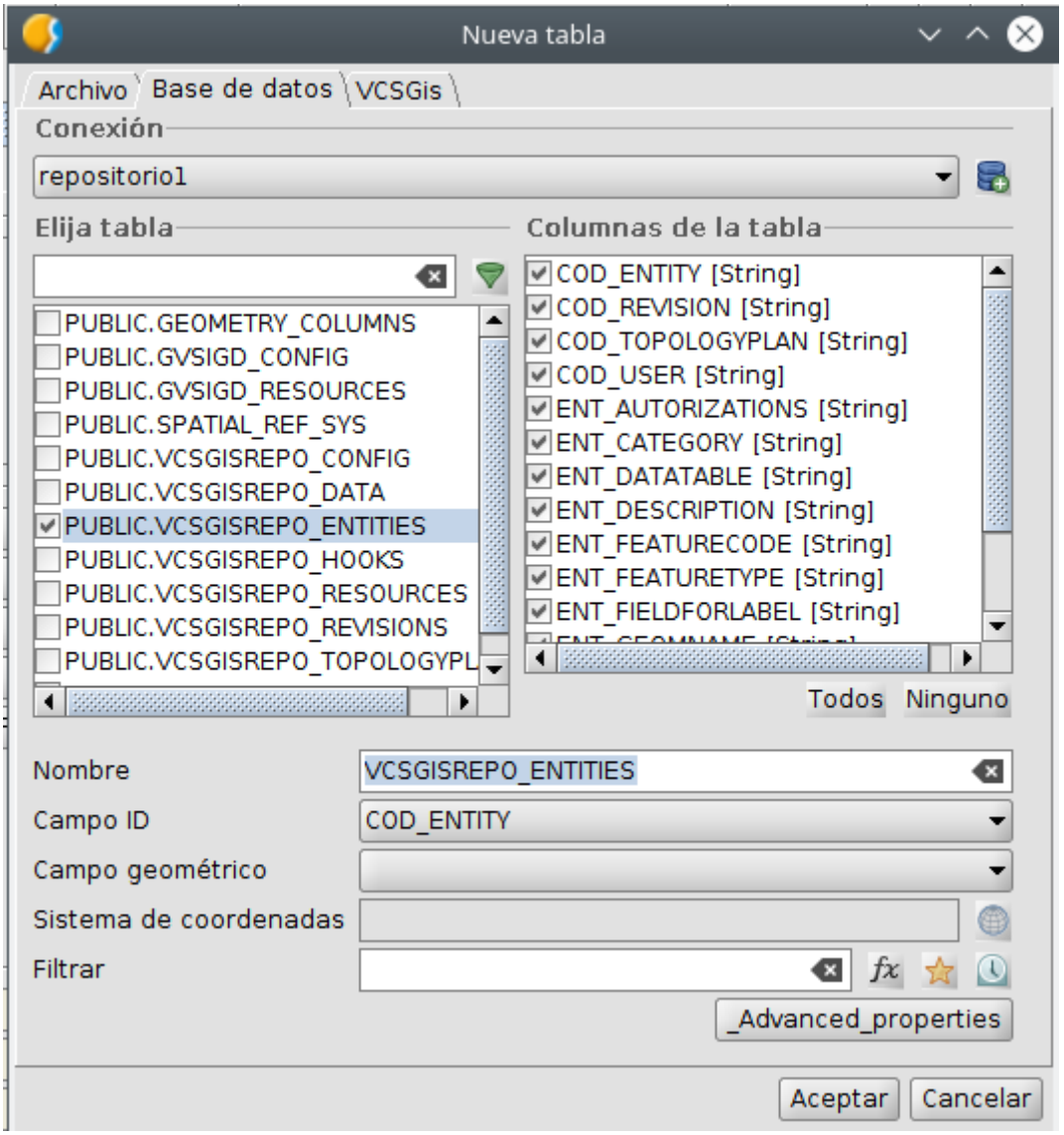


Esa cadena de carácteres es la información a incluir en la definición de un plan topológico de la tabla **VCSGISREPO_TOPOLOGYPLANS**.

La siguiente ilustración muestra la tabla con el registro correspondiente al plan topológico creado en el ejemplo.

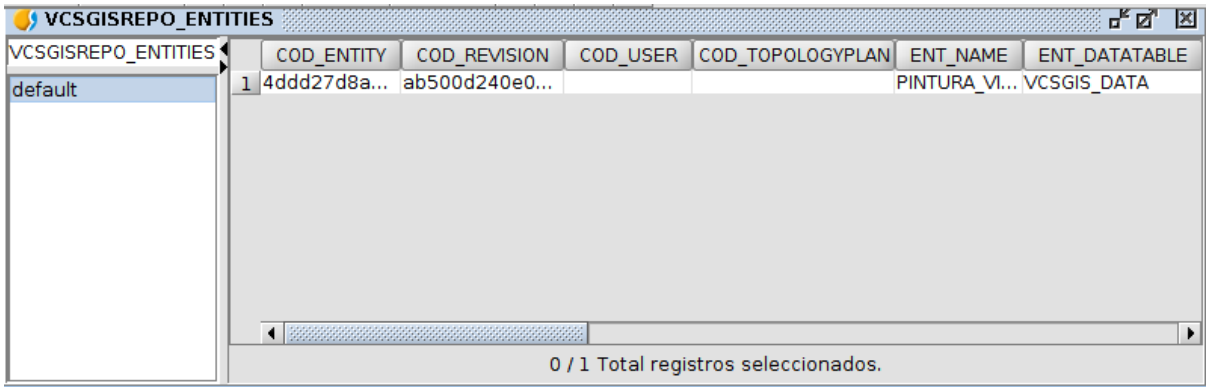


Tras la creación de una nueva entrada en la tabla de planes topológicos, solo queda asignar ese plan a las tablas del repositorio sobre las cuales se busca que cumplan topología. Para realizar lo anterior hay que abrir la tabla **VCSGISREPO_ENTITIES** donde se almacenan las capas/tablas disponibles en el repositorio. El acceso a esta tabla se realiza de igual manera que la anterior.

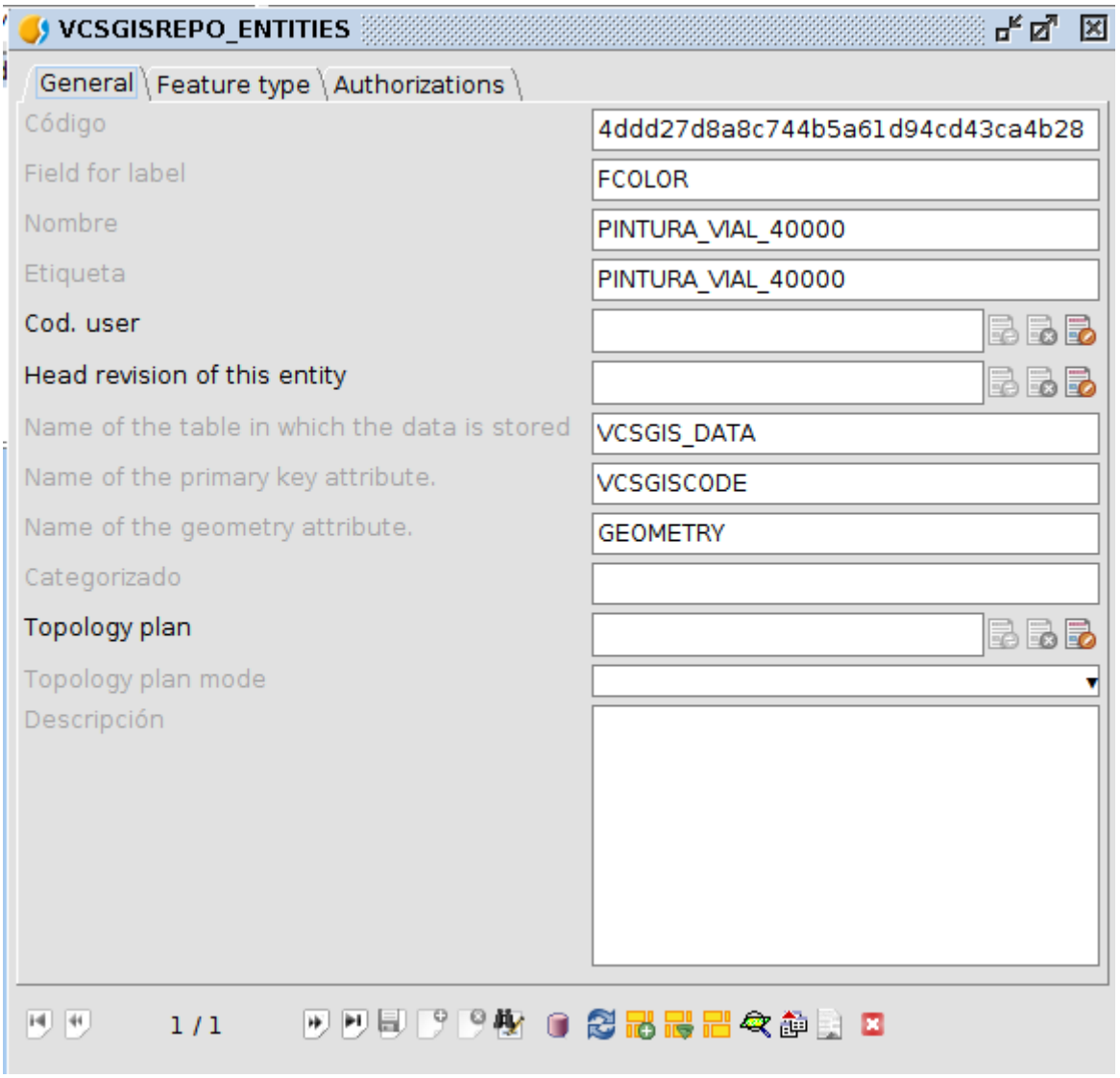


En la siguiente imagen se muestra que la tabla de entidades del repositorio ejemplo solo almacena una capa. Com se puede apreciar no presenta ningún plan

de topología asociado.



Una vez allí hay que obtener su formulario asociado al igual que con la tabla de planes topológicos mediante ella ejecución del comando *Show form*. Una vez desplegado, en el formulario aparecen todas las tablas disponibles del repositorio, las cuales mediante edición permite asignar un plan de topología definidos anteriormente en la tabla correspondiente.



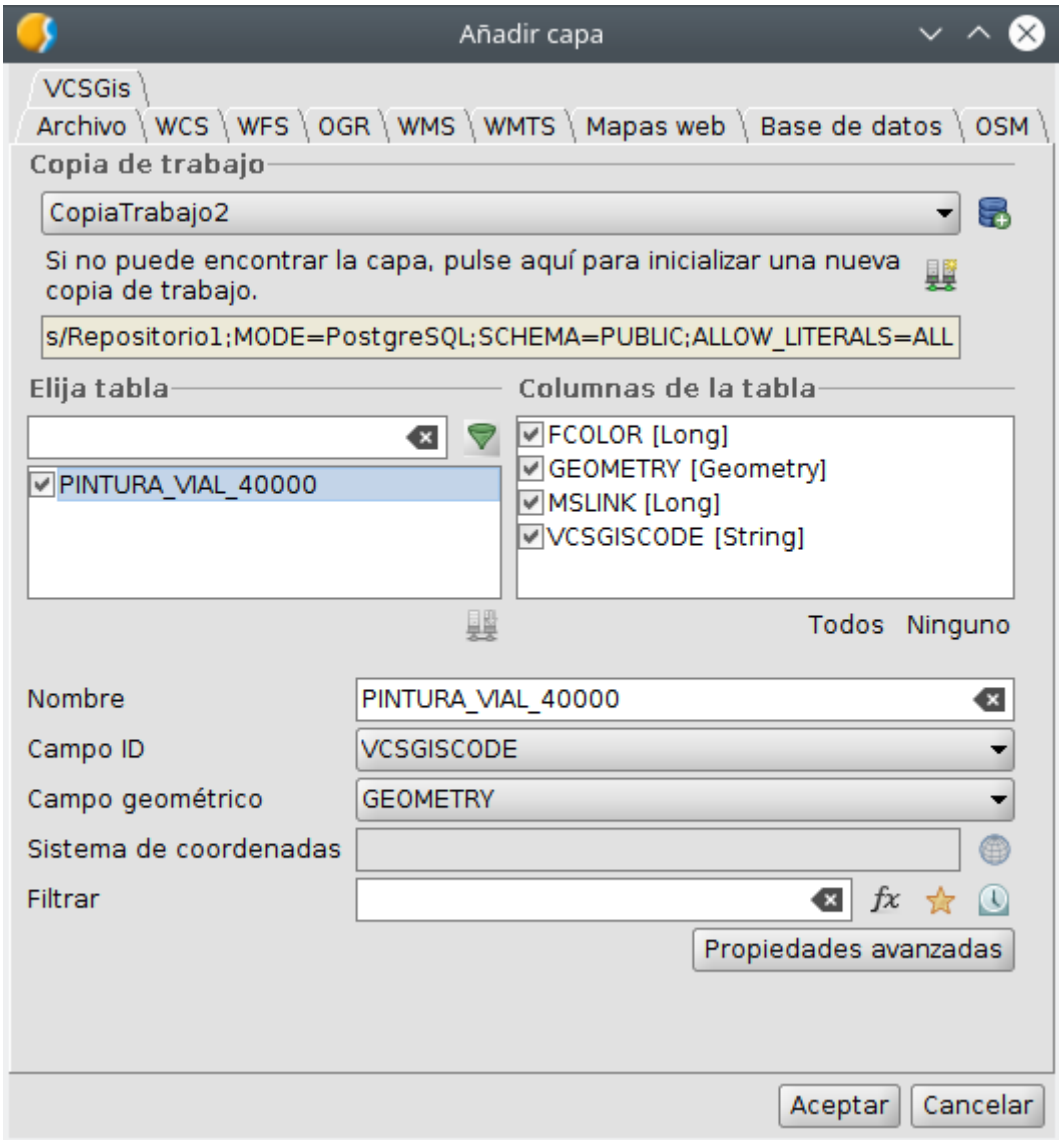
Comenzar a editar el formulario se realiza de la misma manera que en el formulario de planes topológicos. Durate la edición de la entidad hay que asignar el plan correspondiente a la tabla en la pestaña *General* desplegable *Plan Topológico*.

Añadir capa VCSGis usando el diálogo “Añadir Capa” de gvSIG Desktop

Una vez en la ventana de añadir capa existen 2 maneras de proceder para cargar un capa/tabla con control de versiones VCSGis;

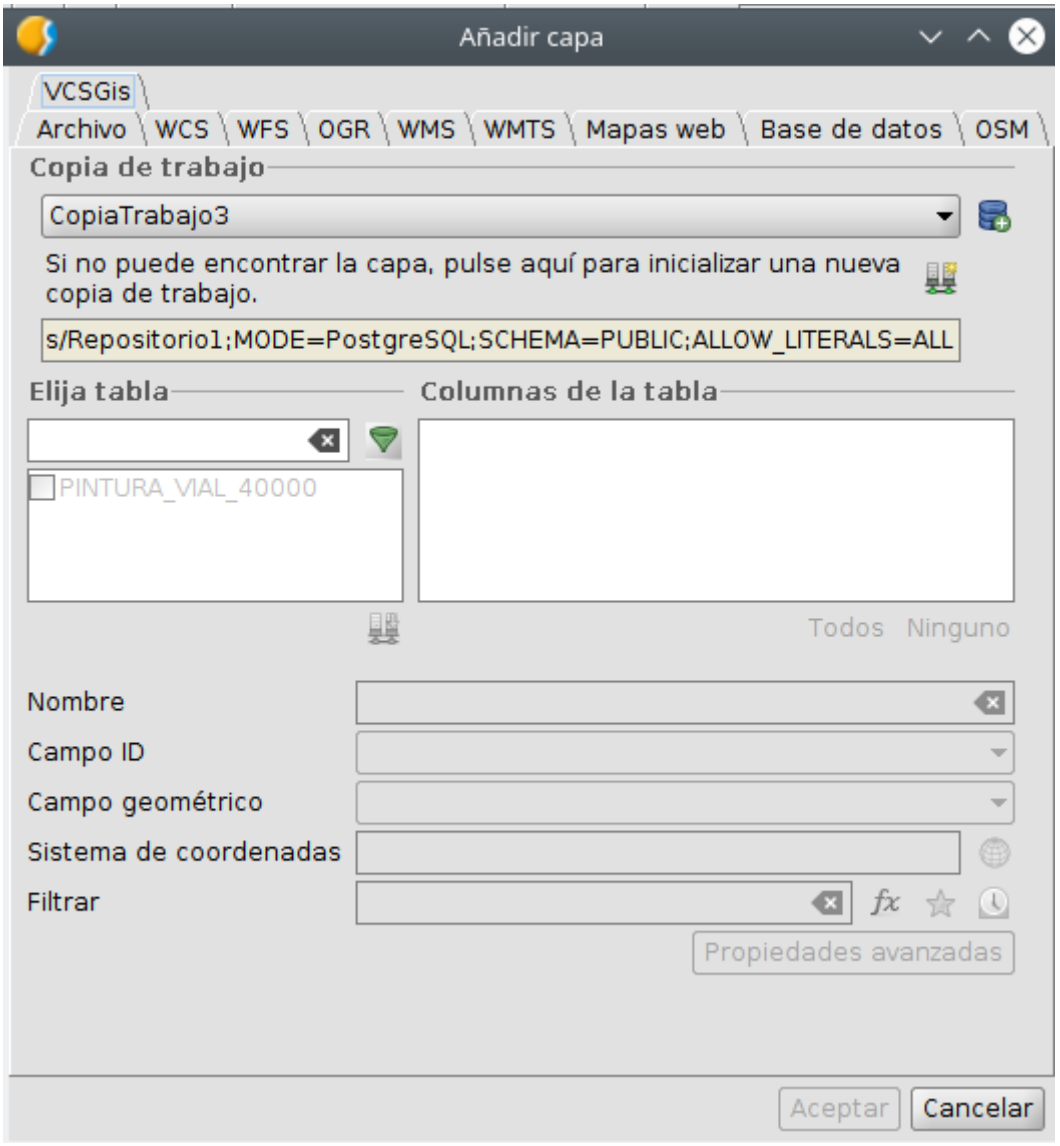
- Seleccionar la capa como si fuera un archivo de base de datos genérico utilizando la conexión a la base de datos asociada a la *Copia de trabajo*.
- Utilizar la pestaña VCSGis que permite cargar de manera automática este tipo de archivos con control de versiones. Opción más lógica.

A continuación, se muestra el aspecto de dicha ventana al seleccionar la pestaña *VCSGis* y cargar una capa la cual ya estaba en local pero que fue eliminada de la vista.

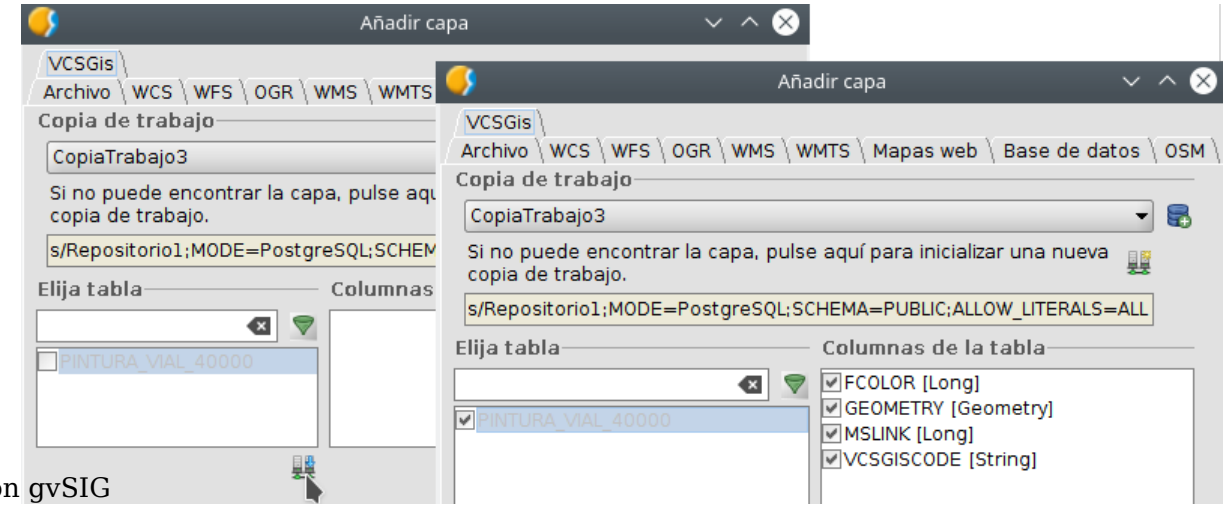


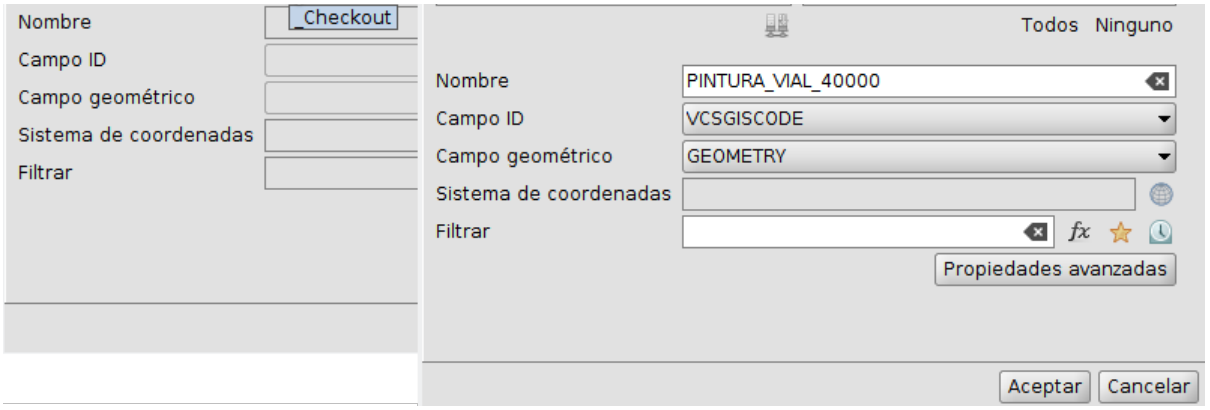
Como se puede apreciar hay que identificar la copia de trabajo y tras eso seleccionar (marcar el check) en el área de elección de tablas la tabla/capa en cuestión. A parte de lo anterior, es importante cumplimentar la información de campos de información presentes.

En el caso de que la copia de trabajo nunca halla presentado la capa que se pretende cargar, la ventana muestra un comportamiento diferentes. Ver siguiente ilustración.



Se puede ver que la capa aparece deshabilitada, para poder cargarla hay que seleccionarla, lo cual habilita el icono que obtiene la copia local (checkout). Tras ejecutar dicho proceso al pulsar dicho icono la ventana presenta el mismo aspecto que si la copia local hubiera tenido en algún momento dicha tabla/capa. Solo queda pulsar *Aceptar* para terminar el proceso.





Usuarios y permisos

La herramienta de control de versiones *VCSGis* de *gvSIG Desktop* incluye un mecanismo de gestión de usuarios básico que le permite discriminar a que recursos tiene acceso un determinado usuario. Este mecanismos se basa en dos sistemas, uno de autenticación y otro de autorización.

Autenticación consiste en un sistema para certificar que el usuario es quien dice ser; lo más común es utilizar una combinación de identificador de usuario único y contraseña, aunque existen otros. *VCSGis* utiliza este sistema de *usuario+contraseña*.

Autorización consiste en dar acceso a una serie de operaciones o recursos a un usuario. Por ejemplo, si un usuario puedo hacer una operación de *commit* o *checkout*, o no la puede hacer, o si puedo acceder a una determinada tabla, solo para hacer *checkout* o tambien para *commit*.

En un repositorio de *VCSGis* podremos activar el mecanismo de *autenticacion*, o el mecanismo de *autenticacion mas autorización*. Si activamos la *autenticación*, garantizaremos que solo los usuario que podemos certificar quienes son, pueden acceder al sistema. Ahora bien, solo con este mecanismo, una vez accedan al repositorio, no tendremos control de que operaciones pueden hacer o a que recursos puede acceder. Si activamos también la *autorización*, podremos especificar tanto las operaciones que puede realizar un usuario, como a que recursos puede acceder.

Por defecto, cuando se crea un *repositorio* de *VCSGis*, este no tiene activados ninguno de estos mecanismos. y tendremos que activarlos manualmente.

Ademas, tendremos que tener en cuenta, que no existen herramientas específicas para realizar las tareas de administración de *VCSGis* relacionadas con la gestión de usuario. Para ello el **administrador** debera acceder a las tablas de configuración del repositorio y realizar la configuración necesaria manipulando estas directamente.

En los siguientes apartados iremos viendo como se pueden llevar a cabo estas tareas de administración.

El mecanismo de control de gestion de usuario de VCSGis podemos verlo a tres niveles:

- Requiriendo solo autenticación de los usuario.
- Requiriendo autenticación y con una autorización básica o por operación.
- Requiriendo autenticación y con una autorización avanzada o por entidad.

Autenticación

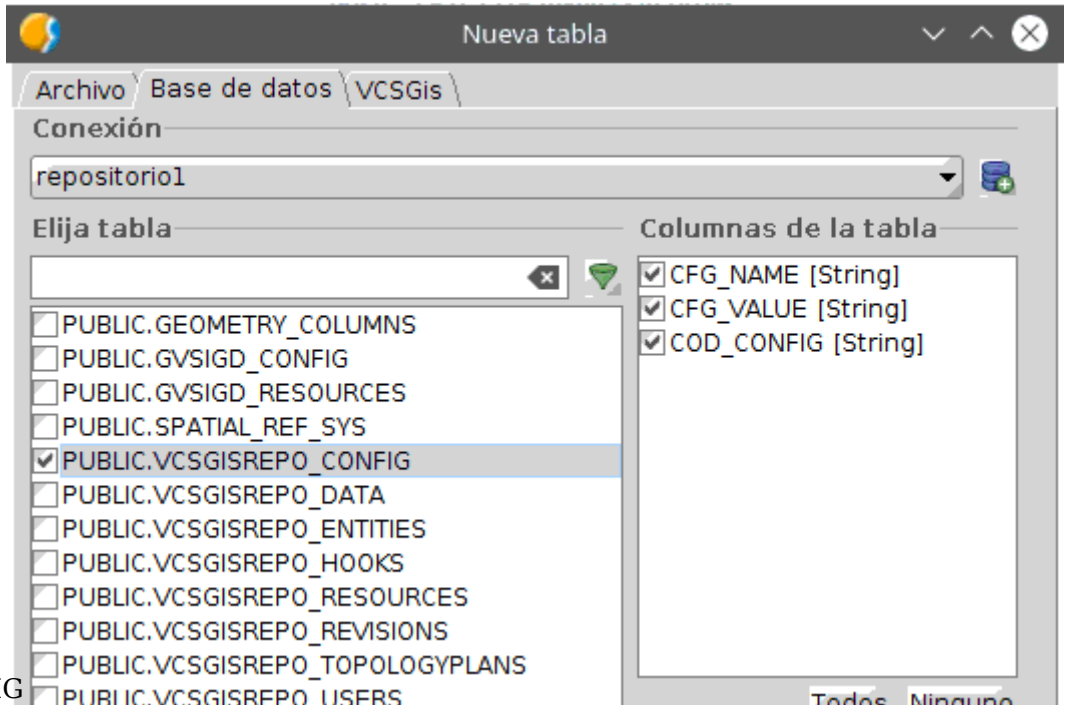
Este nivel se basa en autorizar el acceso a determinada información mediante credenciales, usuario y contraseña concretamente. Si se activa la autenticacion, se garantiza que solo los usuario que puedan certificar su identidad tienen acceso al sistema. Ahora bien, solo con este mecanismo, una vez accedan al repositorio, no se dispone de control de que operaciones pueden realizar o a que recursos puede acceder. Para controlar lo anterior hay que aplicar los niveles de autorización básica y avanzada respectivamente.

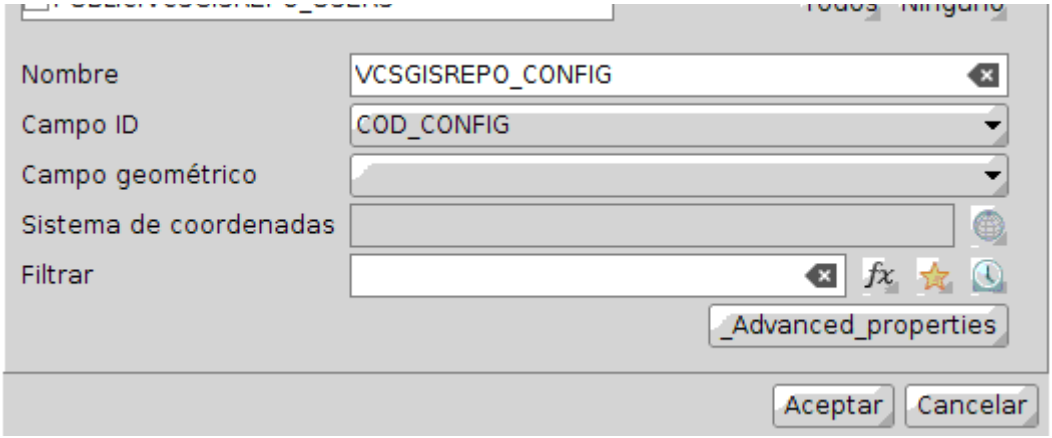
De lo anterior se puede deducir que la autenticación es el nivel de seguridad menos restrictivo y a medida que se implementa la autorización básica y la autorización avanzada el nivel de seguridad se vuelve más restrictivo y por tanto la seguridad del repositorio aumenta.

Dotar el nivel de autenticación a un repositorio se realiza mediante la configuración de dos tablas de dicho repositorio, tabla *PUBLIC.VCSGISREPO_CONFIG* y la tabla *PUBLIC.VCSGISREPO_USERS*.

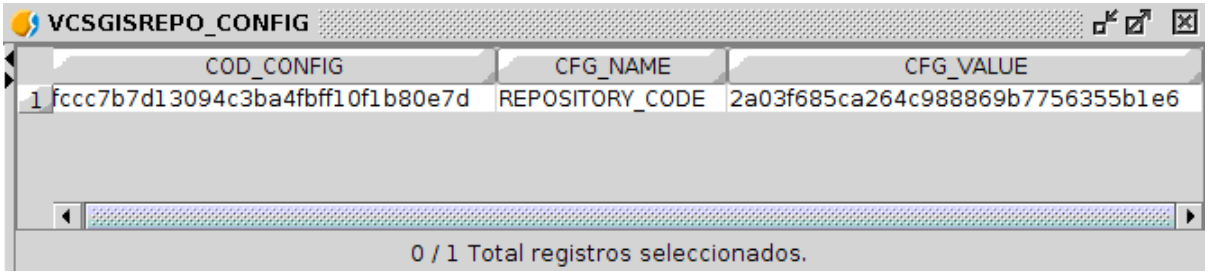
Para abrir las tablas hay que realizarlo desde el *Gestor de proyectos* situado en el menú *Mostrar* de *gvSIG Desktop*. El proceso de abrir una tabla es el genérico a abrir cualquier archivo, primero se selecciona *Tabla* como tipo de datos a abrir, se selecciona la opción de *Nuevo*, lo que habilita una ventana donde se tiene que seleccionar la pestaña *Base de datos*. Esa pestaña muestra en su zona superior un desplegable donde hay que especificar la base de datos donde se encuentra la tabla. Una vez seleccionada la base de datos, en la lista de tablas de esta hay que marcar la tabla en cuestión y pulsa el botón *Aceptar*.

A continuación se muestra la ventana con la que se abre la tabla *PUBLIC.VCSGISREPO_CONFIG*.

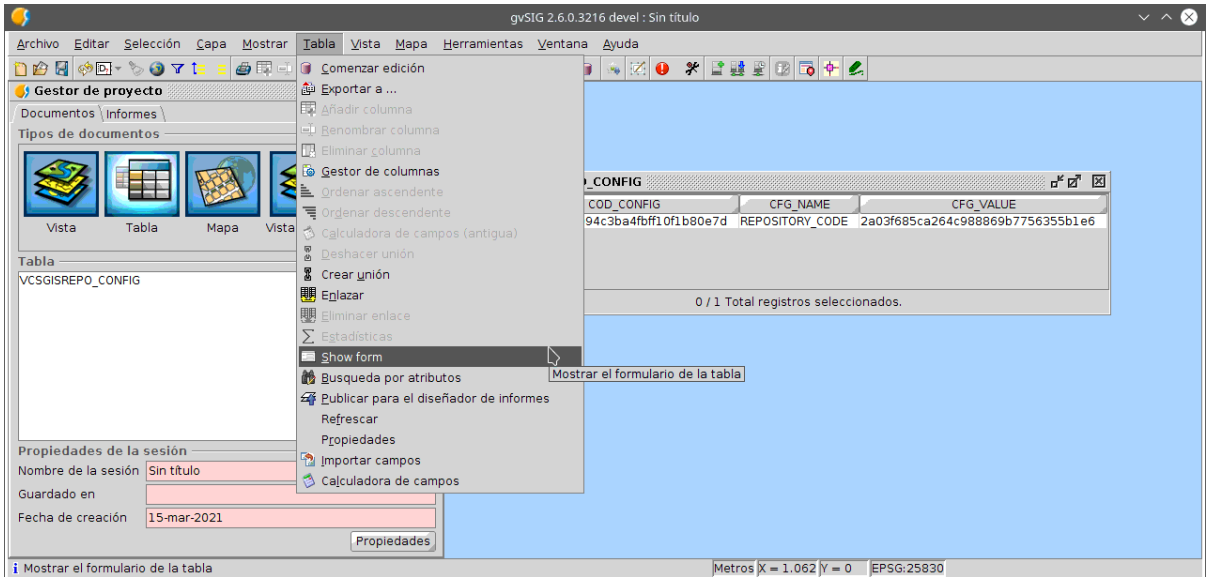




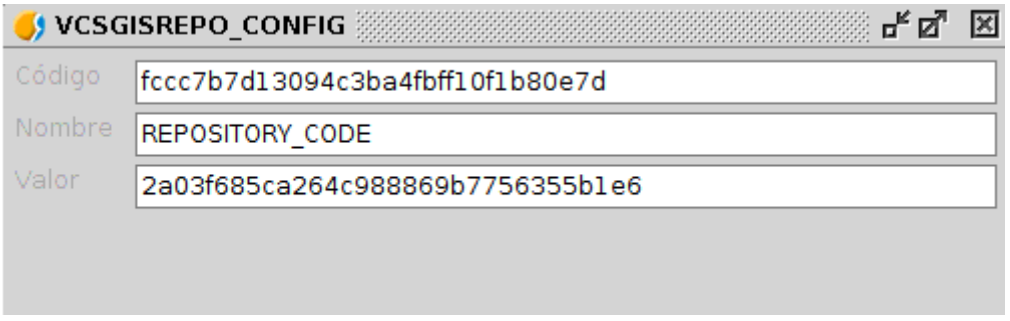
La tabla anterior se muestra en la siguiente imagen.



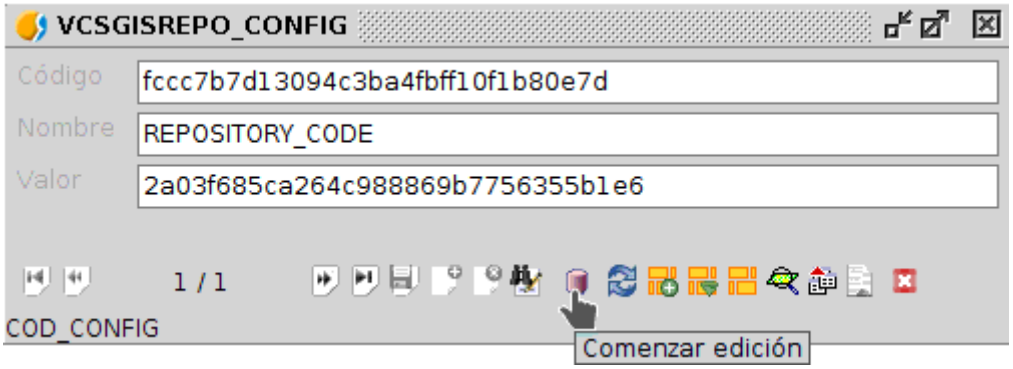
El proceso de asignación el nivel de autenticación se realiza añadiendo un nuevo elemento a esta tabla. Para ello es necesario obtener el formulario asociado la capa *PUBLIC.VCSGISREPO_CONFIG*. Para obtener el formulario de la tabla seleccionaremos la opción *Show form* situada en el menú *Tabla* de *gvSIG Desktop* siempre y cuando la tabla este abierta y seleccionada.



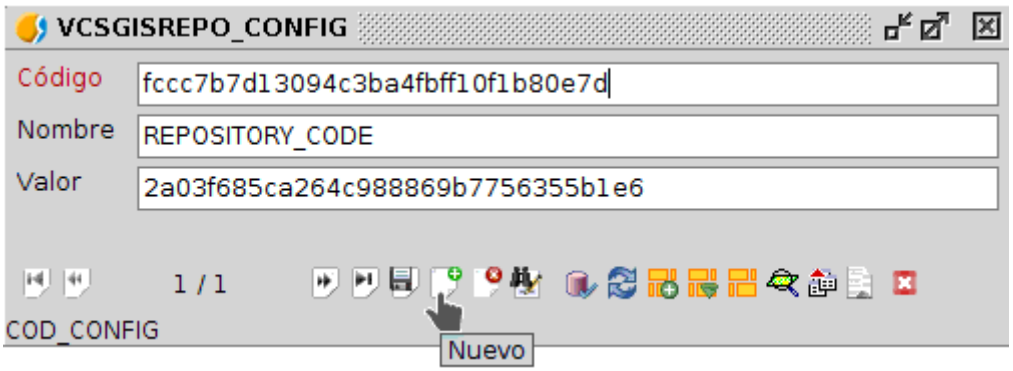
El formulario de la tabla es el siguiente.



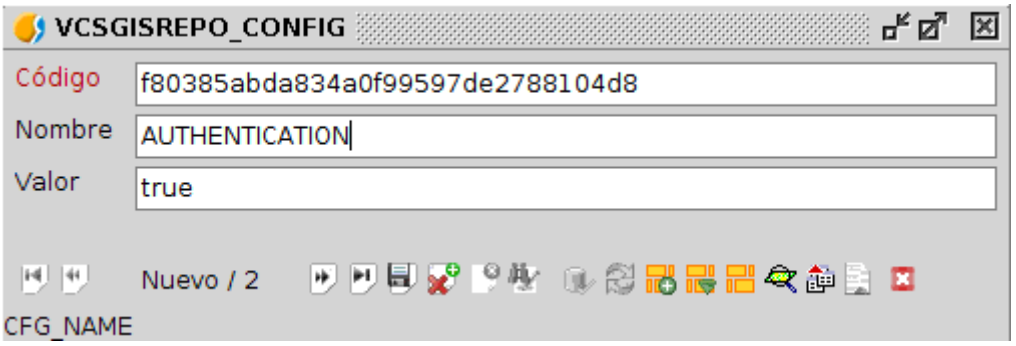
Una vez en el formulario se inicia la edición de la tabla para creación de un nuevo elemento. Este proceso se puede realizar desde el mismo desplegable que se mencionó anteriormente para obtener el formulario, o desde el mismo formulario utilizando el botón *Comenzar edición*.



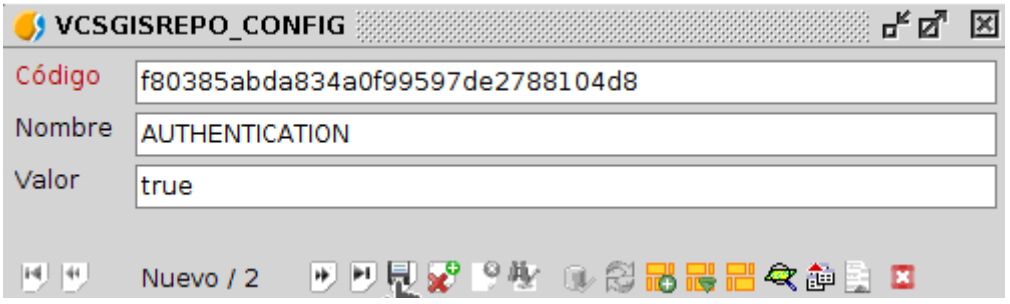
Una vez comenzada la edición se procede a crear un nuevo elemento en la tabla.

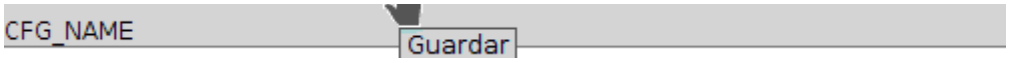


De los diferentes campos del formulario hay que identificar los campos *nombre* y *valor*. En el primero hay que especificar **AUTHENTICATION** y en el segundo **true**.

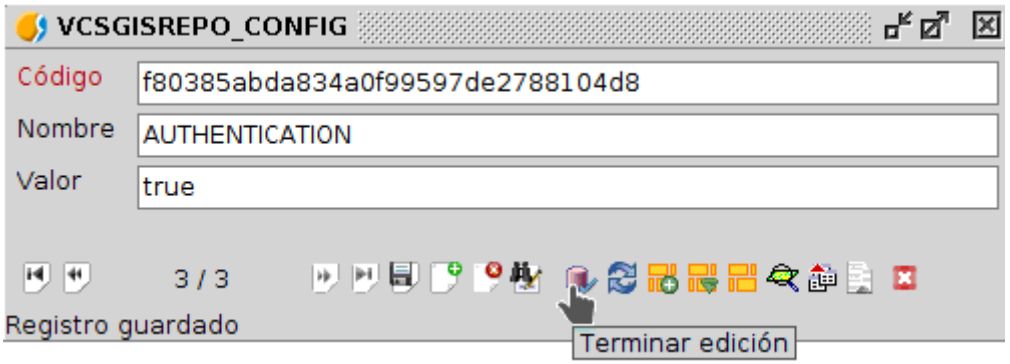


Solo queda guardar los cambios en la entidad.

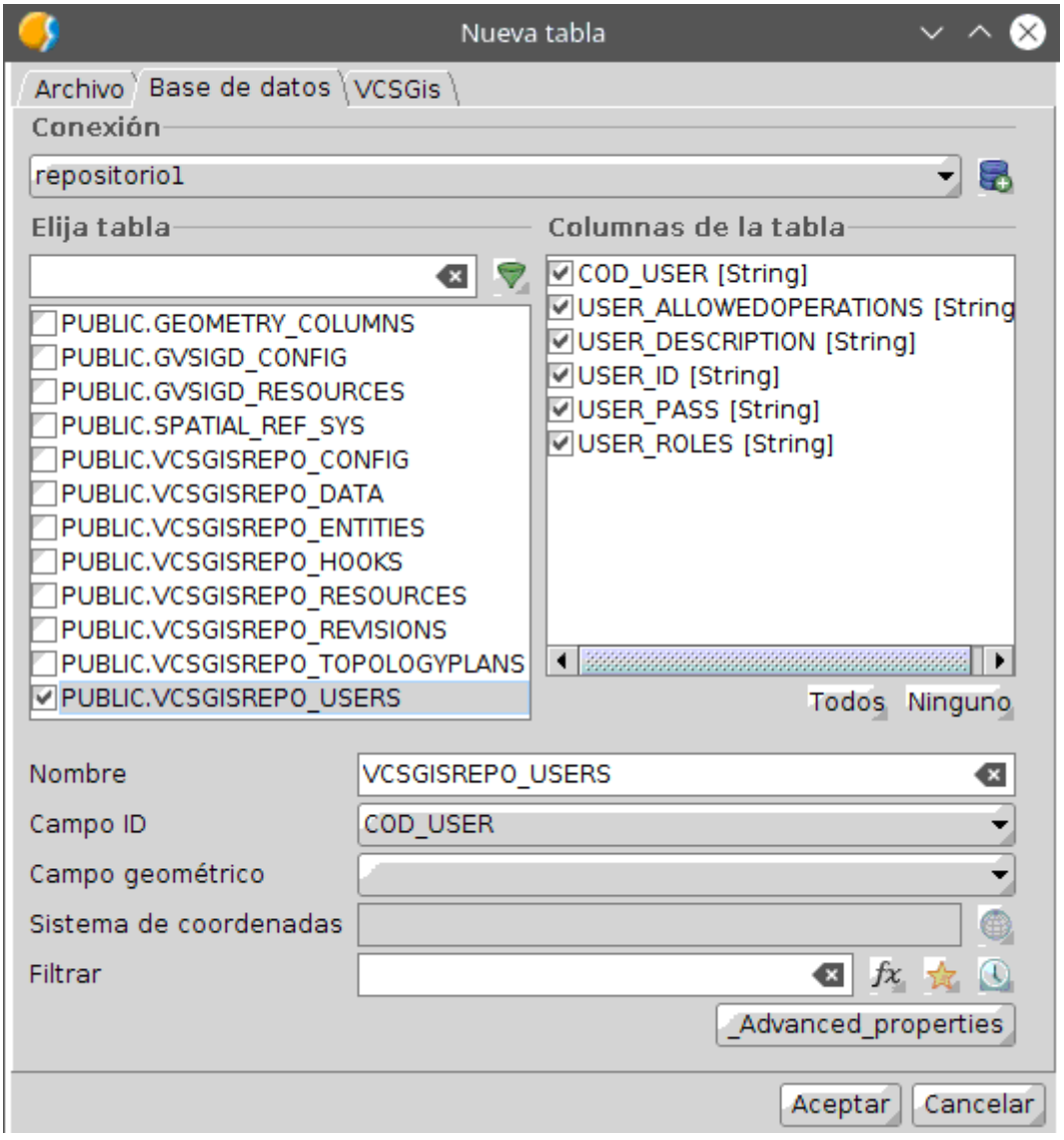




Y terminar la edición de la tabla.



Tras dotar al repositorio del nivel de seguridad basado en autenticación hay que crear los usuarios para poder acceder a este. Para ello se repite el proceso de abrir tabla y posteriormente formulario esta vez de la tabla *PUBLIC.VCSGISREPO_USERS*.

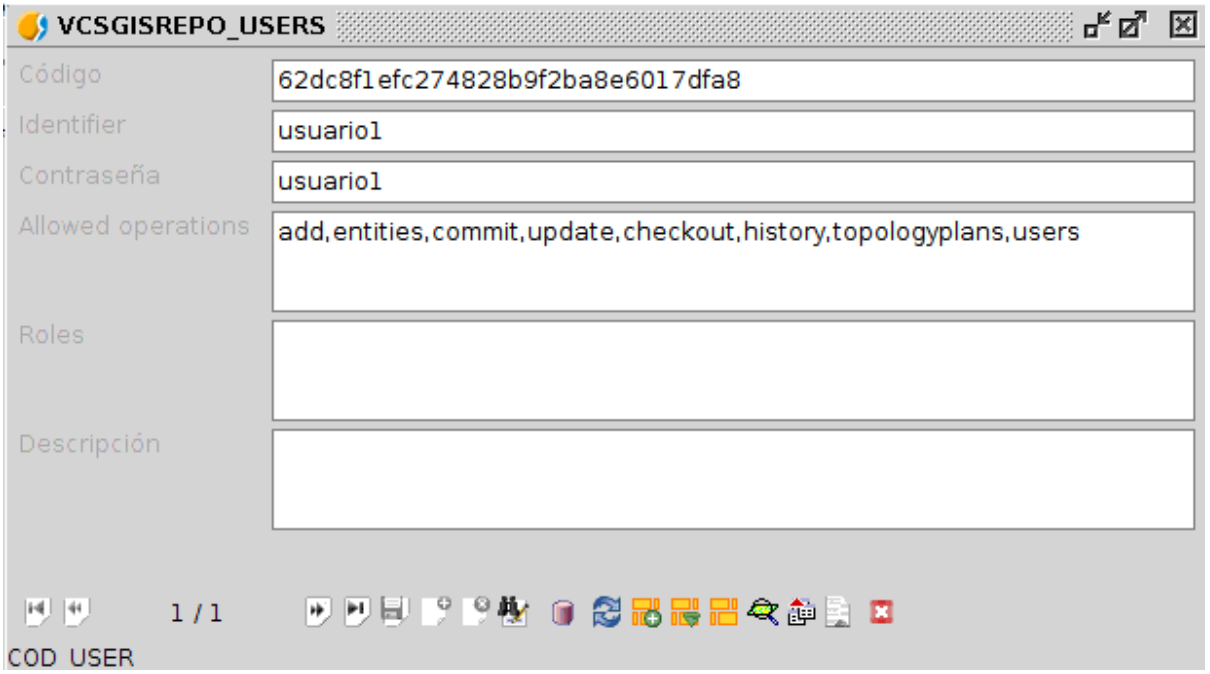


Una vez en el formulario se procede a crear un nuevo usuario siguiendo el flujo de trabajo para la creación de nuevos elementos explicado en la tabla anterior. Este flujo es el siguiente;

- Comenzar edición.
- Nuevo elemento.
- Rellenar campos del formulario.
- Guardar cambios.
- Terminar edición.

Los campos a rellenar para crear un nuevo usuario son el campo *Identificador* y el campo *Contraseña*, dependiendo el nombre de usuario y contraseña del usuario en cuestión.

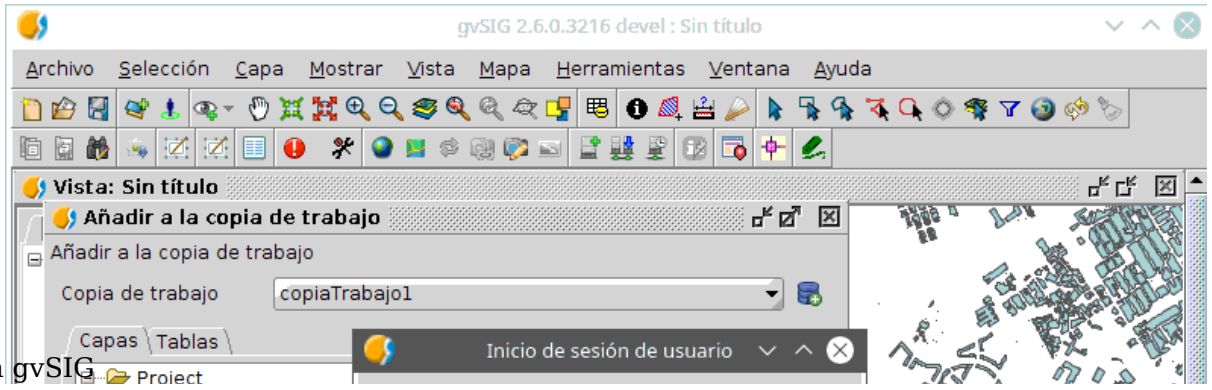
La siguiente ilustración muestra el ejemplo de un usuario creado.

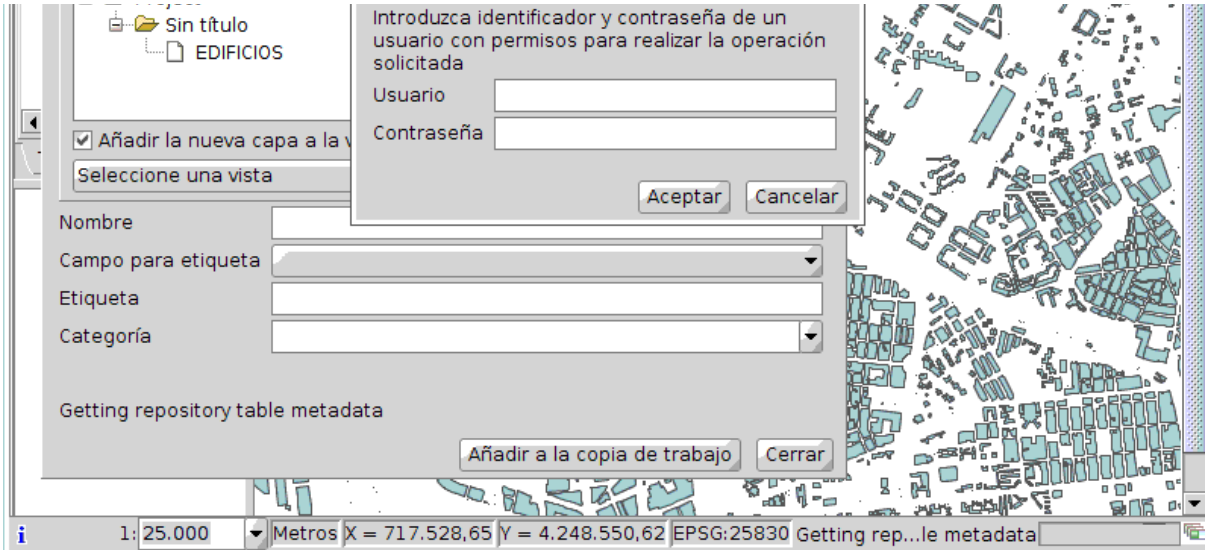


El usuario se llama *usuario1*, su contraseña es *usuario1* y presenta permisos para realizar todas las acciones.

Con lo anterior un repositorio ya puede presentar seguridad a nivel de autenticación y además en el caso del ejemplo ya dispone de un usuario. El siguiente paso lógico sería la creación de una copia de trabajo por parte del usuario y comenzar el trabajo de manera normal. Cuando el usuario se disponga a realizar una de las acciones referidas a la información del repositorio se habilitará una ventana para su identificación mediante nombre de usuario y contraseña.

Un ejemplo de lo anterior se ve en la siguinte ilustración. En la imagen se muestra la ventana que pide los credenciales tras intentar subir una nueva capa a la copa de trabajo.





Autorización básica

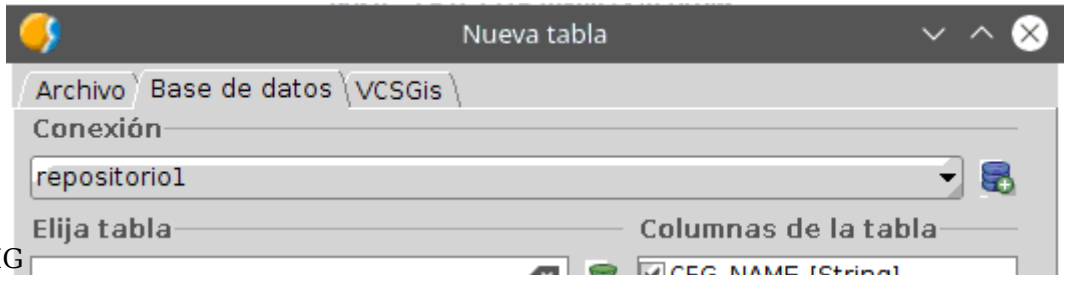
Antes de empezar con la autorización, hay que decir que que no existe un nivel de autorización básica o avanzada como tal sino que solo existe el nivel de autorización. Esta forma de dividir en dos apartados la autorización se debe a que se puede aplicar la autorización de dos maneras diferentes siendo una más restrictiva que la otra. La autorización básica controla las acciones que pueden realizar los diferentes usuarios, mientras que la avanzada define en los propios datos quien y como puede trabajarlos. Por lo tanto la autorización avanzada es más restrictiva que la básica.

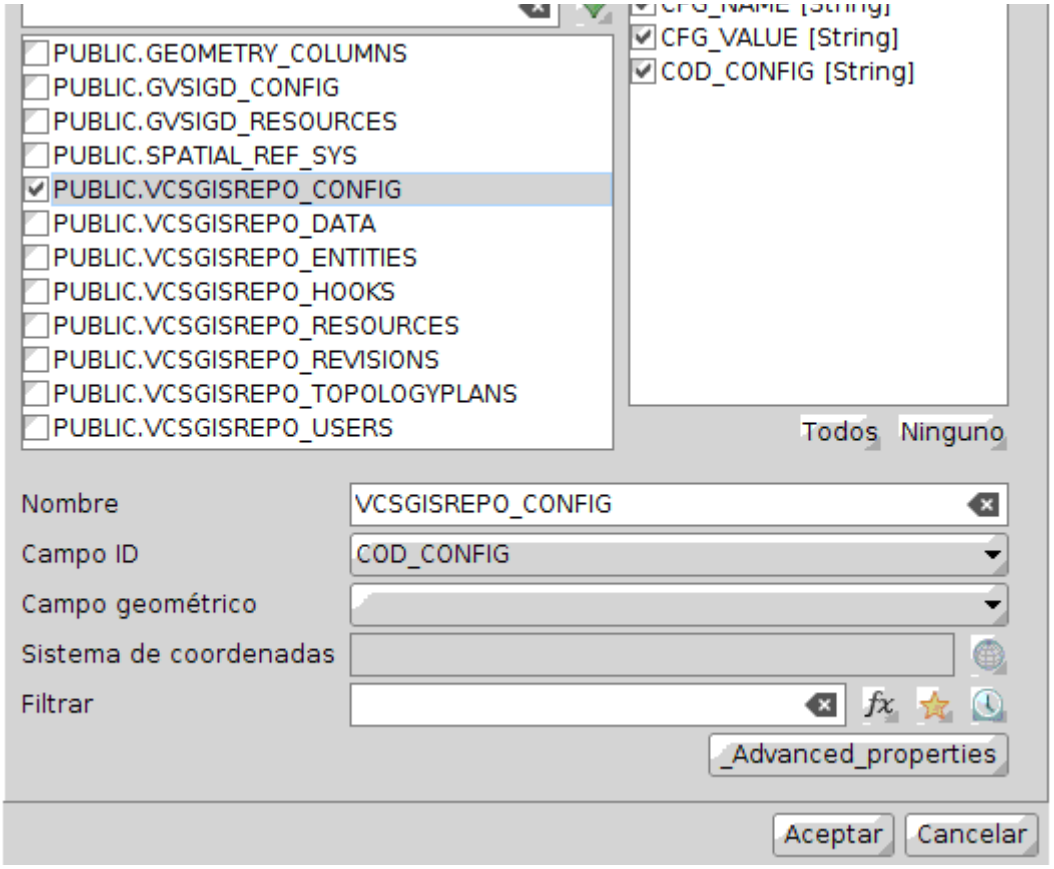
En este apartado se detalla el proceso de configuración para dotar de nivel de seguridad de autorización básica a un repositorio. Para ello y siguiendo el hilo de la documentación de *VCSGis* se utilizan un ejemplo práctico.

Activar el proceso de autorización se realiza de igual manera que al activar el proceso de autenticación, es decir, realizando modificaciones sobre la tabla *PUBLIC.VCSGISREPO_CONFIG*.

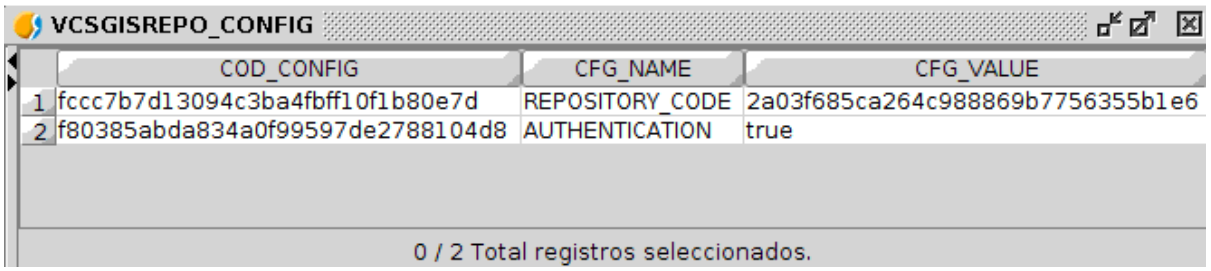
Para abrir dicha tabla hay que realizarlo desde el *Gestor de proyectos* situado en el menú *Mostrar* de *gvSIG Desktop*. El proceso de abrir una tabla es el genérico a abrir cualquier archivo, primero se selecciona *Tabla* como tipo de datos a abrir, se selecciona la opción de *Nuevo*, lo que habilita una ventana donde se tiene que seleccionar la pestaña *Base de datos*. Esa pestaña muestra en su zona superior un desplegable donde hay que especificar la base de datos donde se encuentra la tabla. Una vez seleccionada la base de datos, en la lista de tablas de esta hay que marcar la tabla en cuestión y pulsa el botón *Aceptar*.

A continuación se muestra la ventana con la que se abre la tabla *PUBLIC.VCSGISREPO_CONFIG*.

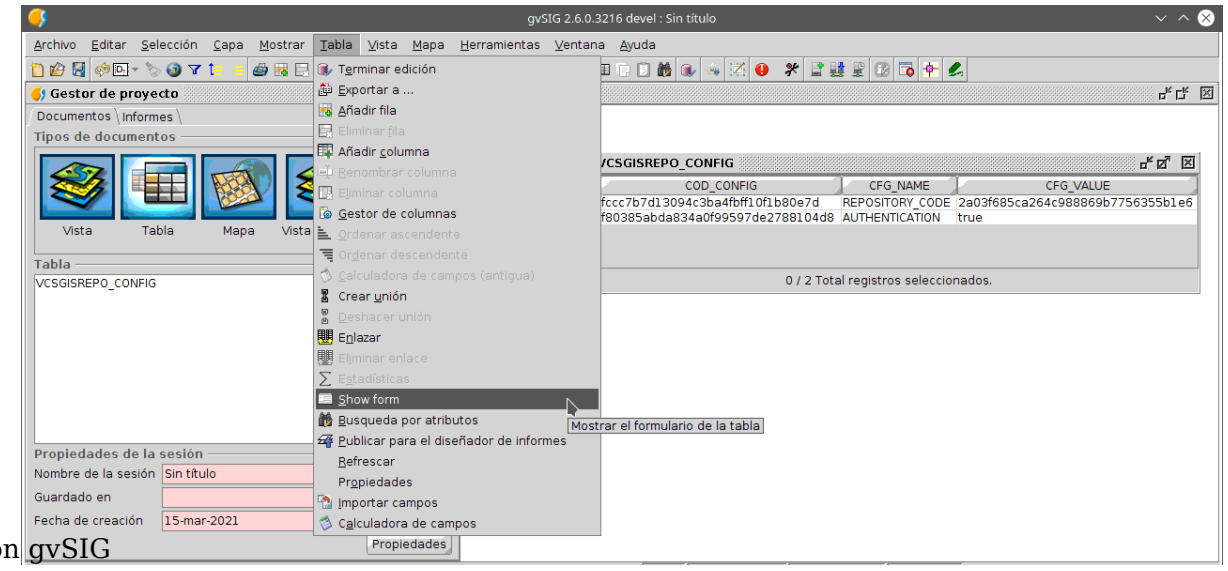




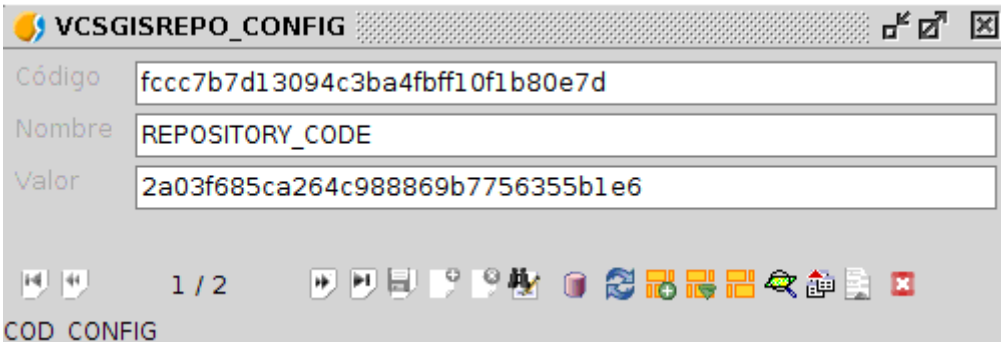
La tabla anterior se muestra en la siguiente imagen.



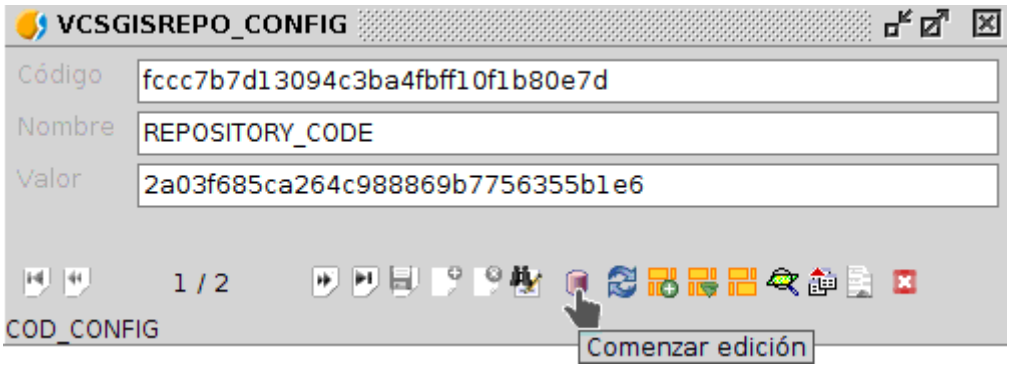
Para dotar a este repositorio ejemplo de este nivel de seguridad solo hay que registrar un nuevo elemento en dicha tabla. Lo anterior puede realizarse mediante el formulario asociado la tabla. Para obtener el formulario de la tabla seleccionaremos la opción *Show form* situada en el menú *Tabla* de *gvSIG Desktop* siempre y cuando la tabla este abierta y seleccionada.



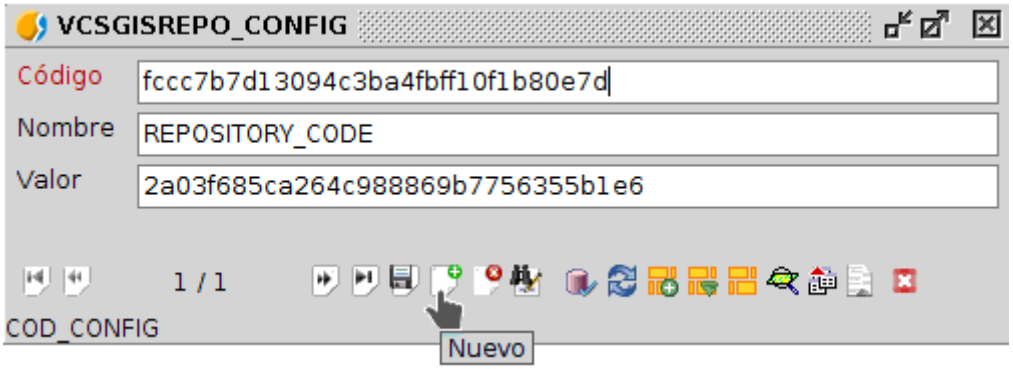
El formulario de la tabla es el siguiente.



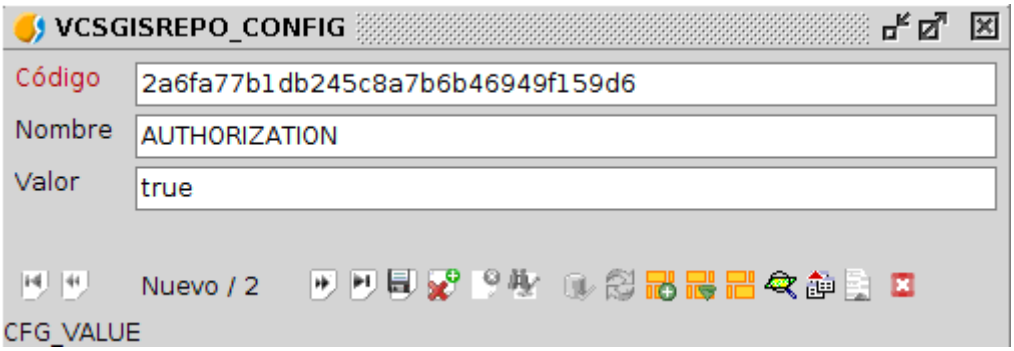
Una vez en el formulario se inicia la edición de la tabla para creación de un nuevo elemento. Este proceso se puede realizar desde el mismo desplegable que se mencionó anteriormente para obtener el formulario, o desde el mismo formulario utilizando el botón *Comenzar edición*.



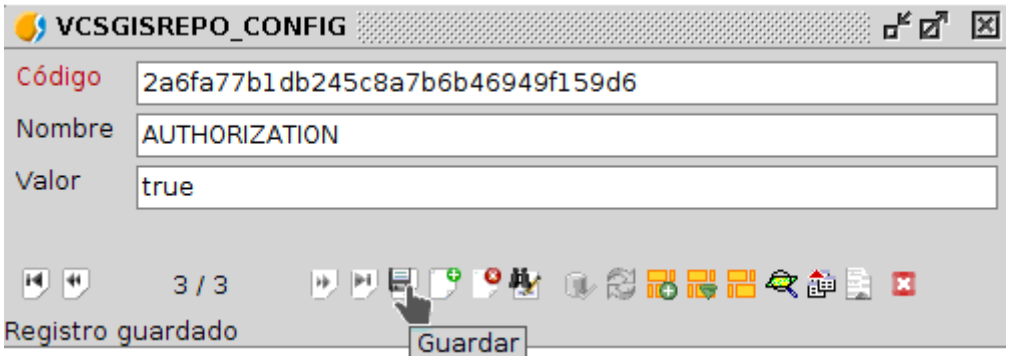
Una vez comenzada la edición se procede a crear un nuevo elemento en la tabla.



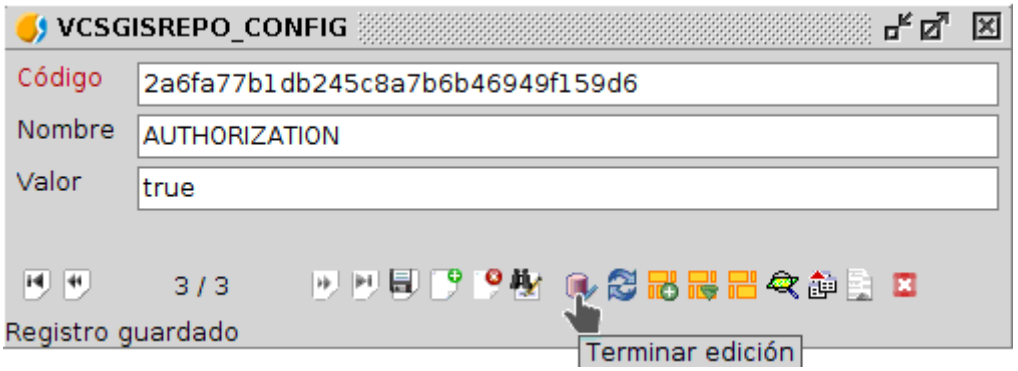
De los diferentes campos del formulario hay que identificar los campos *nombre* y *valor*. En el primero hay que especificar **AUTHORIZATION** y en el segundo **true**.



Solo queda guardar los cambios en la entidad.



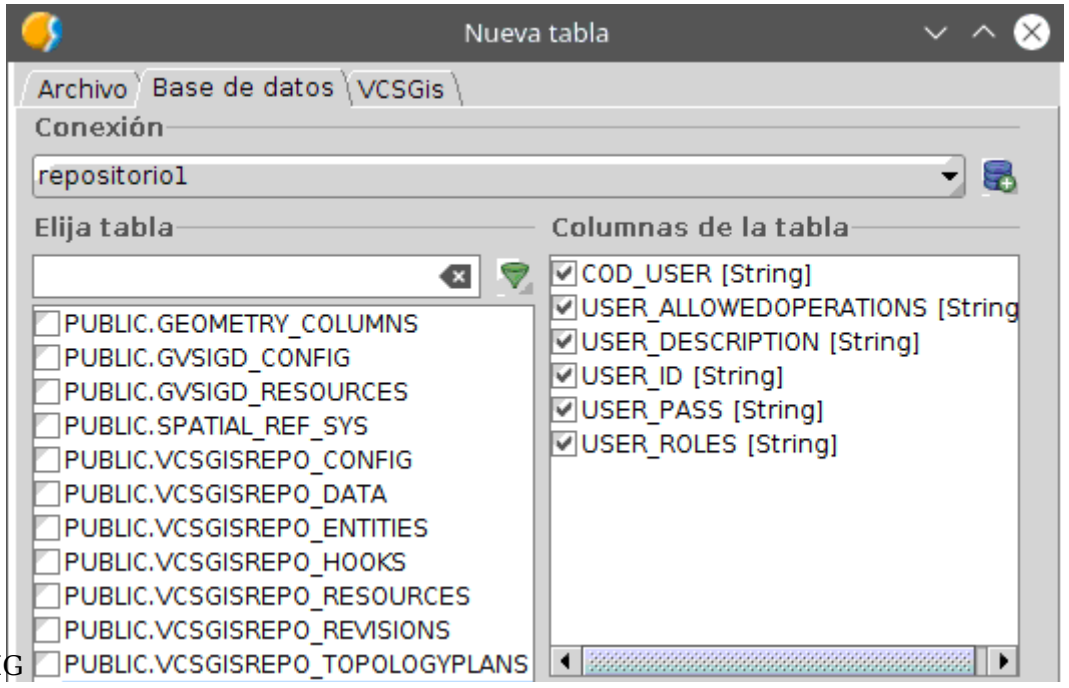
Y terminar la edición de la tabla.

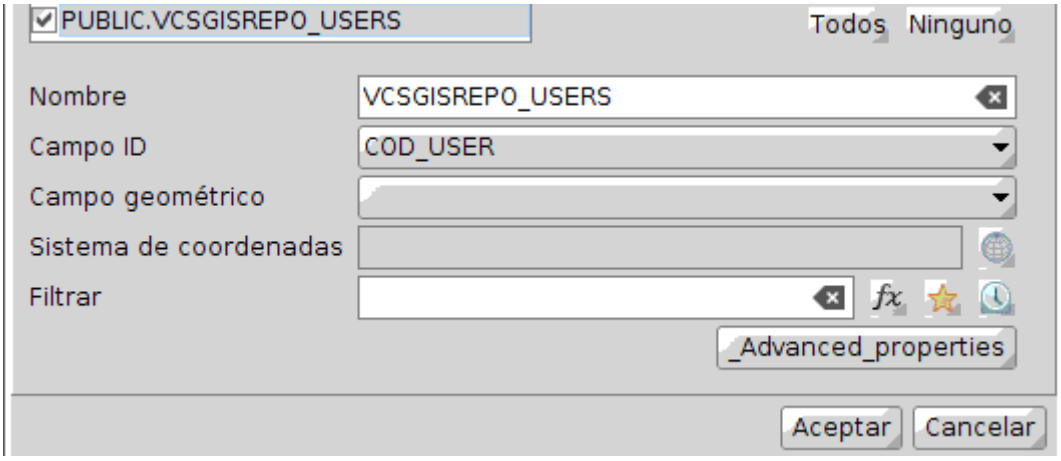


Con lo anterior ya se dispone del nivel de seguridad autorización aplicado en el repositorio.

Solo queda ahora configurar las acciones que pueden realizar los diferentes usuarios autenticados previamente sobre el repositorio ejemplo. Para hacer lo anterior hay que editar dichos usuarios en la tabla *PUBLIC.VCSGISREPO_USERS*.

Para realizar dicha edición de tabla hay que abrir la tabla y sacar el formulario asociado. La tabla se obtiene de igual manera que la anterior al igual que su formulario. El formulario de la tabla *PUBLIC.VCSGISREPO_USERS* se muestra acontinuación.





El flujo de trabajo para la edición de usuarios en el formulario anterior es el siguiente;

- Comenzar edición.
- Editar elemento.
- Rellenar/cambiar campos del formulario.
- Guardar cambios.
- Terminar edición.

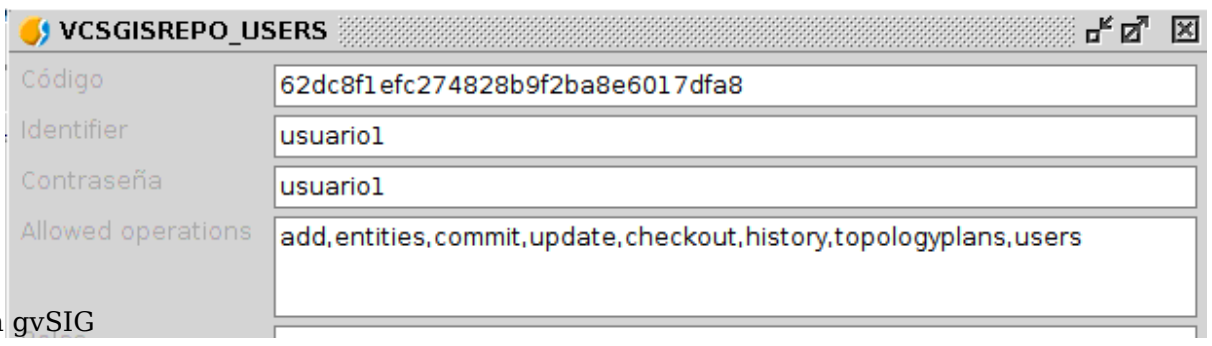
El campo en cuestión sobre el cual hay que realizar las modificaciones es el campo *acciones*.

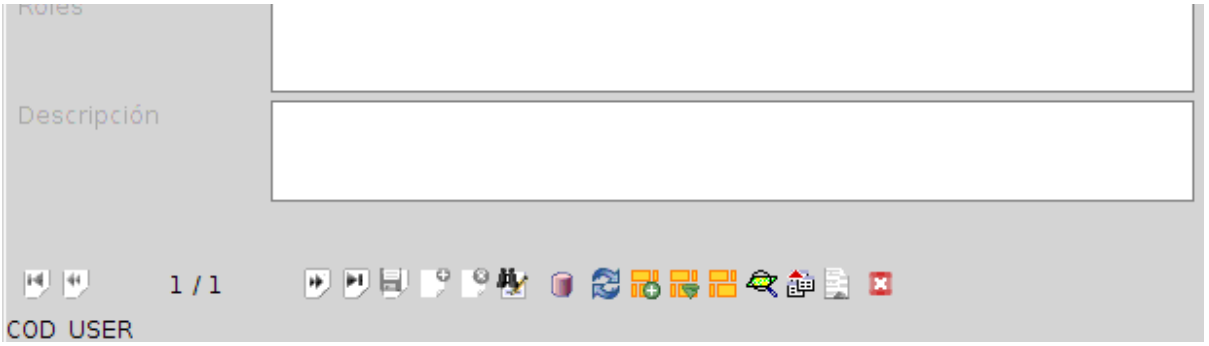
Como se detalla anteriormente hay campos de estos elementos que son propios para cada usuario, pero las acciones son un parámetro generales y como su propio nombre indica son las acciones que ese usuario puede realizar en el repositorio. Las acciones disponibles sobre el repositorio se listan a continuación:

- **add**. Añadir una nueva tabla al repositorio.
- **entities**. Listar las entidades existentes en el repositorio.
- **commit**. Subir cambios a las capas del repositorio.
- **update**. Descargar los cambios de las capas del repositorio.
- **checkout**. Descargar una capa del repositorio.
- **history**. Consulta el historial de revisiones.
- **topologyplan**. Descargar la tabla con los planes de topología.
- **user**. Descargar la tabla de usuarios.

  Un usuario sin acceso a las acciones *entities*, *topologyplan* y *user* no solo no podrá descargar dicha información sino que tampoco tendrá autorización para cualquier otra acción relacionada con los elementos que habilitan las acciones anteriores.

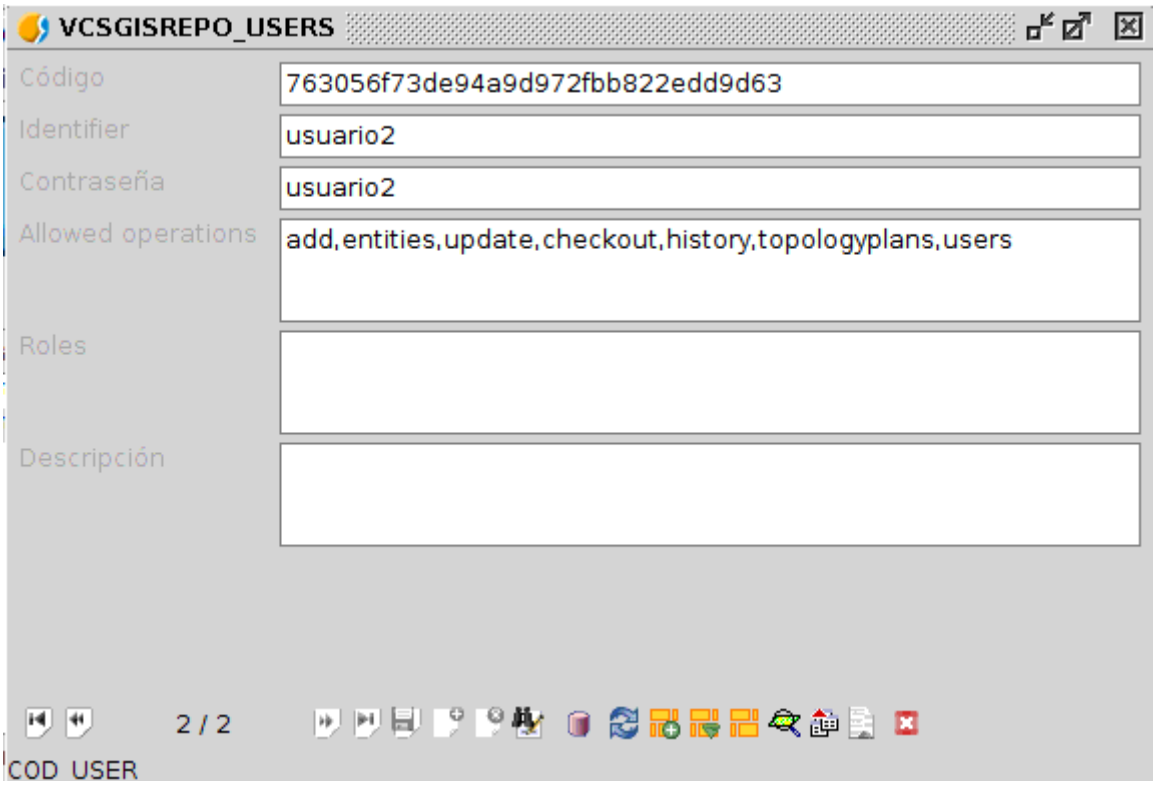
La siguiente ilustración muestra el ejemplo de un usuario editado con acciones.



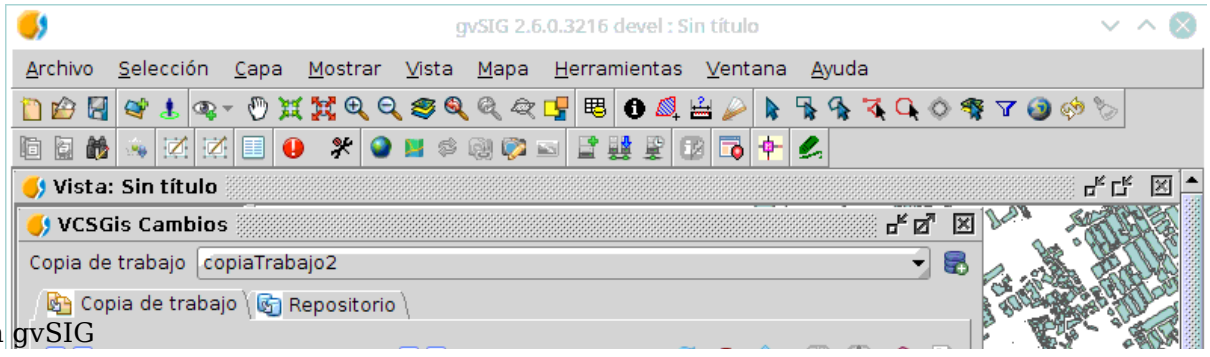


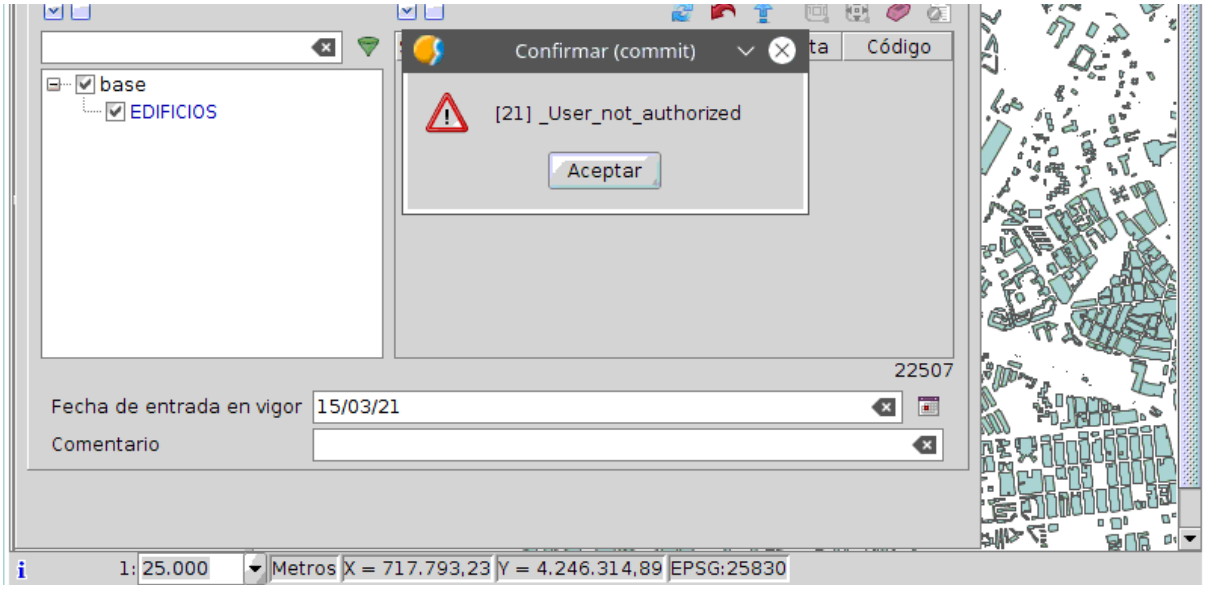
El usuario se llama *usuario1*, su contraseña es *usuario1* y presenta permisos para realizar todas las acciones posibles sobre el repositorio ejemplo.

Existe como es lógico, la posibilidad de que un usuario no presente autorización e algunas de las acciones. El siguiente usuario ejemplo, *usuario2*, presenta todas las acciones menos la acción *commit*.



Si tras realizar cambios sobre una capa/tabla del repositorio sujeta al control de versiones este usuario intenta subir dichos cambios locales al repositorio, el sistema le mostrará la ventana para introducir credenciales. Al identificarse como *usuario2* no lo permitirá mostrando la ventana del login hasta que se introduzca un usuario y contraseña válido o se pulse el botón *Cancelar*, el cual muestra el siguiente cuadro de diálogo.





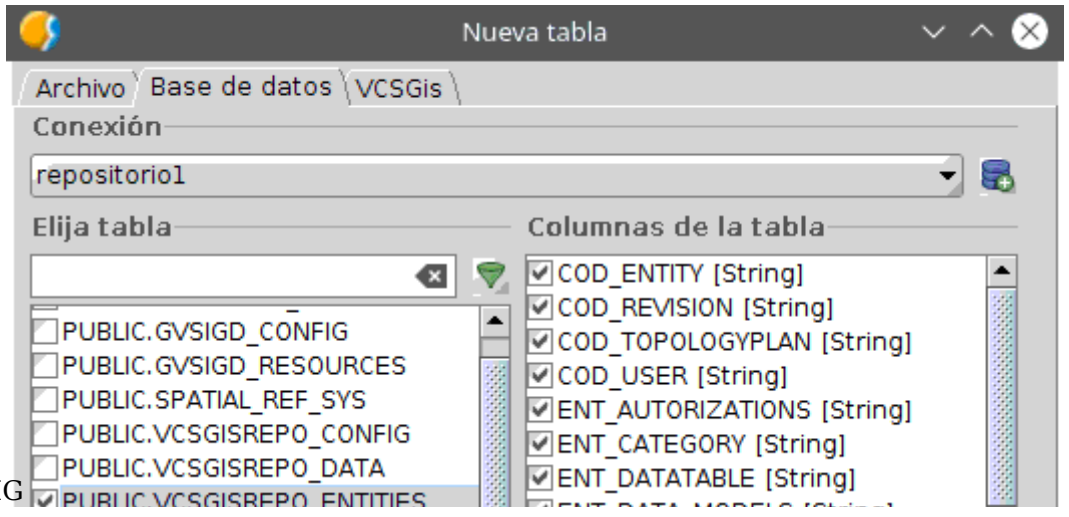
Autorización avanzada

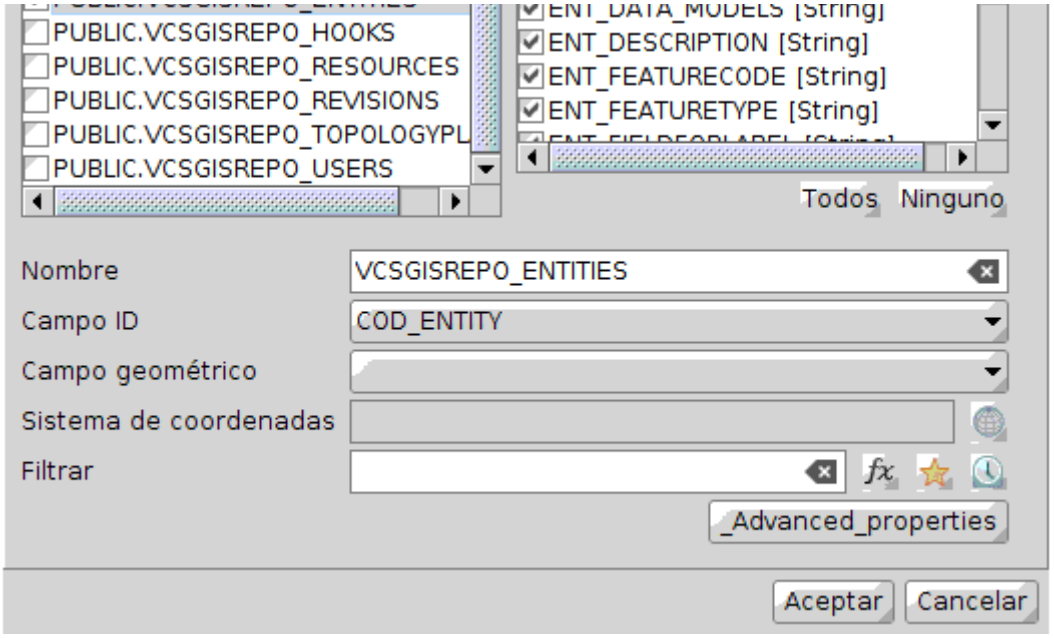
La autorización avanzada representa el mayor grado de seguridad ofrecido por la herramienta *VCSGis* de *gvSIG Desktop*. Como se detalla anteriormente se basa en definir en el propio dato los usuarios y acciones autorizados a trabajar con este.

Para realizar dicha definición hay que realizar modificaciones sobre la tabla *PUBLIC.VCSGISREPO_ENTITIES*. Dicha tabla lista los diferentes elementos registrados en el repositorio, es decir las capa/tablas. De modo que, sobre los elementos anteriores hay que definir que acciones pueden realizarse sobre la capa y quien puede llevarla a cabo.

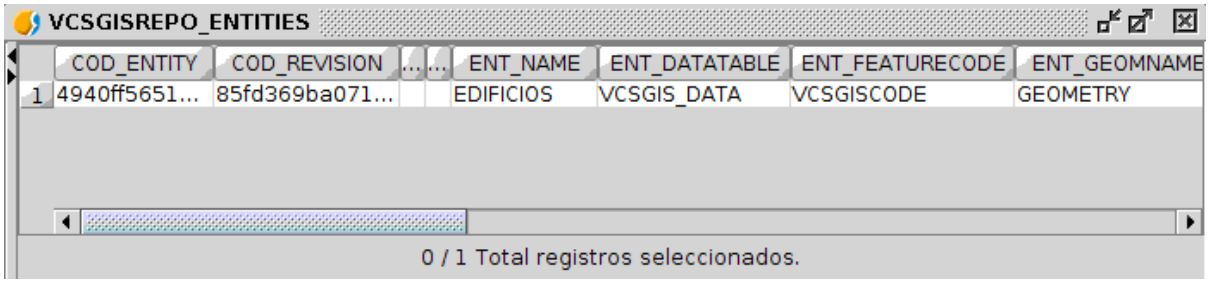
Para abrir las tablas hay que realizarlo desde el *Gestor de proyectos* situado en el menú *Mostrar* de *gvSIG Desktop*. El proceso de abrir una tabla es el genérico a abrir cualquier archivo, primero se selecciona *Tabla* como tipo de datos a abrir, se selecciona la opción de *Nuevo*, lo que habilita una ventana donde se tiene que seleccionar la pestaña *Base de datos*. Esa pestaña muestra en su zona superior un desplegable donde hay que especificar la base de datos donde se encuentra la tabla. Una vez seleccionada la base de datos, en la lista de tablas de esta hay que marcar la tabla en cuestión y pulsa el botón *Aceptar*.

A continuación se muestra la ventana con la que se abre la tabla *PUBLIC.VCSGISREPO_ENTITIES*.

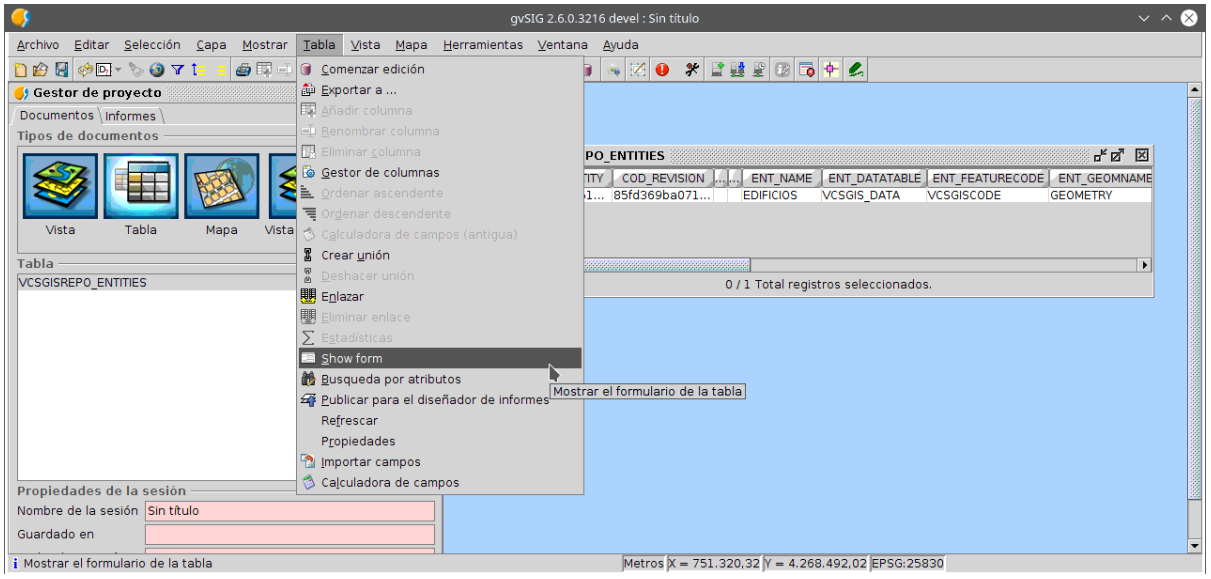




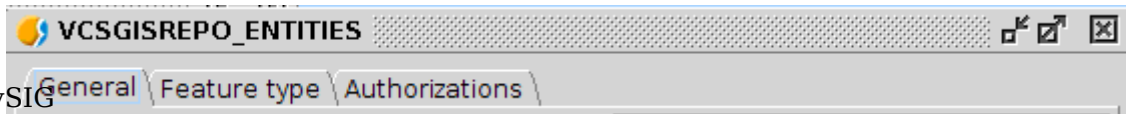
La tabla anterior se muestra en la siguiente imagen.

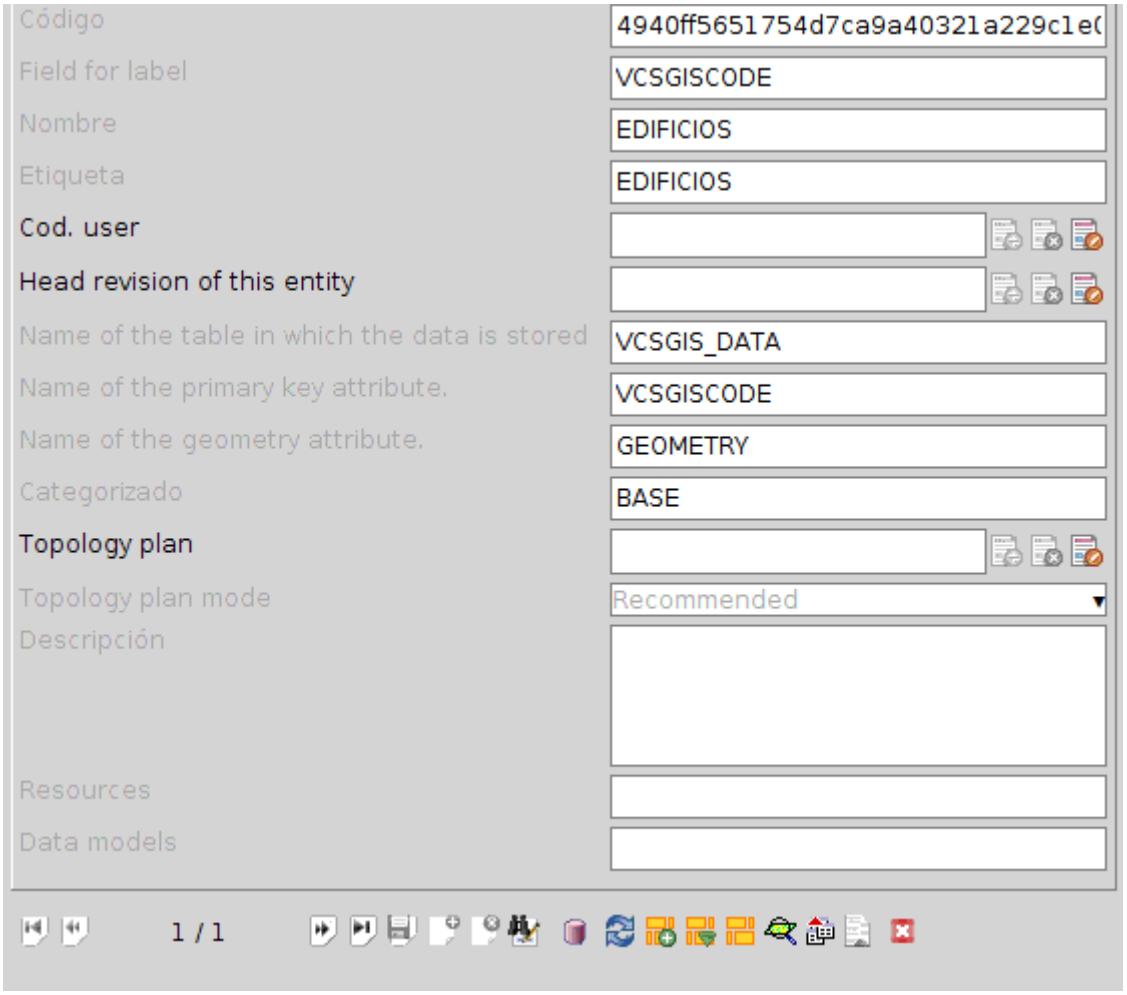


El proceso de definición de acciones y usuarios por dato se realiza modificando elementos a esta tabla. Para ello es necesario obtener el formulario asociado la capa *PUBLIC.VCSGISREPO_ENTITIES*. Para obtener el formulario de la tabla seleccionaremos la opción *Show form* situada en el menú *Tabla* de *gvSIG Desktop* siempre y cuando la tabla este abierta y seleccionada.

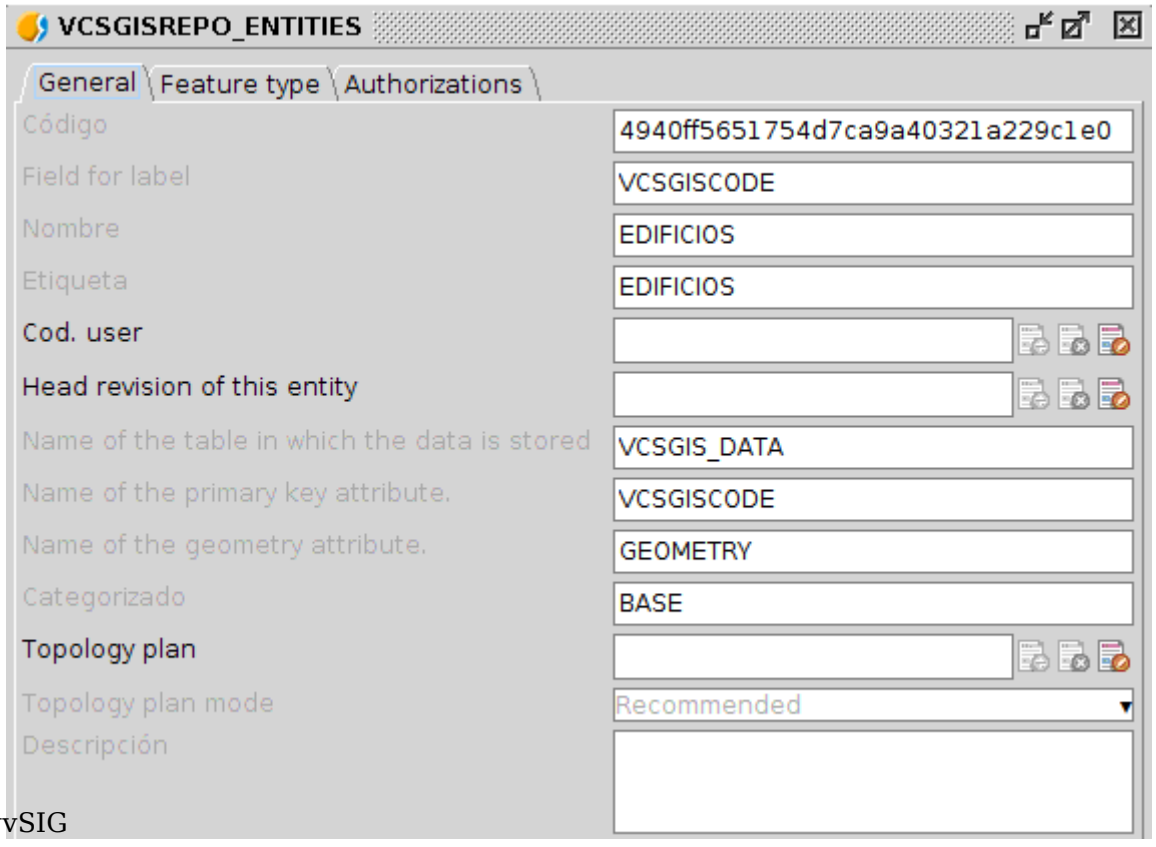


El formulario de la tabla es el siguiente.

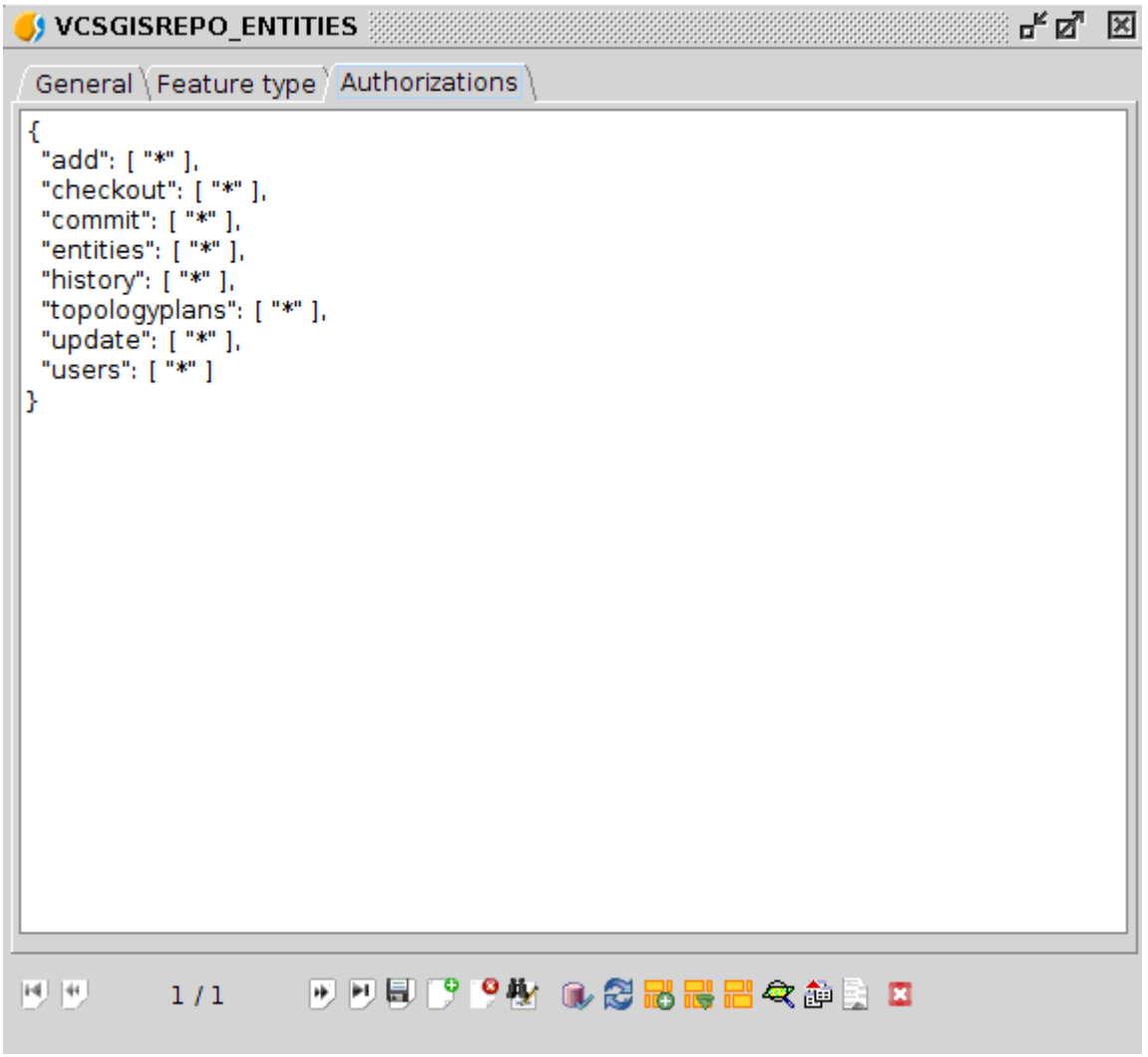




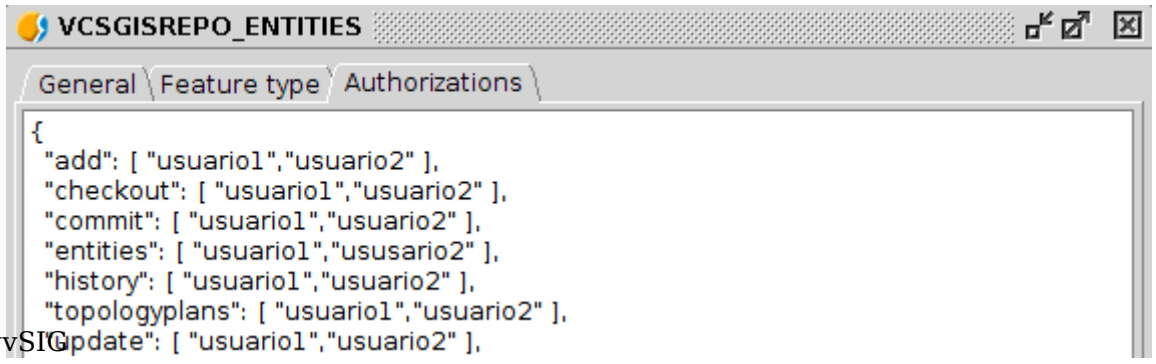
Una vez en el formulario se inicia la edición de la tabla para la modificación del elemento. Este proceso se puede realizar desde el mismo desplegable que se mencionó anteriormente para obtener el formulario, o desde el mismo formulario utilizando el botón *Comenzar edición*.

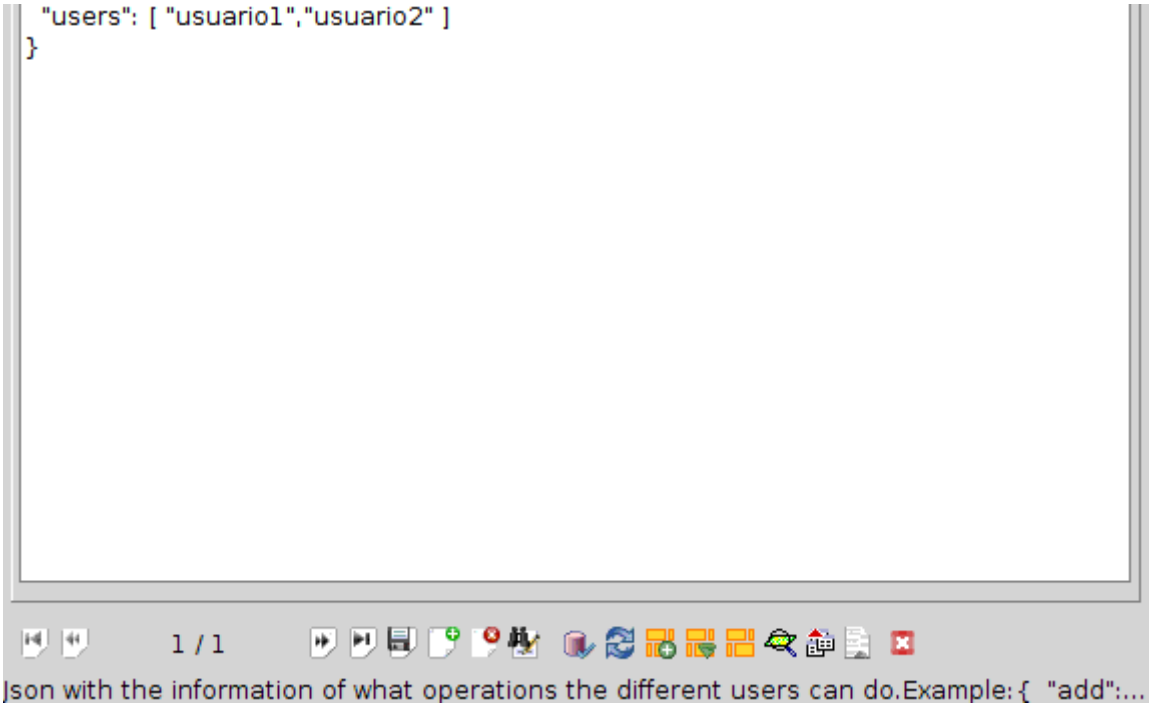


De los diferentes elementos del formulario hay que seleccionar la pestaña *Authorization* la cual despliega un panel sobre el cual hay que modificar un texto en formato `Json` el cual asigna a las diferentes acciones, los usuarios que pueden realizarlas sobre dicha entidad o capa.

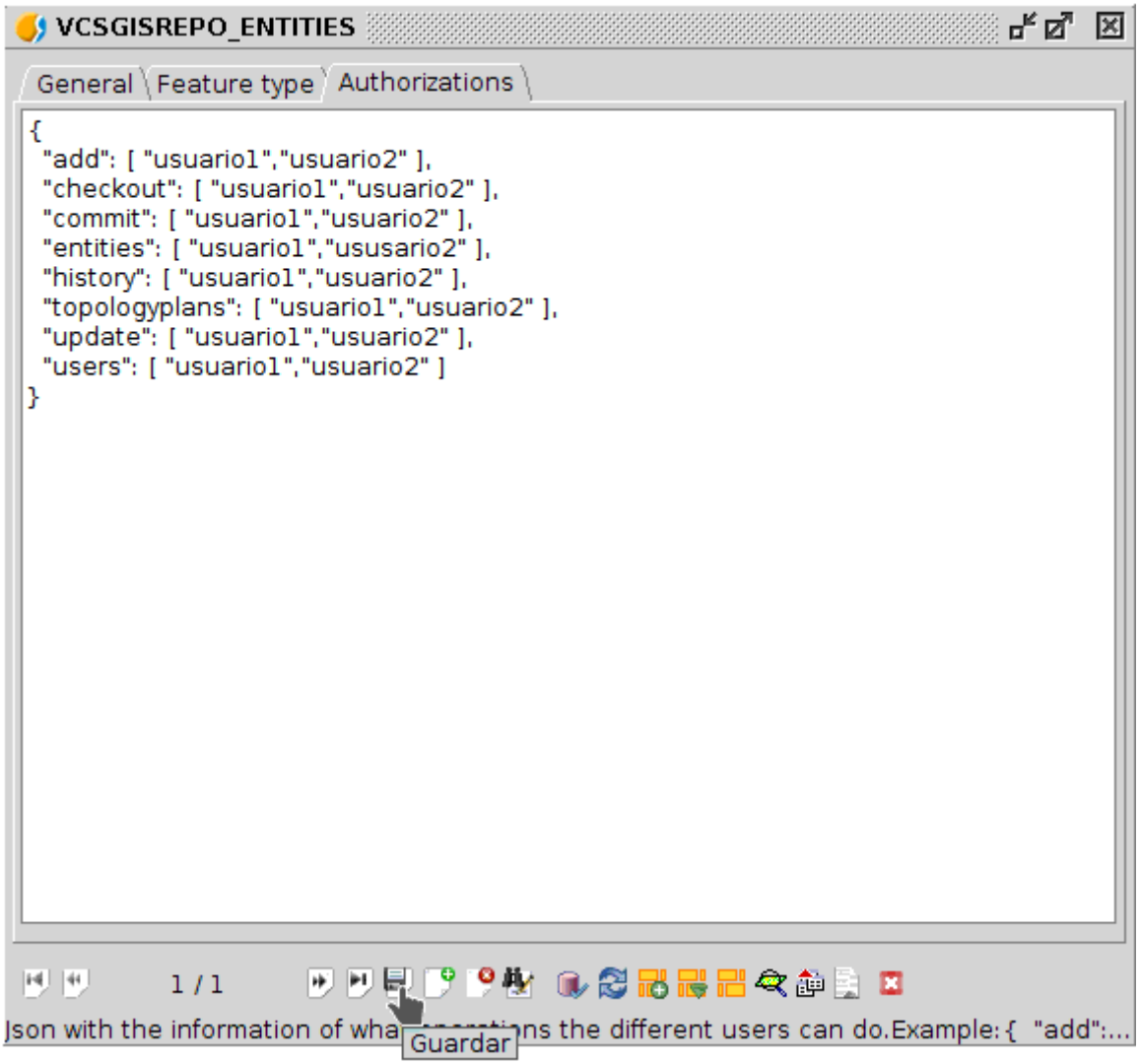


En el caso del ejemplo que se muestra en la siguiente imagen los usuarios *usuario1* y *usuario2*, creado previamente en los apartados anteriores, van a disponer de las posibilidad de realizar todas las acciones sobre la capa seleccionada.





Solo queda guardar los cambios en la entidad.

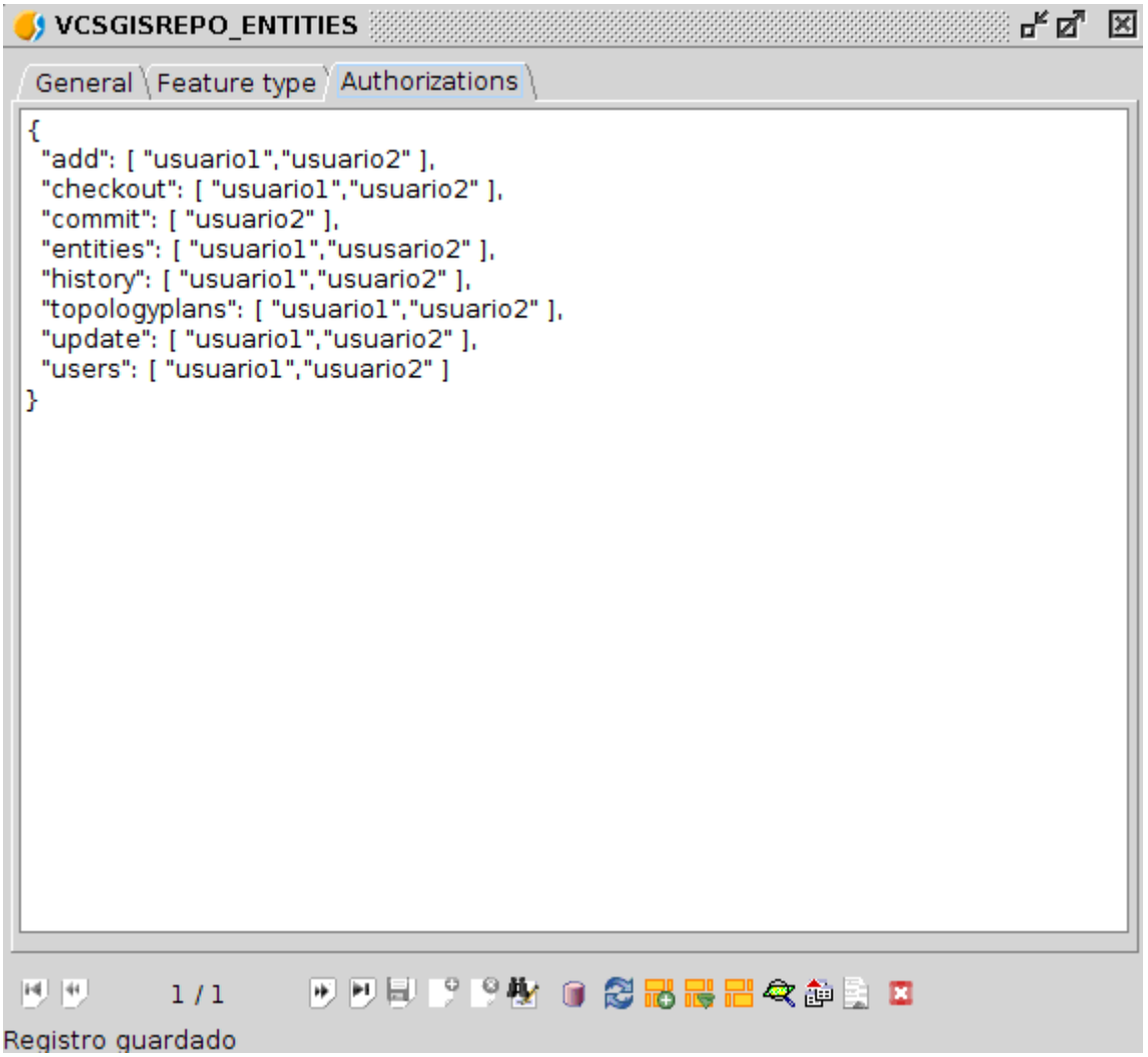


Y terminar la edición de la tabla.

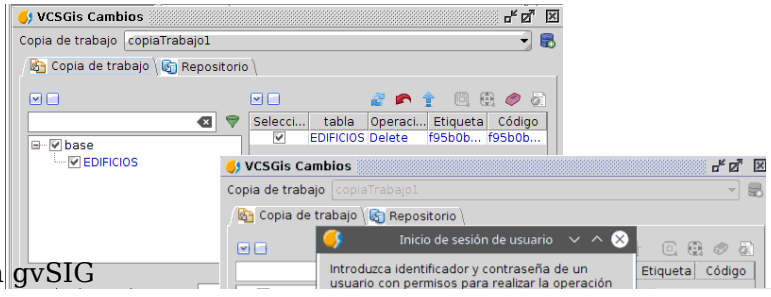
9_tEditarFormPUBLICVCSGISREPO_CONFIG_128

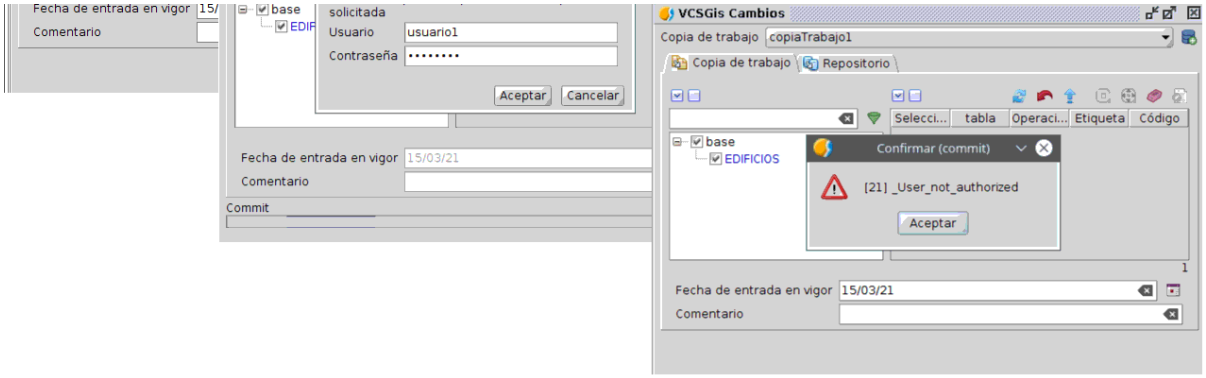
Tras lo anterior la entidad ha quedado configurada con la definición de acciones permitidas por usuario, pero es importante destacar que la definición propia de cada usuario se mantiene activa. De la frase anterior se deduce, que el *usuario2* aún teniendo permitido el *commit* en la entidad configurada, no podrá realizarlo pues de base este no tiene permitida esa acción.

De igual manera pero en sentido contrario, si tras realizar todo lo anterior se configura que el *usuario1* no pueda realizar *commit* sobre la entidad, por mucho que este presente la acción habilitada en su configuración de usuario no podrá hacer *commit* sobre la entidad.



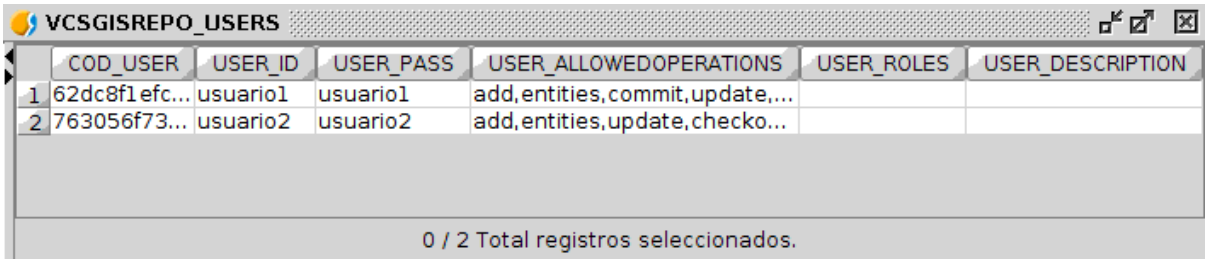
Si tras realizar cambios sobre una capa/tabla del repositorio sujeta al control de versiones este usuario intenta subir dichos cambios locales al repositorio el sistema le mostrará la ventana para introducir credenciales. Al identificarse como *usuario1* no lo permitirá mostrando la ventana del login hasta que se introduzca un usuario y contraseña válido o se pulse el botón *Cancelar*, el cual muestra una ventana que indica que hay un error en la autorización.





Además de lo anteriormente aplicado a la configuración de la autorización avanzada en *VCSGis*, la herramienta dispone de la posibilidad de aplicar un sistema de roles en la gestión de usuarios. Un usuario puede presentar además de su identificador propio tantos roles como se deseen. Los roles se definen en la tabla *PUBLIC.VCSGISREPO_USERS* en cada usuario en su creación o modificándolos posteriormente.

Para añadir los roles hay que abrir la tabla y posteriormente el formulario asociado.

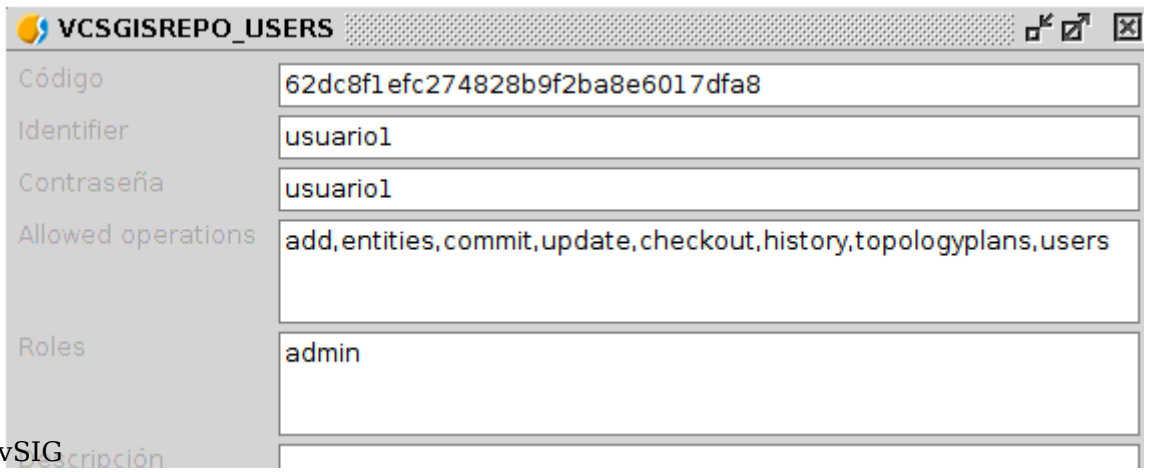


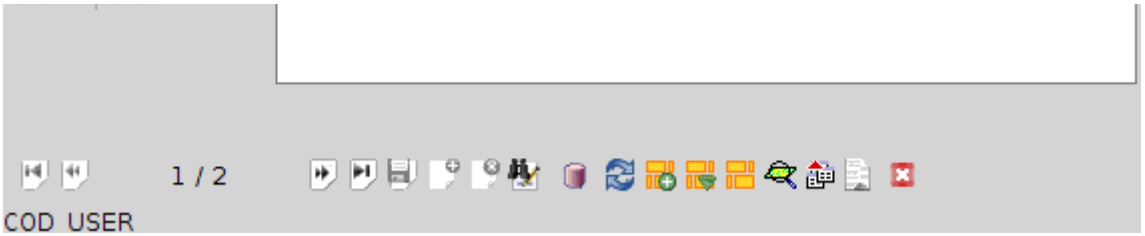
Una vez en el formulario se procede a crear un nuevo usuario o modificarlo siguiendo el siguiente flujo de trabajo;

- Comenzar edición.
- Nuevo elemento más rellenar campos del formulario o modificar campos del formulario.
- Guardar cambios.
- Terminar edición.

El campo en cuestión sobre el cual hay que asignar el rol o roles del usuario es el campo *roles*.

La siguiente ilustración muestra el ejemplo de un usuario editado con roles.

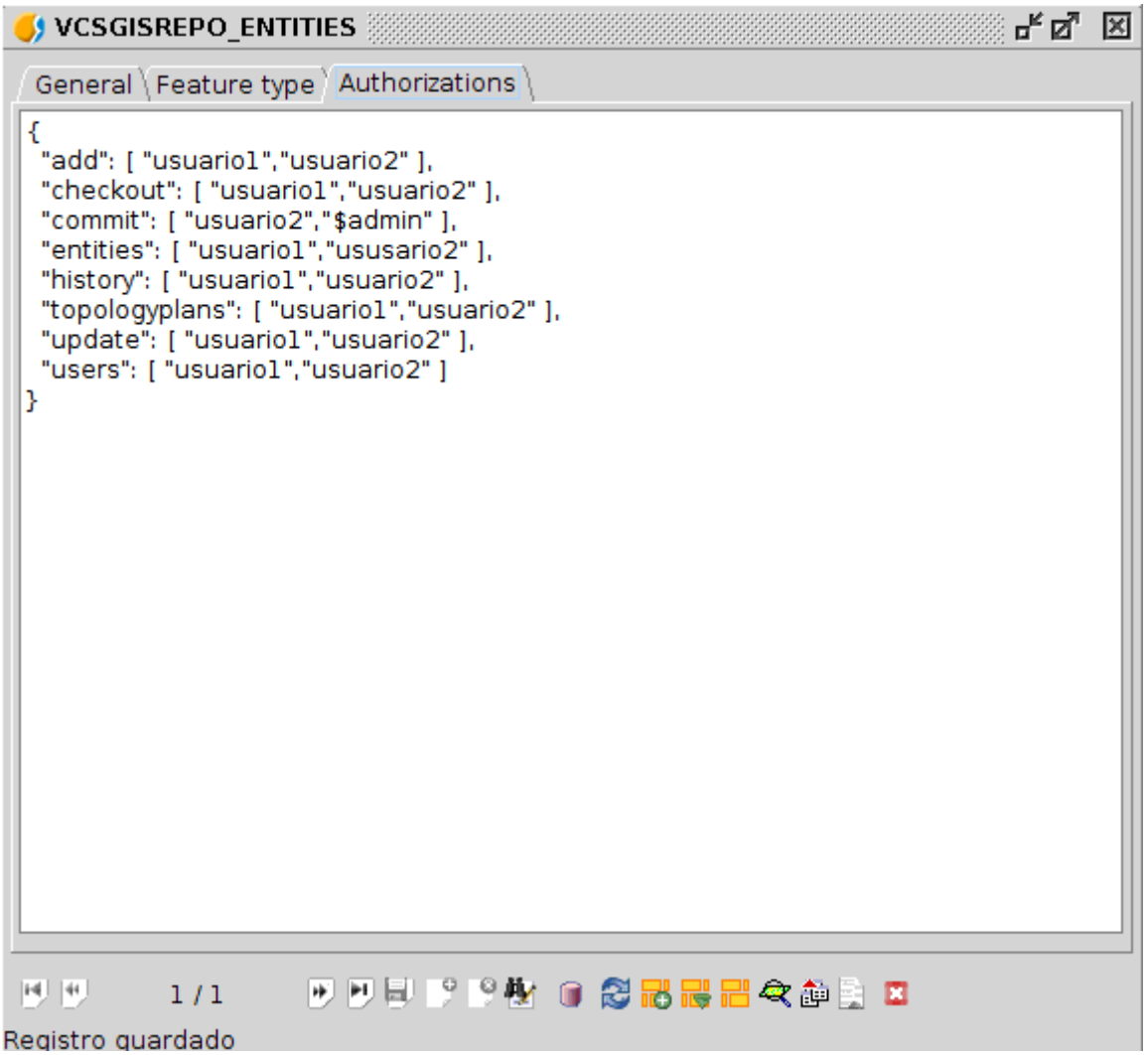




El *usuario1* presenta ahora el rol de administrador, *admin*.

Con esta nueva posibilidad la configuración del nivel de seguridad autorización avanzada ofrece una nueva posibilidad. Esta nueva posibilidad no es otra que permitir en la definición de la acciones y usuarios permitidos para cada dato permitir no solo usuarios concretos, sino grupos de usuarios o roles.

Para hacer lo anterior solo hay que realizar la modificación de la entidad de igual manera que al inicio del apartado, y en el panel de la pestaña *Authorization* del formulario de la tabla *PUBLIC.VCSGISREPO_ENTITIES* en las diferentes acciones especificar también la palabra que identifique el rol con un símbolo `$` previo. La siguiente ilustración muestra dicho cuadro de diálogo para la entidad seleccionada incluyendo que todo miembro del rol *admin* pueda realizar la acción *commit*.



Tras lo anterior solo hay que guardar cambios en la entidad y terminar edición en la tabla. A partir de ese momento los usuarios con el rol *admim* pueden realizar *commits* en dicha entidad.

Asignación de recursos a una entidad

En *gvSIG desktop* podemos asociar *recursos* a una capa o tabla. Estos recursos, cuando estamos trabajando con tablas de tipo fichero, como puede ser un shape, se almacenan junto a este. Asi por ejemplo, podemos asociar a un shape, un fichero “.cpg”, con la informacion sobre la codificacion a usar para los textos, un fichero “.gvsleg”, para que tenga una leyenda por defecto cuando lo carguemos, un fichero ".gvslab o “.sld” para que tenga asociado un etiquetado por defecto, “.frm” en el que especifiquemos que aspecto debe tener el formulario asociado a la tabla, incluso podriamos asociar informes personalidados, “.report”. Todos estos tipos de ficheros con informacion complementaria a los datos de la tabla se consideran recursos asociados a ella. Siendo los recursos citados solo una muestra de los que podriamos asociar a una tabla.

Ahora bien, cuando estamos trabajando con VCSGis, las tablas se encuentran almacenadas en un repositorio remoto, y se descargan a una base de datos local para trabajar con ellas, entonces… ¿ podemos asociar recursos a estas tablas que estan en un base de datos bajo un sistema de control de versiones ?

VCSGis provee mecanismos para poder asociar esos recursos, y almacenarlos en el propio control de versiones de forma que estos se descarguen automaticamente a la copia local cuando se descargan las tablas. Para hacer esto, es preciso la intervencion del administrador del repositorio de versiones, y una vez configurado como gestionar los recursos de cada tabla, los usuarios ya podran acceder a ellos de forma trasparente. Con esto se garantiza que todos los usuarios ya no solo dispongan de los datos de una misma tabla, sino también del modo de representar esta asi como de los ficheros de recursos que esten asociados a ella.

Tabla de recursos

En VCSGis, La tabla de recursos es una tabla formada por dos campos, *nombre* y *value*. El campo *nombre* de tipo `String`, identifica de manera única el recurso y la capa/tabla a la cual esta asociada, mientras que el campo *value* de tipo `byteArray` almacena el recuso en sí, es decir el archivo. Para realizar la asignación de este recurso, previa a su creación y actualización con los diferentes recursos, hay que realizar modificaciones en la entidades del repositorio de la tabla **VCSGISREPO_ENTITIES** asignándole a cada una de de las tablas o entidades del repositorio en que tabla estan almacenados los recursos de ella.

Ejemplo asignación recursos

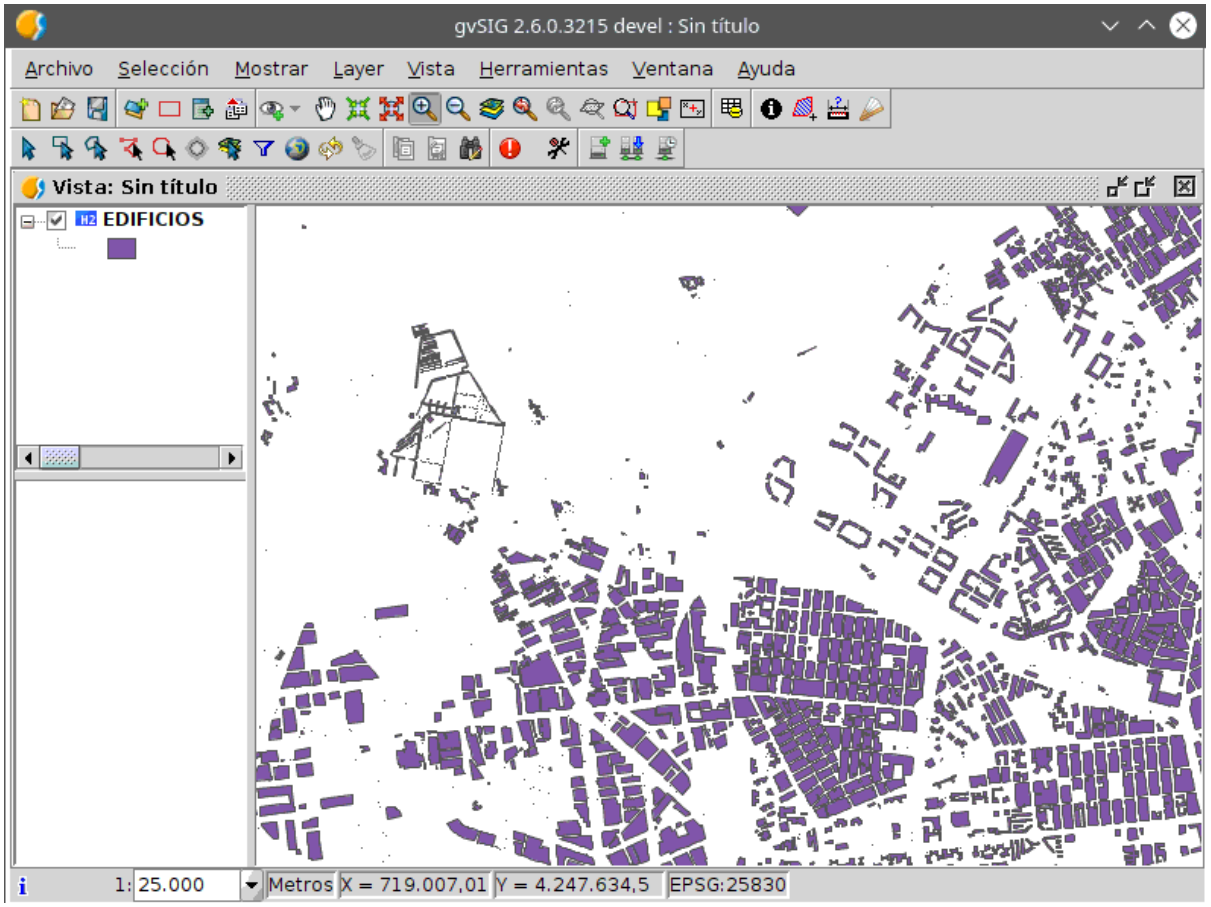
Para facilitar la comprensión del proceso de asignación de recursos a una determinada capa a continuación se detalla un ejemplo. Este se basa asignar una leyenda de colores a una capa presente en un repositorio.

La capa utilizada para la realización del ejemplo llamada *EDIFICIOS* es un archivo `shape` que almacena la geometría de los edificios de la ciudad de Alicante. Esta capa presenta las siguientes características;

- Repositorio: *aytoALC*
- Copia de trabajo: *usuario1*

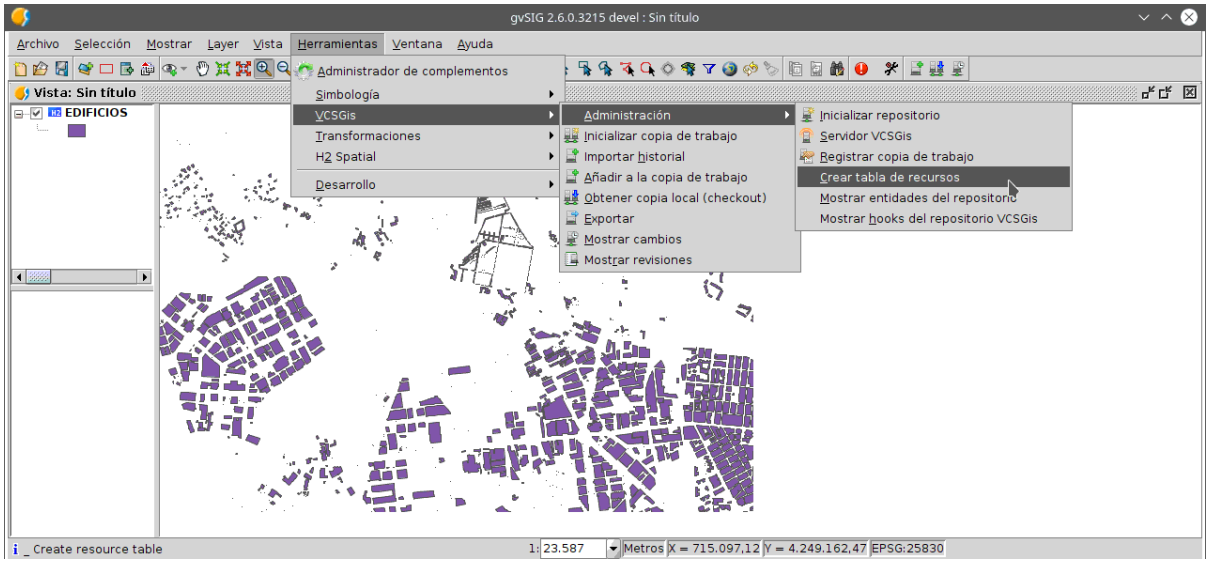
- Categoría: *BASE*

La siguiente ilustración muestra la capa cargada en *gvSIG Desktop*.

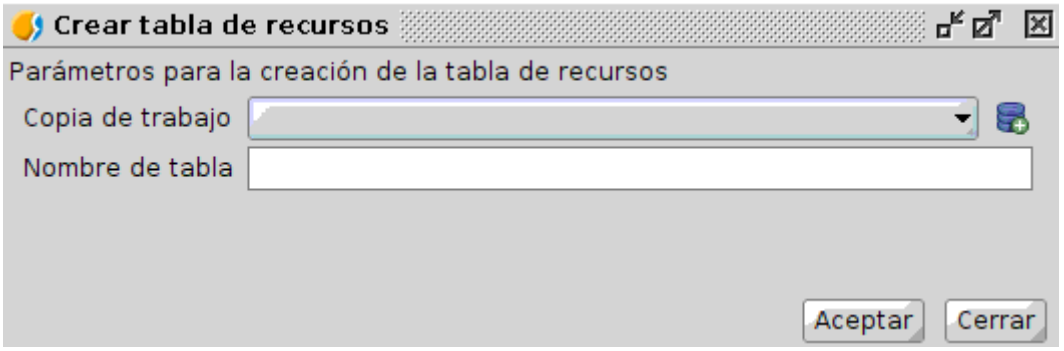


Tal y como se puede ver, la simbología presenta una leyenda por defecto que no se corresponde con la utilizada para esta capa. Para cambiarla y registrarla en el repositorio de modo que siempre que se utilice dicha capa se presente con la leyenda correcta hay que registrar dicha leyenda como recurso.

En primer lugar hay que crear la tabla de recursos. Dicho proceso se realiza ejecutando la opción *Crear tabla de recursos* situada en el menú *Herramientas*, submenú *VCSGis*, submenú *Administración*.



Realizar lo anterior muestra el siguiente cuadro de diálogo.



En la ventana hay que seleccionar en el primer desplegable llamado *Copia de trabajo* la copia de trabajo donde esta la capa, *usuario1*.

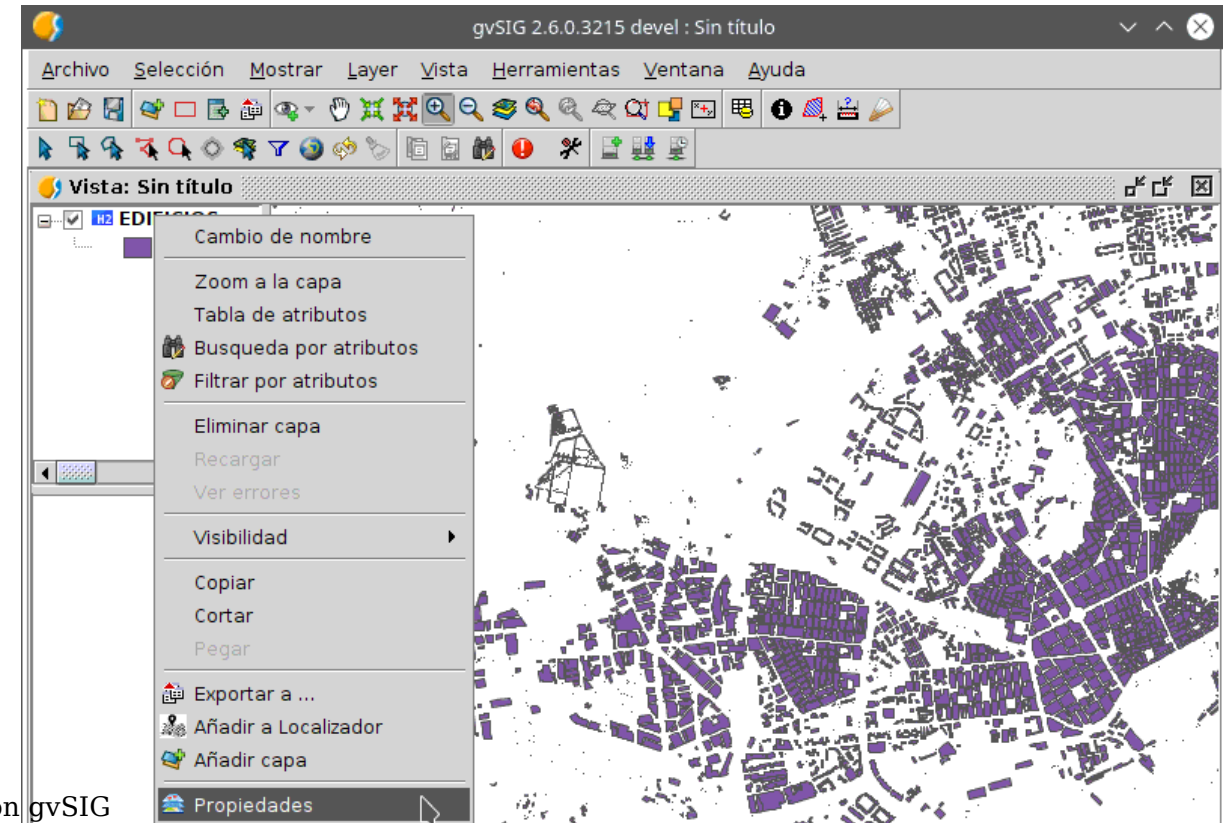
En el segundo componente hay que detallar el nombre de la tabla de recursos. Se nombra la nueva tabla de recursos como *BASE_RESOURCES* ya que se pretende que todos los recursos de la categoría *BASE* se encuentren en dicha tabla.

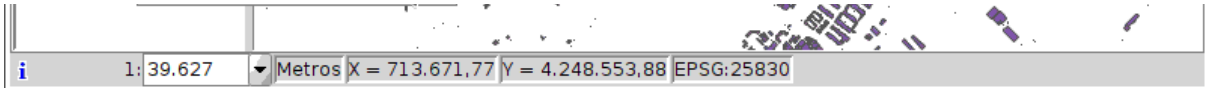
> Hay que especificar que se pueden crear tantas tablas de recursos como se deseen. De modo que se puede crear una única tabla de recursos que almacene todos ellos o varias de estas que almacenen los recursos de determinadas categorías de datos por ejemplo.

El proceso de creación finaliza pulsando el botón *Aceptar* en la esquina inferior derecha de la ventana.

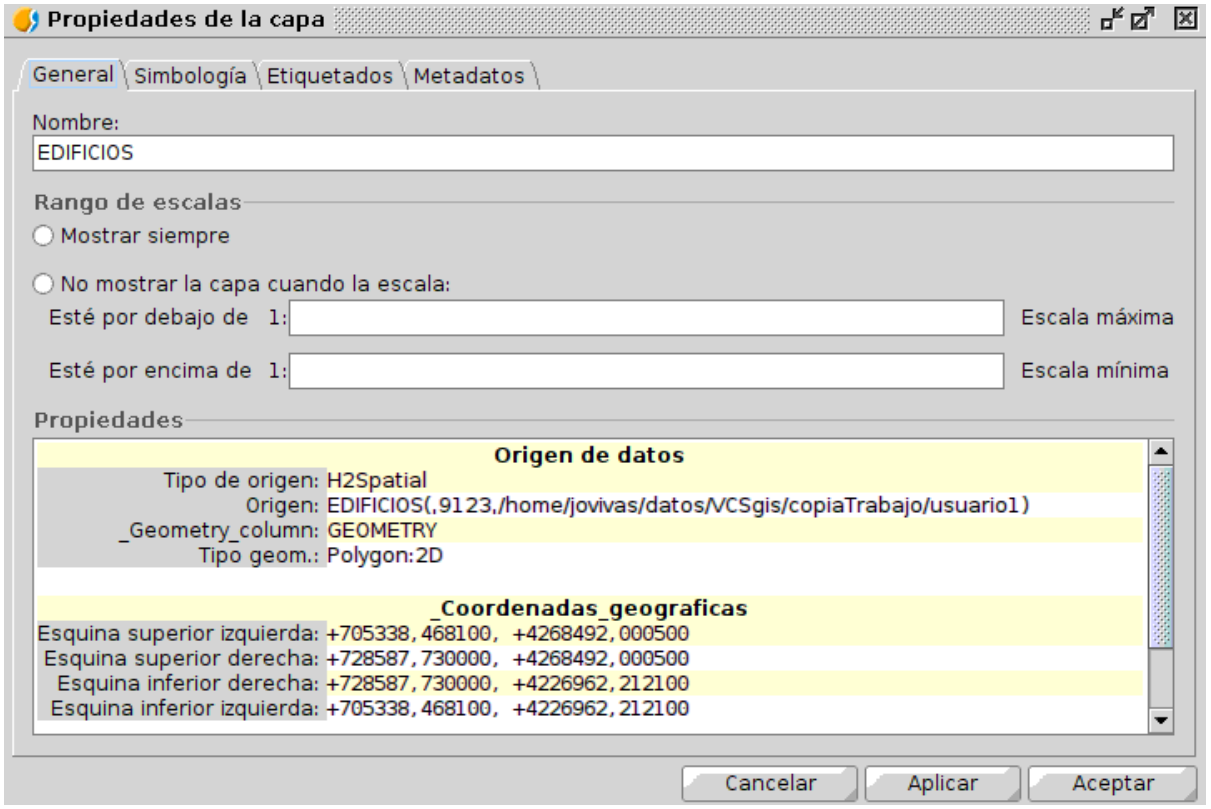
El siguiente paso a seguir es almacenar un recurso en la tabla recientemente creada. Para ello es necesario disponer de él o crearlo desde cero. En el caso del ejemplo se procede a crear de cero la leyenda para la capa *EDIFICIOS*.

Para crear una leyenda hay que ir al árbol de capas situado en el *Toc* de *gvSIG Desktop* y tras selección de la capa en cuestión pulsar botón derecho del mouse y seleccionar la opción *Propiedades*.

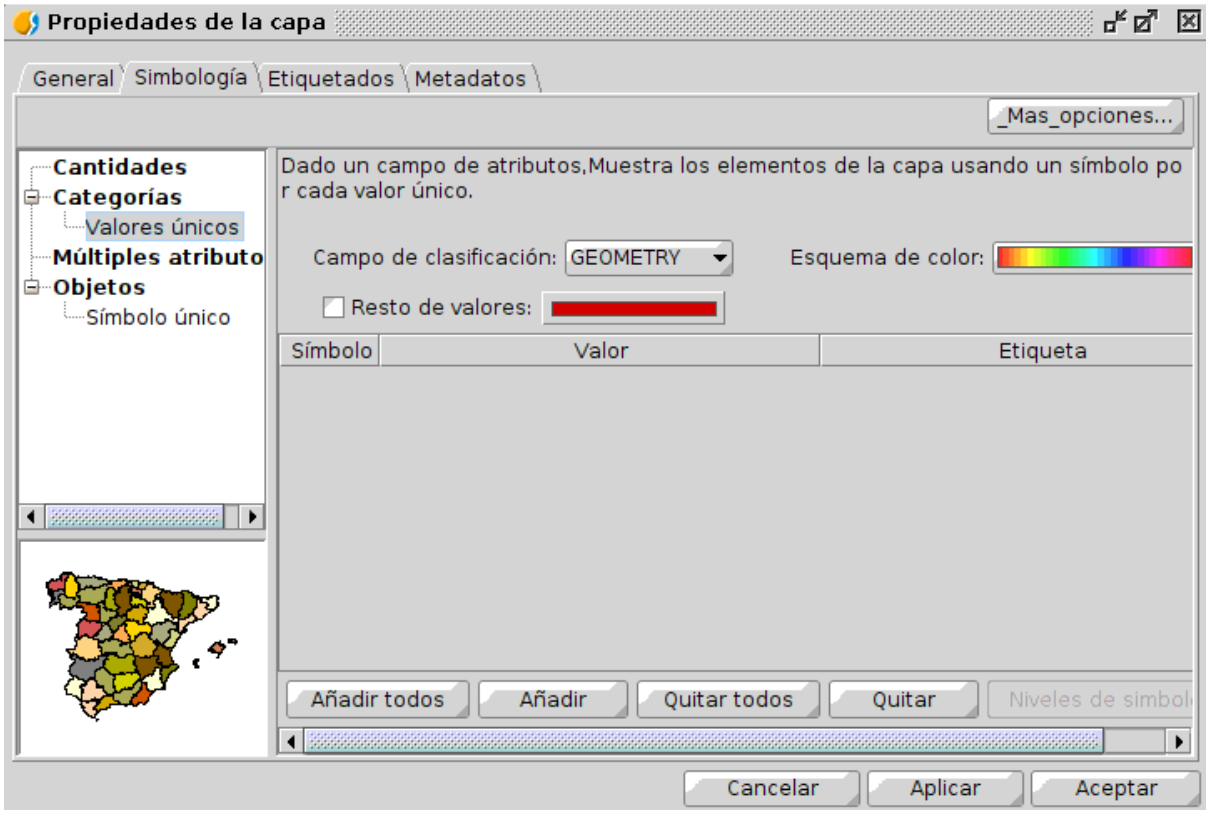




La opción anterior despliega la siguiente ventana.

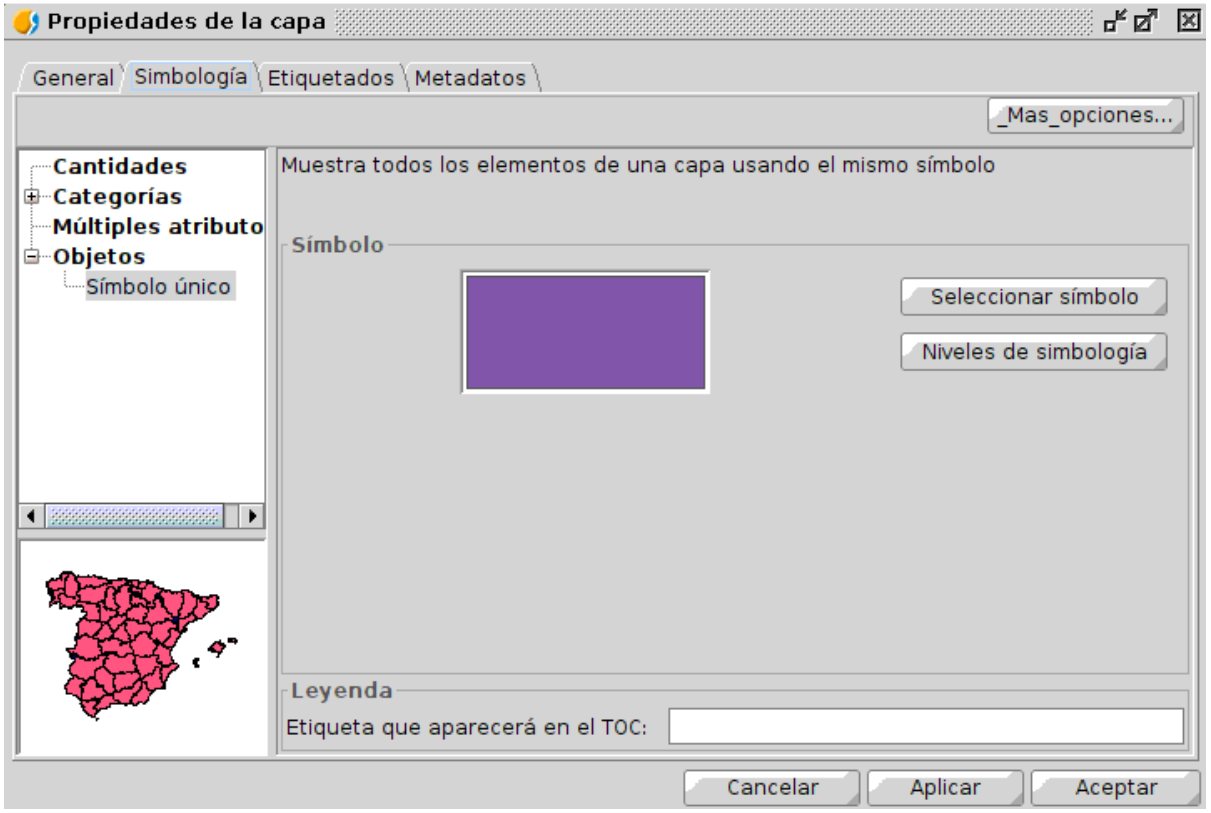


Se selecciona la pestaña *Simbología* pues en esta se especifica todo lo referente a la representación gráfica de la capa. El panel de de la pestaña anterior es el siguiente.

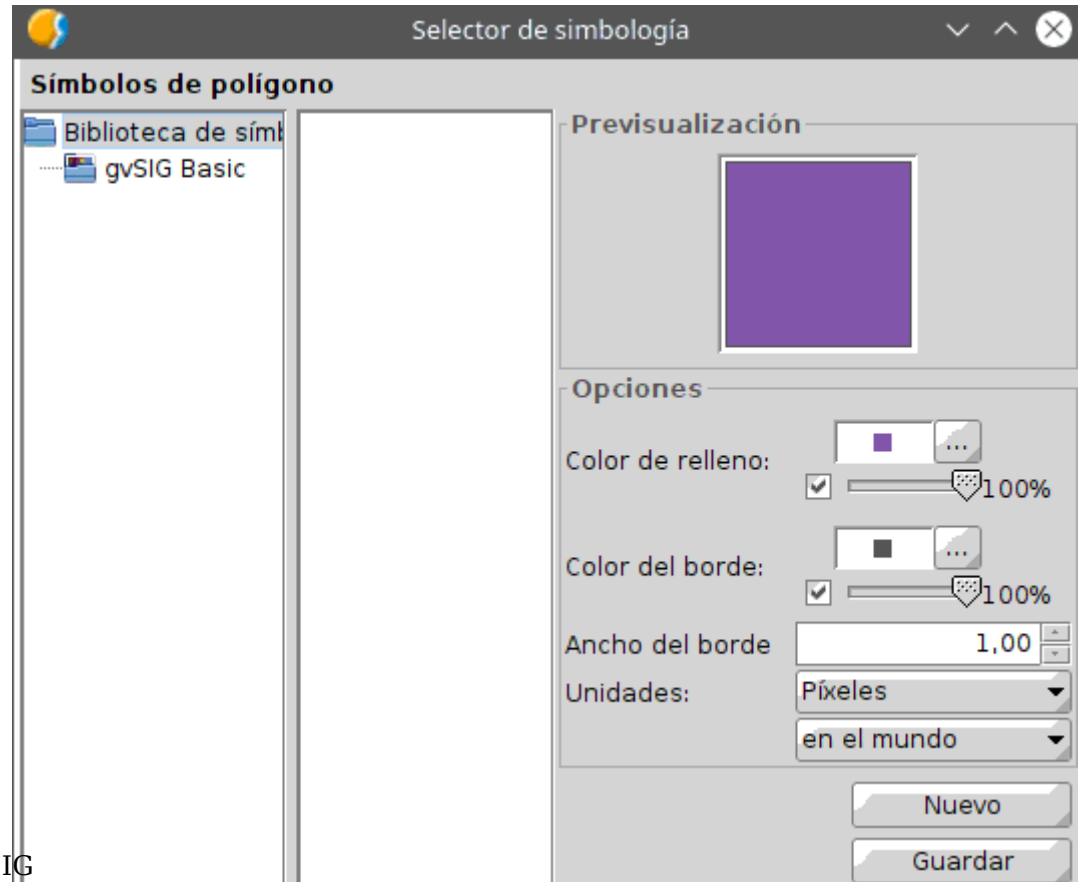


Una vez allí y continuando con el caso del ejemplo, se selecciona la opción *Símbolo único* de dentro de *Objetos* del cuadro de opciones de simbología situado en la parte izquierda del panel.

Dicha selección habilita el siguiente panel en la ventana.



En el panel se inicia el proceso de definición de nuestra leyenda pulsando el botón *Selección símbolo* el cual despliega el cuadro de diálogo *Selector de simbología*.

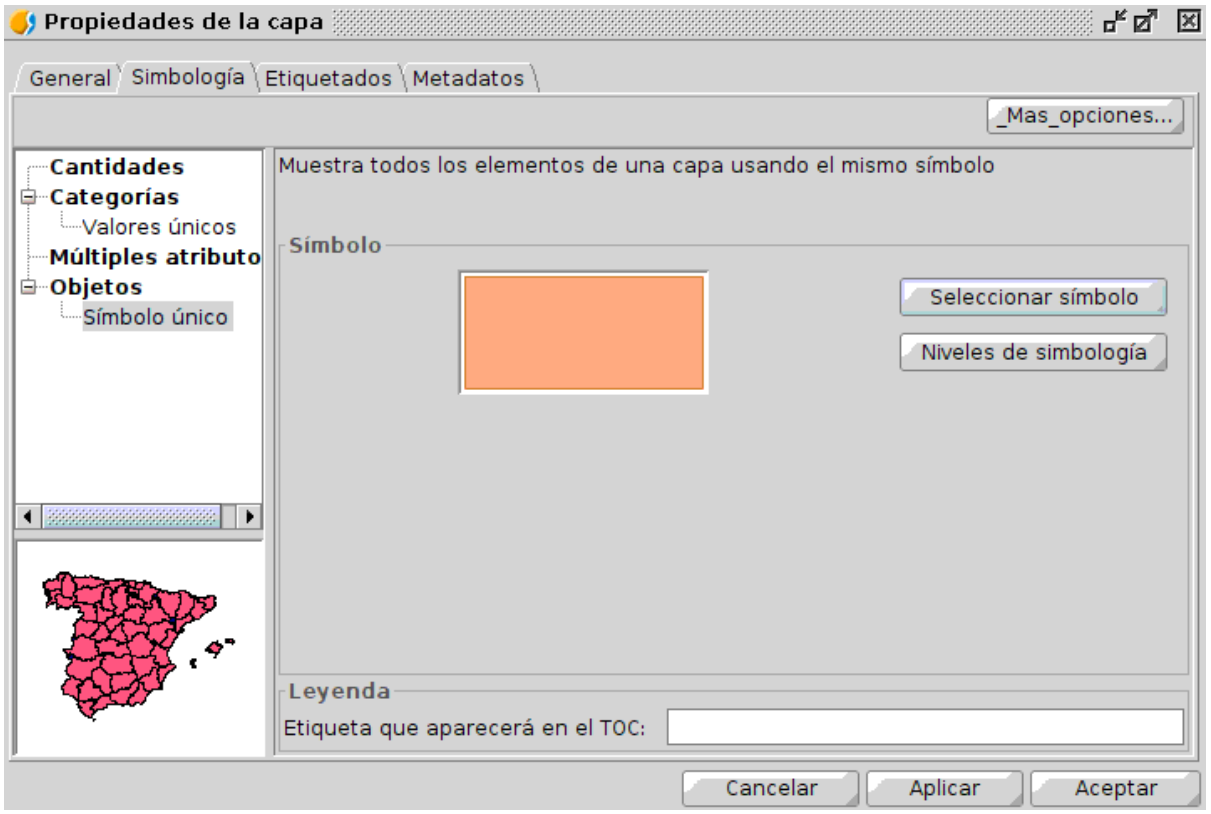


En esta nueva ventana hay que definir las características gráficas de los elementos de la capa, en este caso polígonos. Por lo tanto hay que definir como se van a representar sus bordes e interior. Los parámetros para los bordes e interior de los polígonos de la capa edificios se detallanan a continuación.

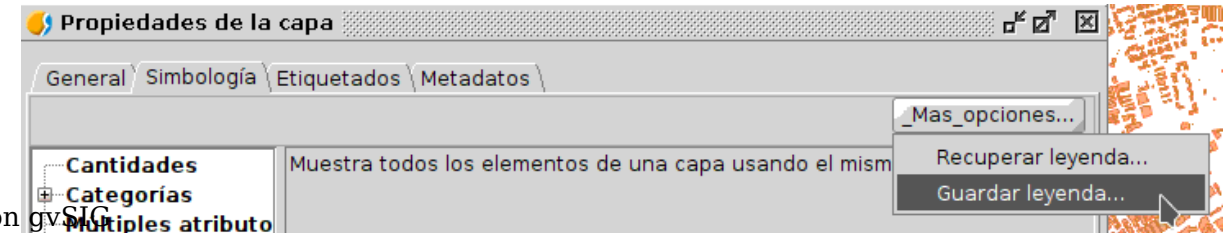
Componente	Color(RGB)	Opacidad	Ancho (píxel)
Borde	231 120 58	100%	1
Interior	248 190 132	100%	-

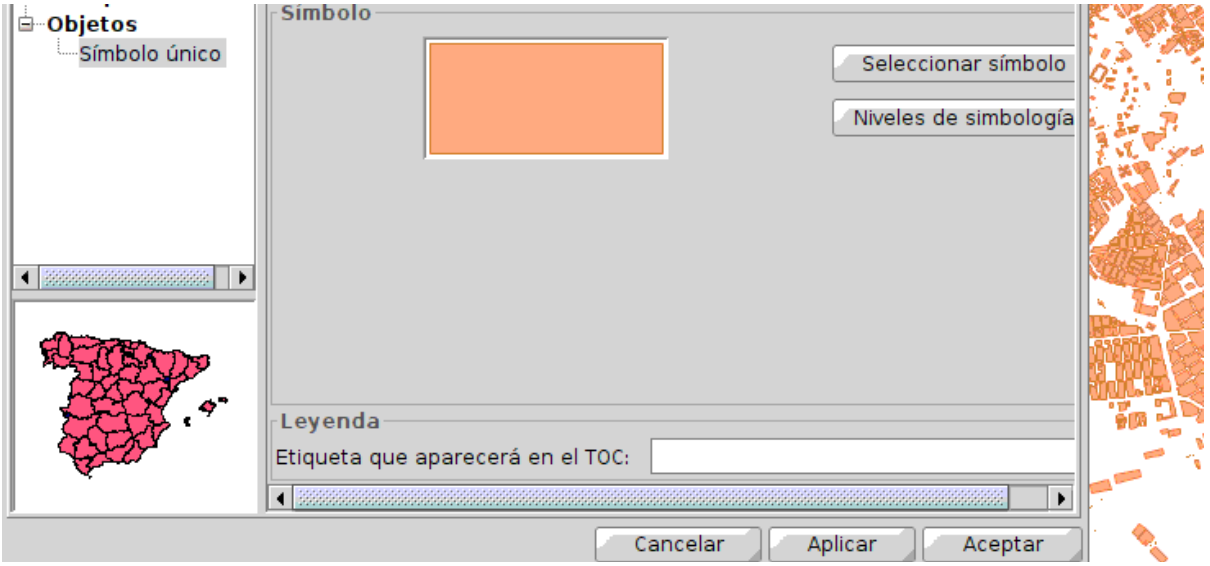
Una vez especificados se aceptan los cambios pulsando el botón *Aceptar* y se cierra el cuadro de diálogo

Tras lo anterior ya de nuevo en el panel de la pestaña *Simbología* de la ventana *Propiedades* de la capa *EDIFICIOS* se puede ver que el símbolo único ha cambiado según la definición especificada anteriormente.



Solo queda ahora almacenar esa simbología pulsando el botón *Más opciones* situado en la zona superior derecha de la ventana y seleccionar a continuación la opción *Guardar leyenda*.



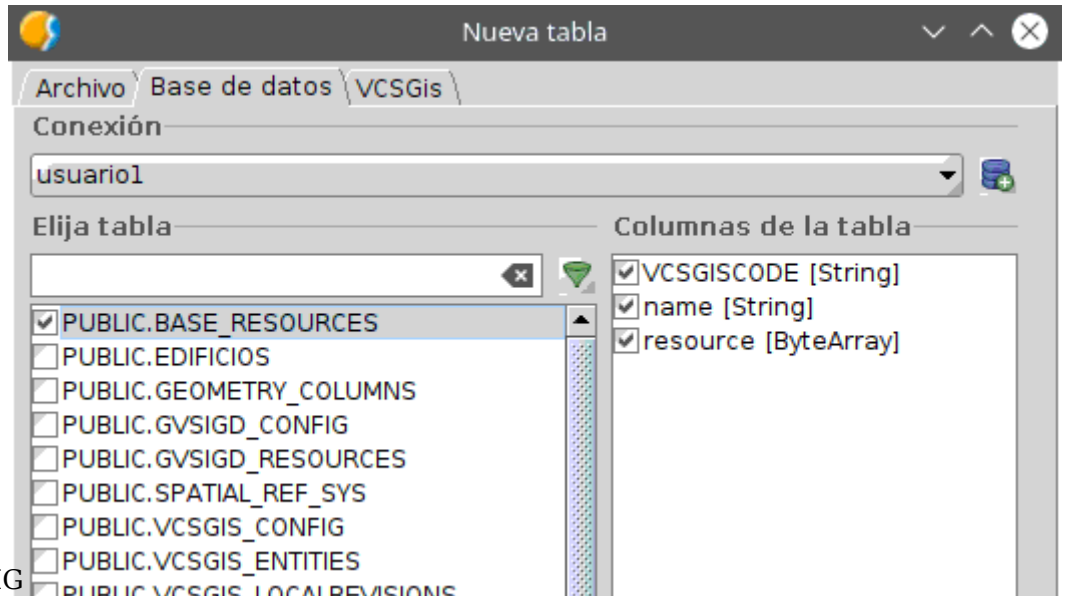


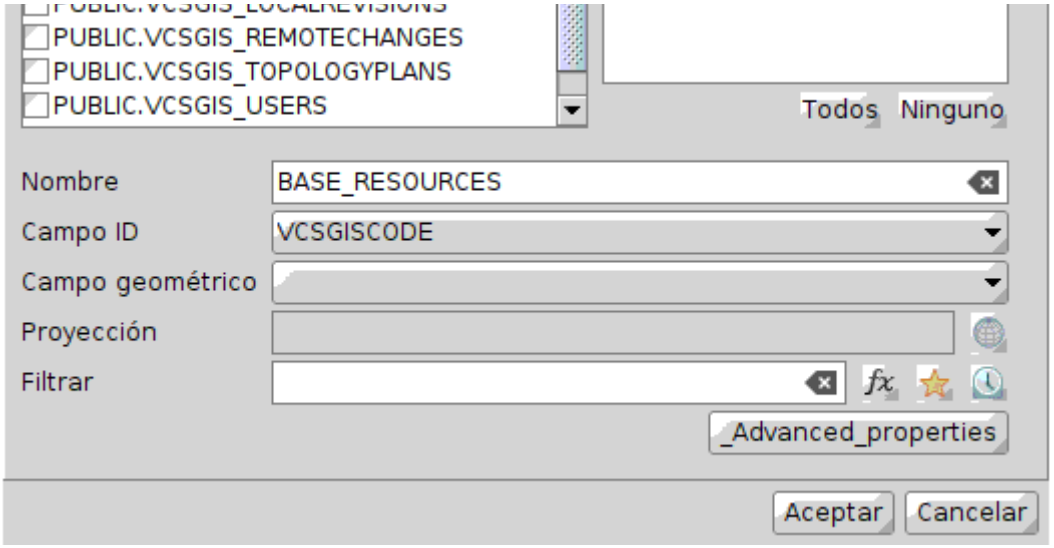
La opción anterior habilita un cuadro de diálogo que nos permite indicar la ruta donde almacenar la leyenda y el nombre del fichero que almacenará esta. En el caso del ejemplo se se almacena en una carpeta destinada a almacenar leyendas y el nombre del archivo es el mismo que la capa con la extension `.gvsleg`, es decir *EDIFICIOS.gvsleg*. Hay que destacar que el archivo con el recurso siempre tiene que llamarse de la misma manera que la capa de la que es recurso.

Una vez guardada la leyenda, pulsamos los botones *Aplicar* y *Aceptar* para terminar el proceso de definicion de la simbología de la capa.

Con el recurso ya creado hay que introducir este en la tabla de recursos, *BASE_RESOURCES*.

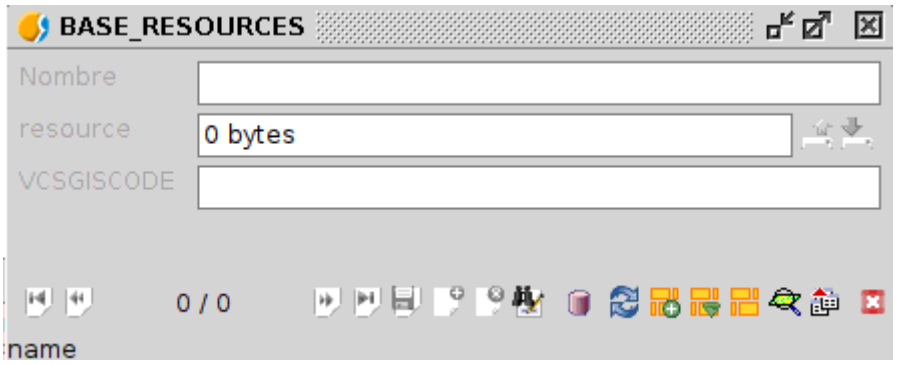
En primer lugar hay que abrir la tabla *BASE_RESOURCES*. Para abrir la tabla hay que realizarlo desde el *Gestor de proyectos* situado en el menú *Mostrar* de *gvSIG Desktop*. El proceso de abrir una tabla es el genérico a abrir cualquier archivo, primero se selecciona *Tabla* como tipo de datos a abrir, se selecciona la opción de *Nuevo*, lo que habilita una ventana donde se tiene que seleccionar la pestaña *Base de datos*. Esa pestaña muestra en su zona superior un desplegable donde hay que especificar la base de datos donde se encuentra la tabla, *usuario1*. Una vez seleccionada la base de datos, en la lista de tablas de esta hay que marcar la tabla en cuestión y pulsa el botón *Aceptar*.



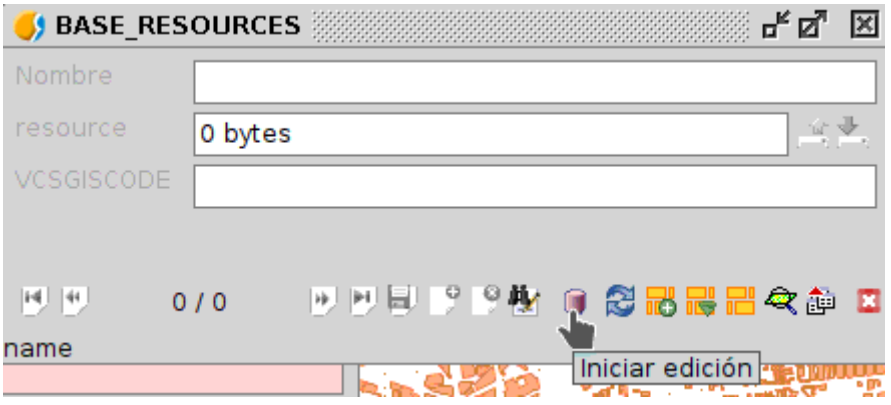


Como resultado se obtiene la tabla vacía. Para rellenarla hay que obtener su formulario, seleccionando la opción *Show form* situada en el menú *Tabla* de *gvSIG Desktop* siempre y cuando la tabla este abierta y seleccionada.

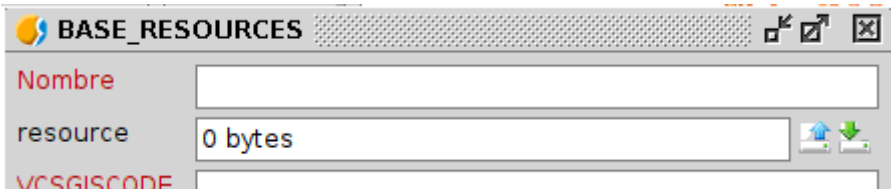
El formulario de la tabla *BASE_RESOURCES* es el siguiente.



Para introducir un valor en la tabla hay que poner esta en edición ya sea desde el mismo desplegable que se mencionó anteriormente para obtener el formulario, o desde el mismo formulario utilizando el botón *Comenzar edición*.

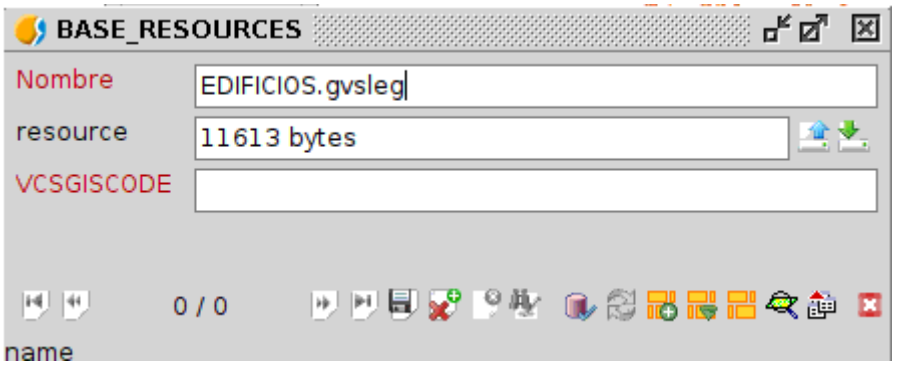


Una vez en edición hay que crear un nuevo elemento mediante el botón *Nuevo* situado en la zona inferior del formulario e indicado en la imagen siguiente.

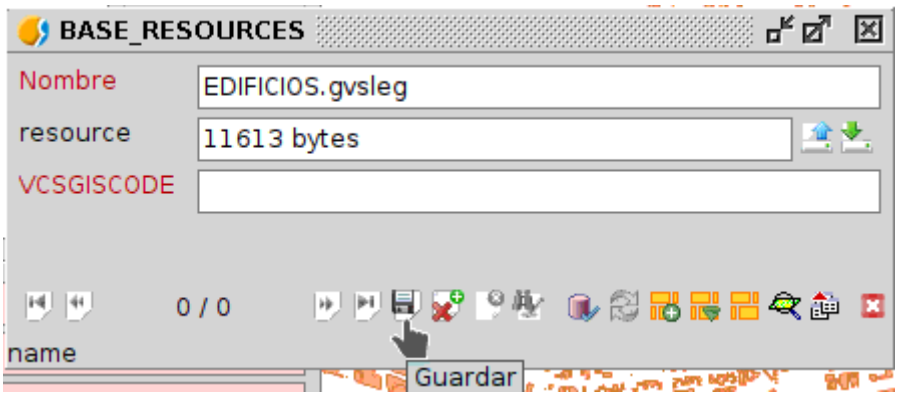


Especificamos en el campo *nombre* el nombre completo del archivo más la extensión del recurso que se desea almacenar y en el campo *value* gracias al botón izquierdo se puede subir a la tabla dicho fichero.

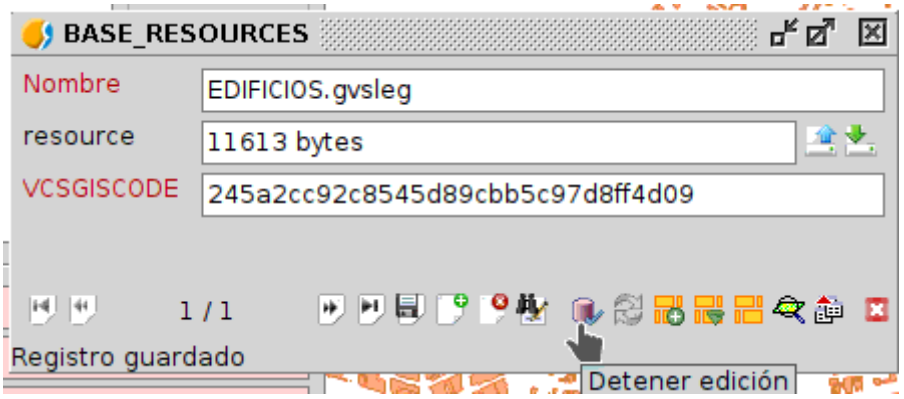
En el caso del ejemplo la configuración del formulario es la siguiente.



Una vez creado el nuevo elemento de la tabla solo queda guardar.

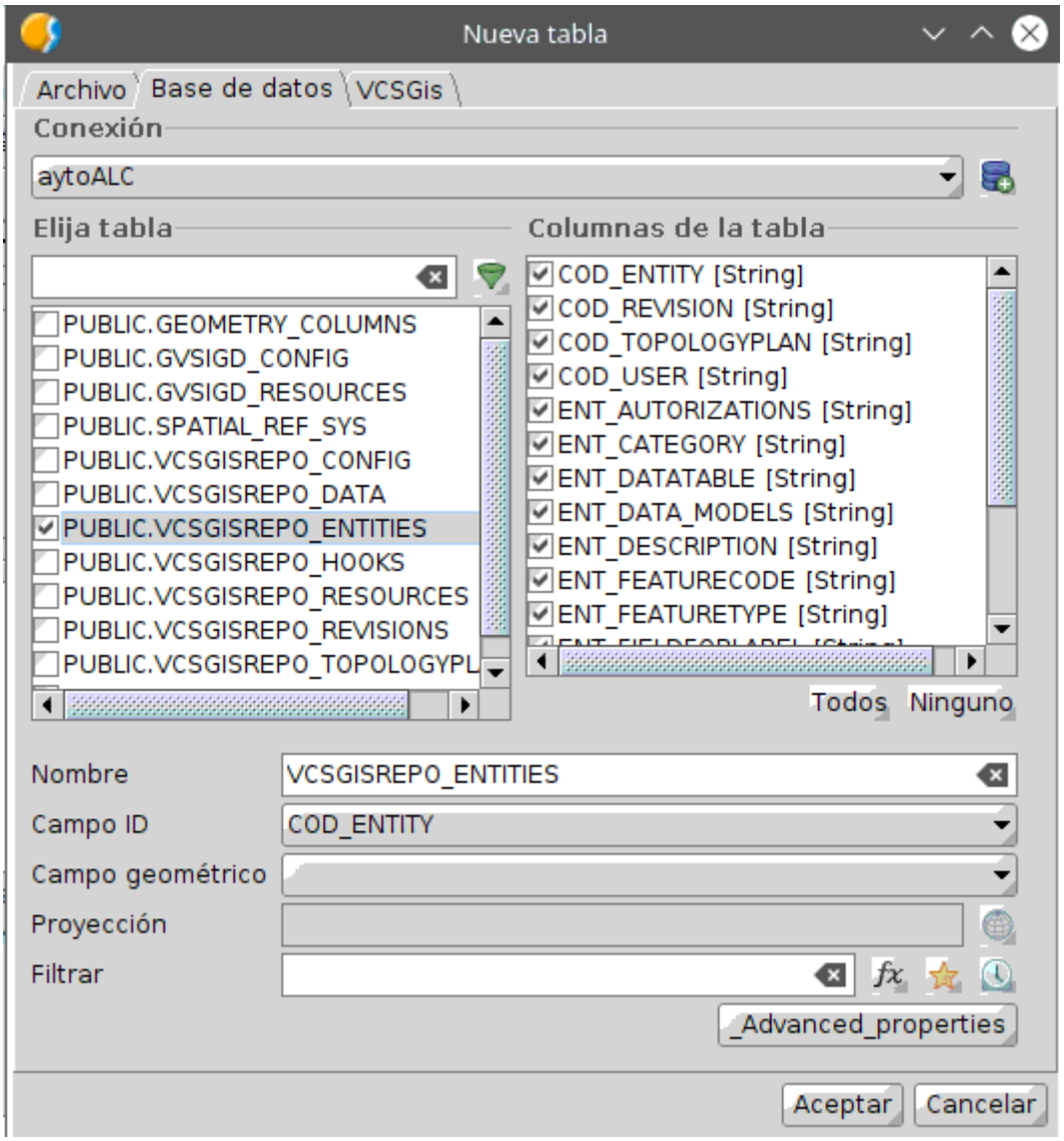


Y posteriormente finalizar la edición de la tabla.



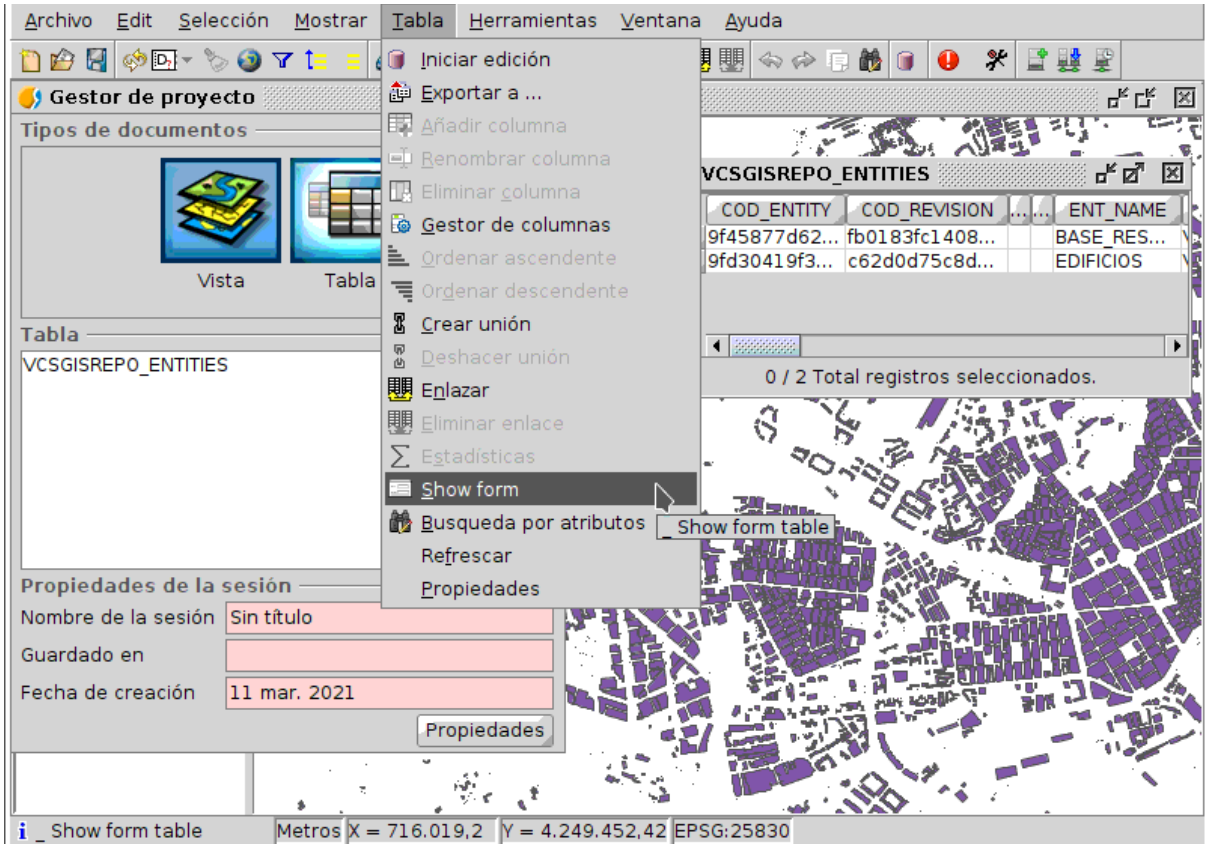
Tras crear y rellenar la tabla hay que subir esta al repositorio. Para hacer lo anterior hay que añadir la tabla a la copia local, opción *Añadir a la copia de trabajo* del menú *Herramientas*, submenú *VCSGis*, y posteriormente realizar un *commit* tras ejecutar la opción *Mostrar cambios* situada en el menú *Herramientas*, submenú *VCSGis*.

Tras la correcta definición de la tabla recursos hay que asignar a la capa *EDIFICIOS* del repositorio dicha tabla, para ello hay que abrir la tabla *PUBLIC.VCSGISREPO_ENTITIES* situada en el repositorio *aytoALC*. Para abrir la tabla hay que realizarlo desde el *Gestor de proyectos* situado en el menú *Mostrar* de *gvSIG Desktop*. El proceso de abrir una tabla es el genérico a abrir cualquier archivo, primero se selecciona *Tabla* como tipo de datos a abrir, se selecciona la opción de *Nuevo*, lo que habilita una ventana donde se tiene que seleccionar la pestaña *Base de datos*. Esa pestaña muestra en su zona superior un desplegable donde hay que especificar la base de datos donde se encuentra la tabla, *aytoALC*. Una vez seleccionada la base de datos, en la lista de tablas de esta hay que marcar la tabla en cuestión y pulsa el botón *Aceptar*.

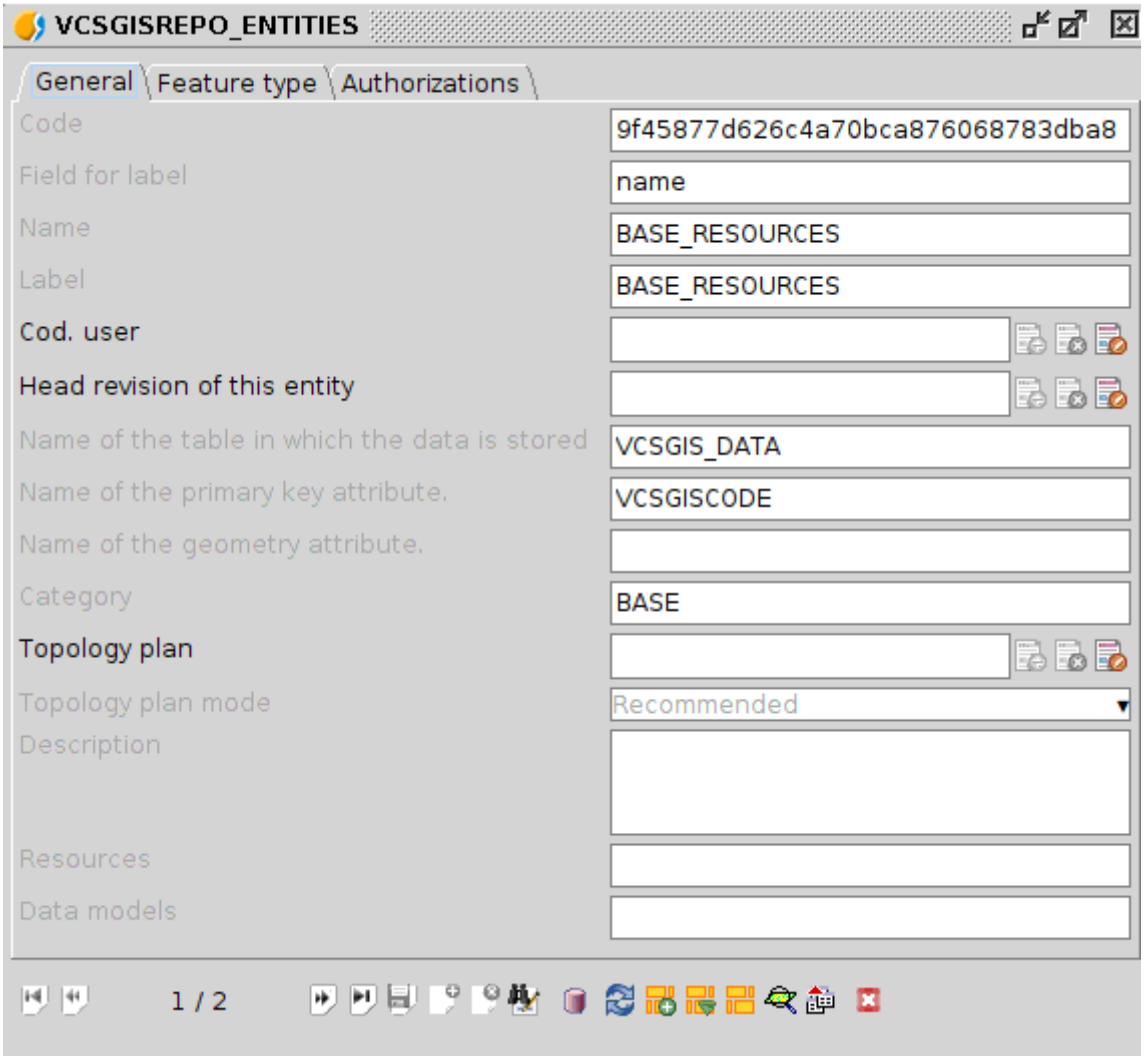


La siguiente imagen muestra la tabla *PUBLIC.VCSGISREPO_ENTITIES* donde por el memento hay dos entidades, la capa/tabla *EDIFICIOS* y la tabla *BASE_RESOURCES*.

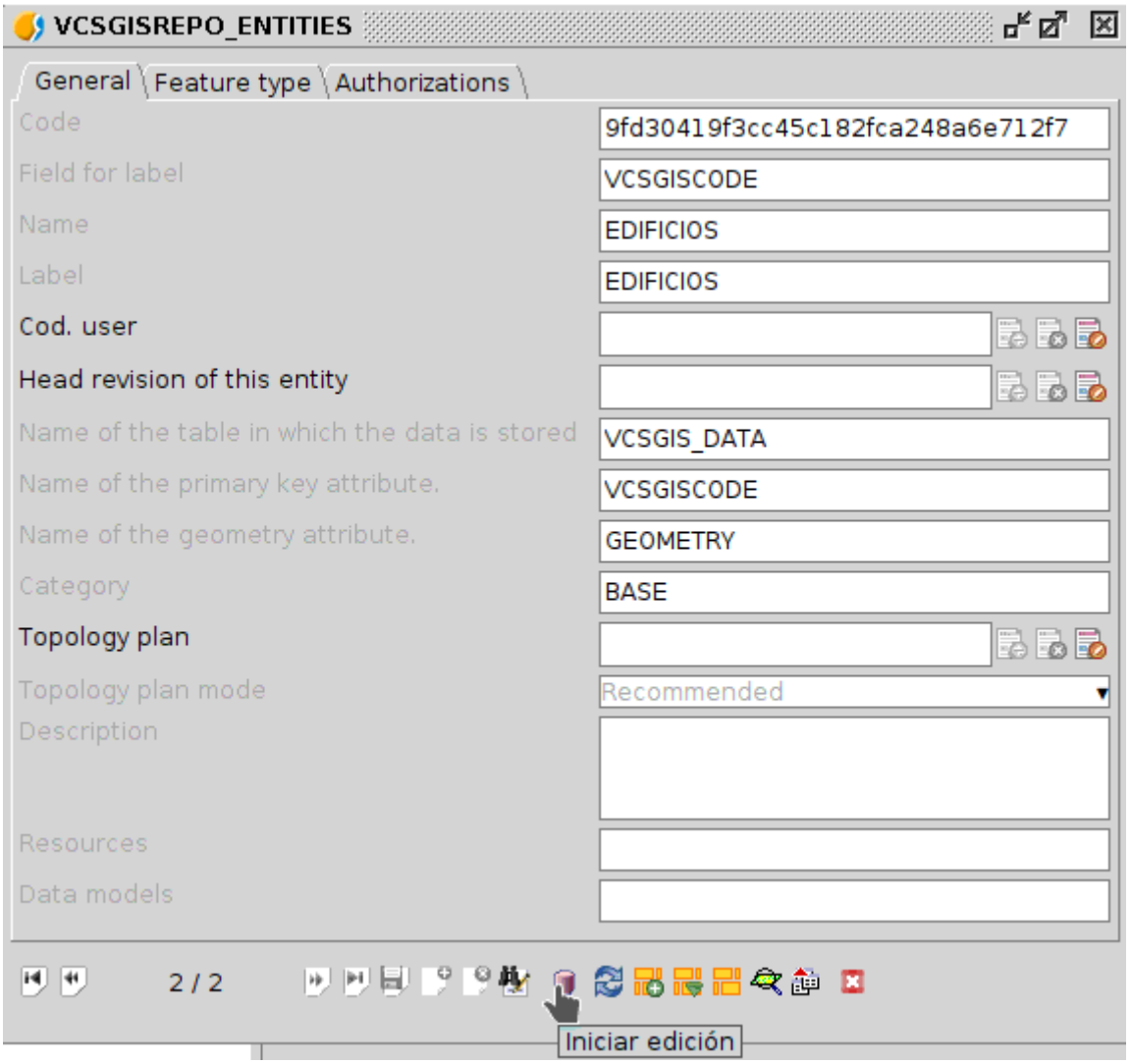
El proceso de asignación de los recursos se realiza modificando la entidad donde se desea asignar, en este caso la capa *EDIFICIOS*. Para ello es necesario obtener el formulario asociado la capa *PUBLIC.VCSGISREPO_ENTITIES*. Para obtener el formulario de la tabla seleccionaremos la opción *Show form* situada en el menú *Tabla* de *gvSIG Desktop* siempre y cuando la tabla este abierta y seleccionada.



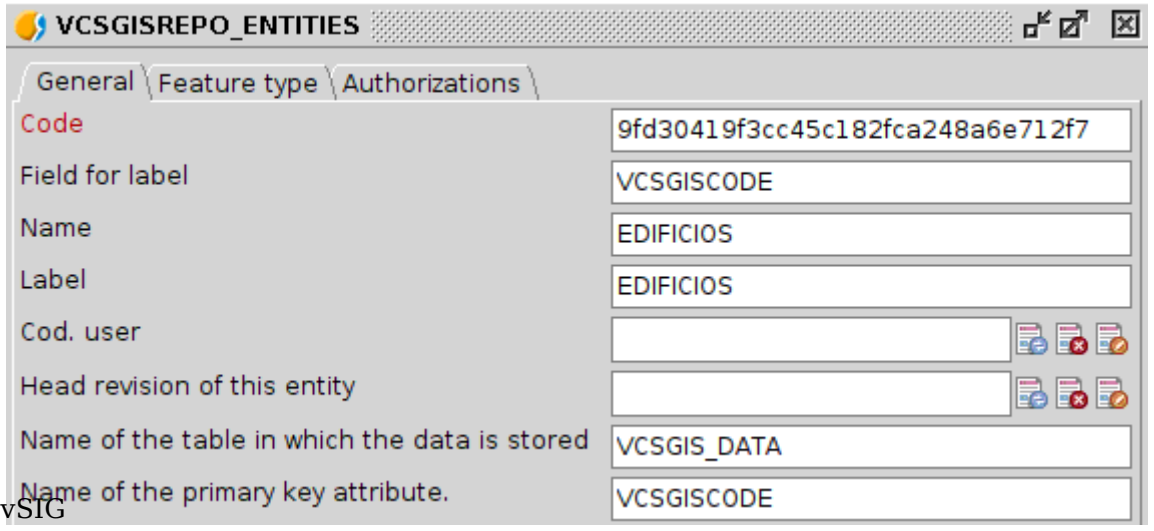
El formulario de la tabla es el siguiente.

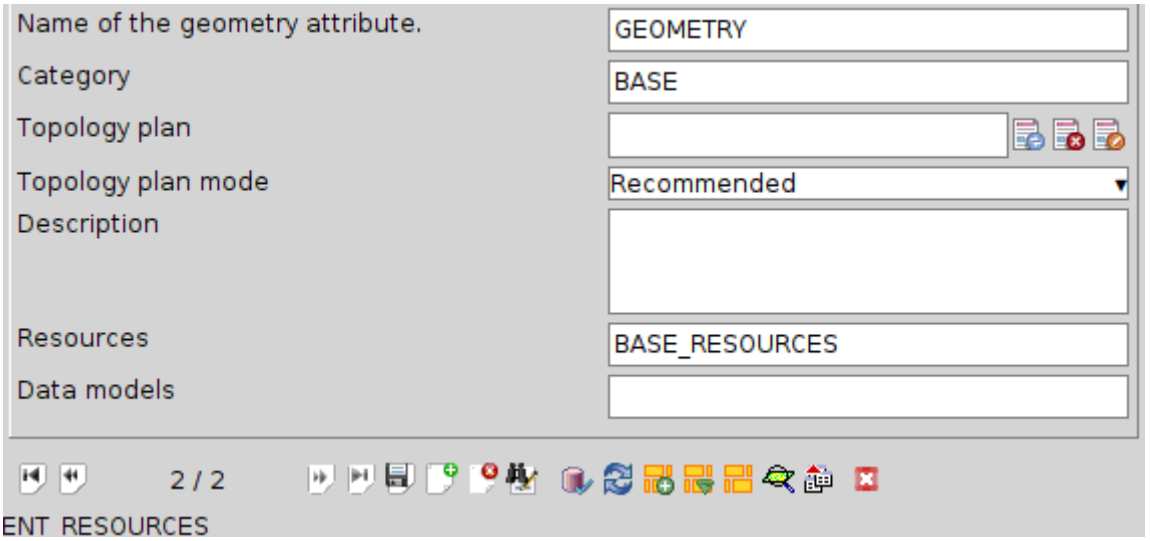


Una vez en el formulario se identifica el elemento que hace referencia a la tabla/capa *EDIFICIOS* y se inicia la edición de la tabla para la modificación de este. Este proceso se puede realizar desde el mismo desplegable que se mencionó anteriormente para obtener el formulario, o desde el mismo formulario utilizando el botón *Comenzar edición*.

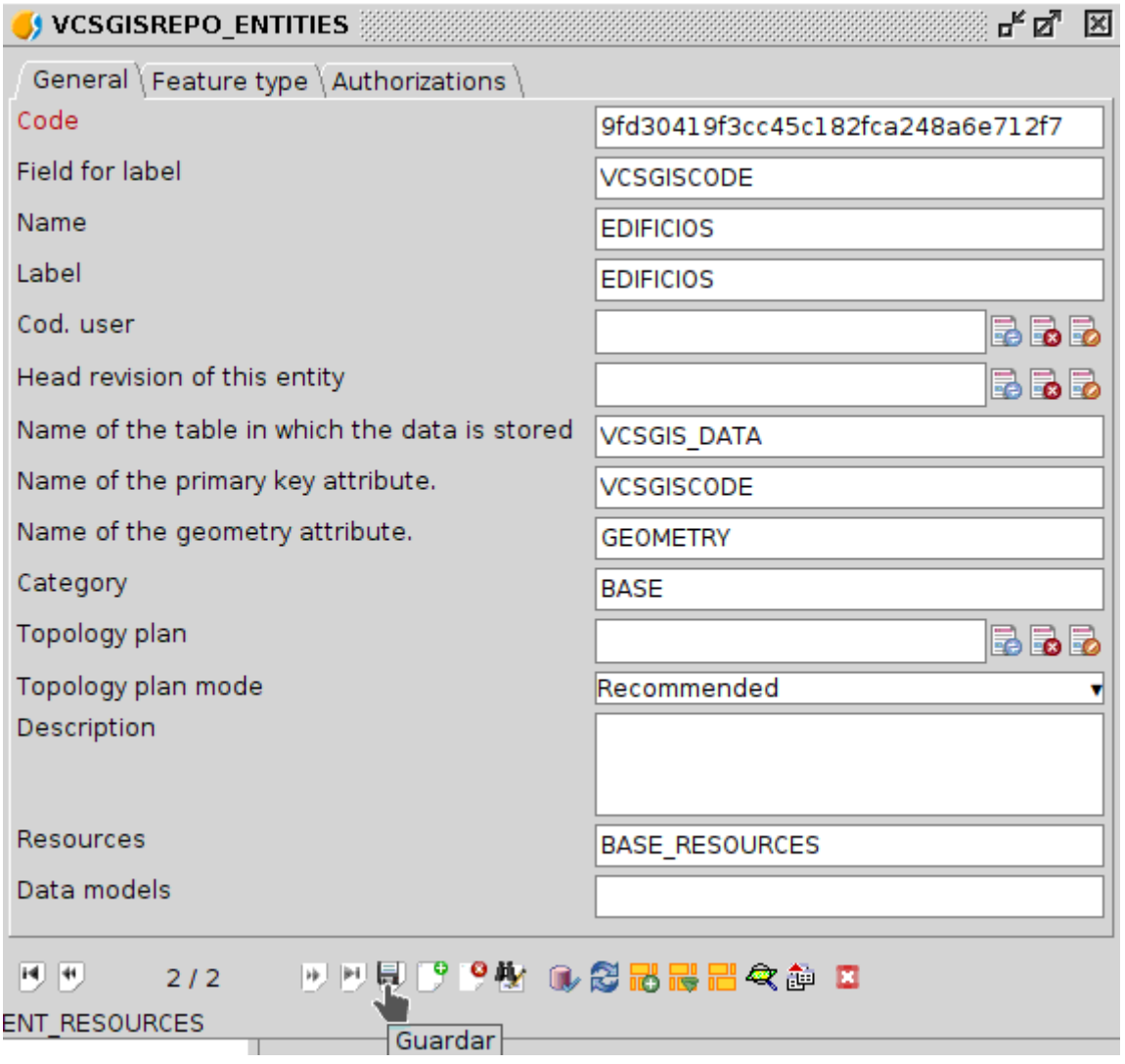


De los diferentes campos del formulario hay que identificar el refrente a los recursos, llamado, *Resources*. En este hay que especificar el nombre de nuestra tabla de recursos, *BASE_RESOURCES*, ya que la tabla/capa edificios pertenece a la categoría *BASE*.

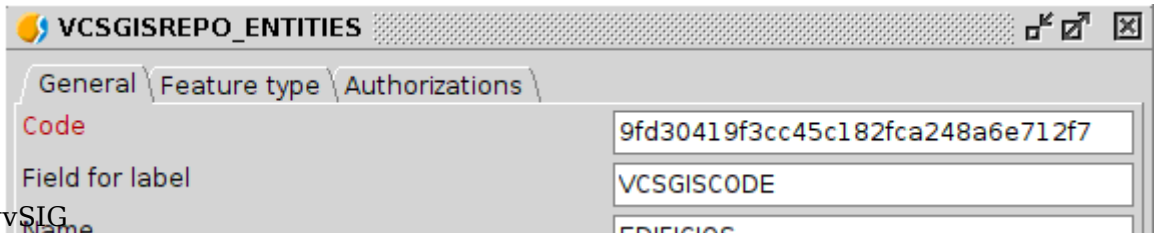


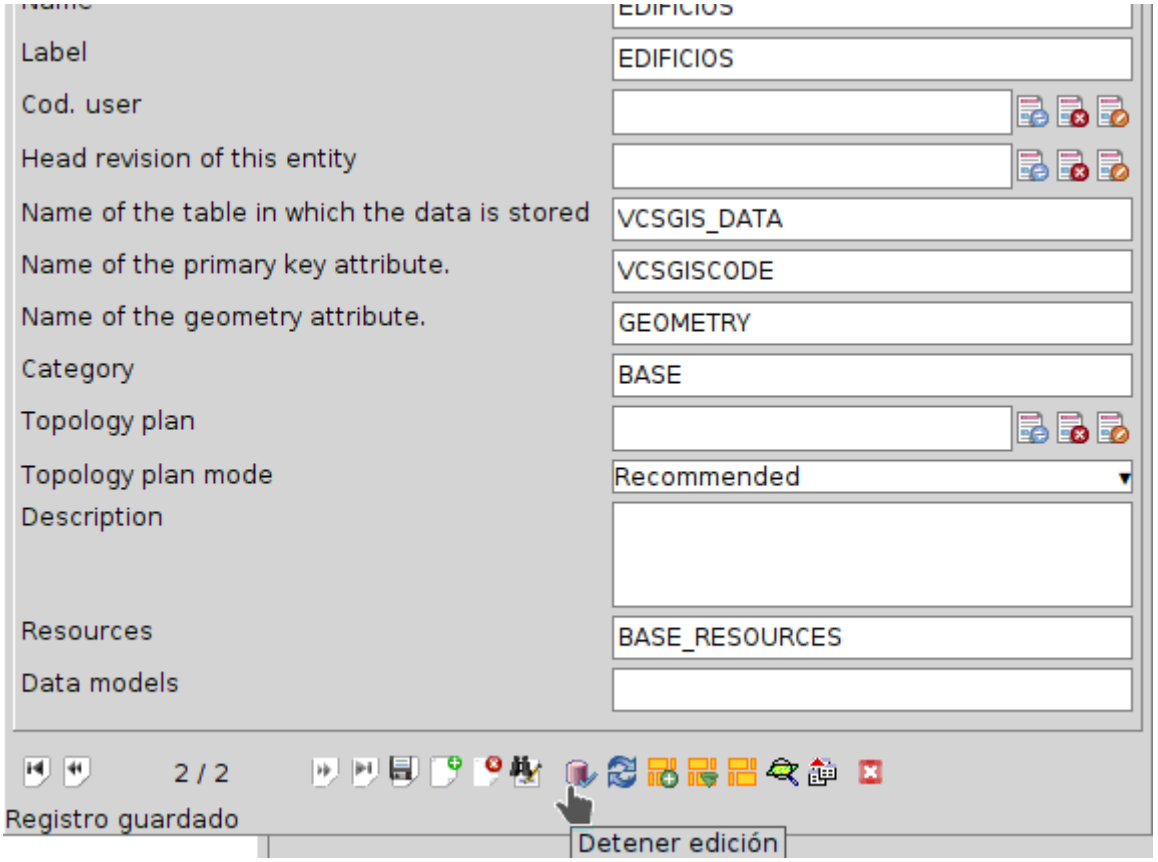


Solo queda guardar los cambios en la entidad.



Y terminar la edición de la tabla.





Tras lo anterior la asignación de los recursos mediante la tabla de recursos *BASE_RESOURCES* a la capa *EDIFICIOS* ha concluido.

Para confirmar el proceso de asignación de un recurso de la tabla de recursos a una capa solo hay que hacer checkout de esta tabla en cuestión, tal y como se indica en el apartado *Añadir una capa del repositorio*.

Tras todo lo realizado anteriormente cualquier usuario que realice una descarga del repositorio de la tabla *BASE_RESOURCES* y la capa *EDIFICIOS* presentará la misma leyenda asociada.

Modelo de datos

Trabajar con problemas complejos muchas veces precisa realizar un análisis adecuado, y definir un modelo de datos para representarlos de manera correcta. Normalmente el modelo de datos se materializa en un conjunto de tablas, registros, relaciones y restricciones de una base de datos relacional.

Con *gvSIG Desktop* se puede acceder a esas tablas, realizar consultas, visualizar sus datos y si tienen información geográfica representarla en un mapa. De modo que, es posible ver cada una de las tablas que conforman el modelo de forma independiente o ver ese conjunto de tablas como un modelo de datos coherente, como un todo. Si se adopta la última opción, hay que añadir a *gvSIG Desktop* información para que el sofware sea consciente de las relaciones que hay entre tablas, las restricciones establecidas para algunos de sus atributos, así como cual es la forma de acceder a cada una de las tablas del modelo de datos. Normalmente se emplean dos herramientas de *gvSIG Desktop* para realizar lo anterior, el *gestor de columnas* y los *repositorios de datos* o *espacios de trabajo de base de datos*.

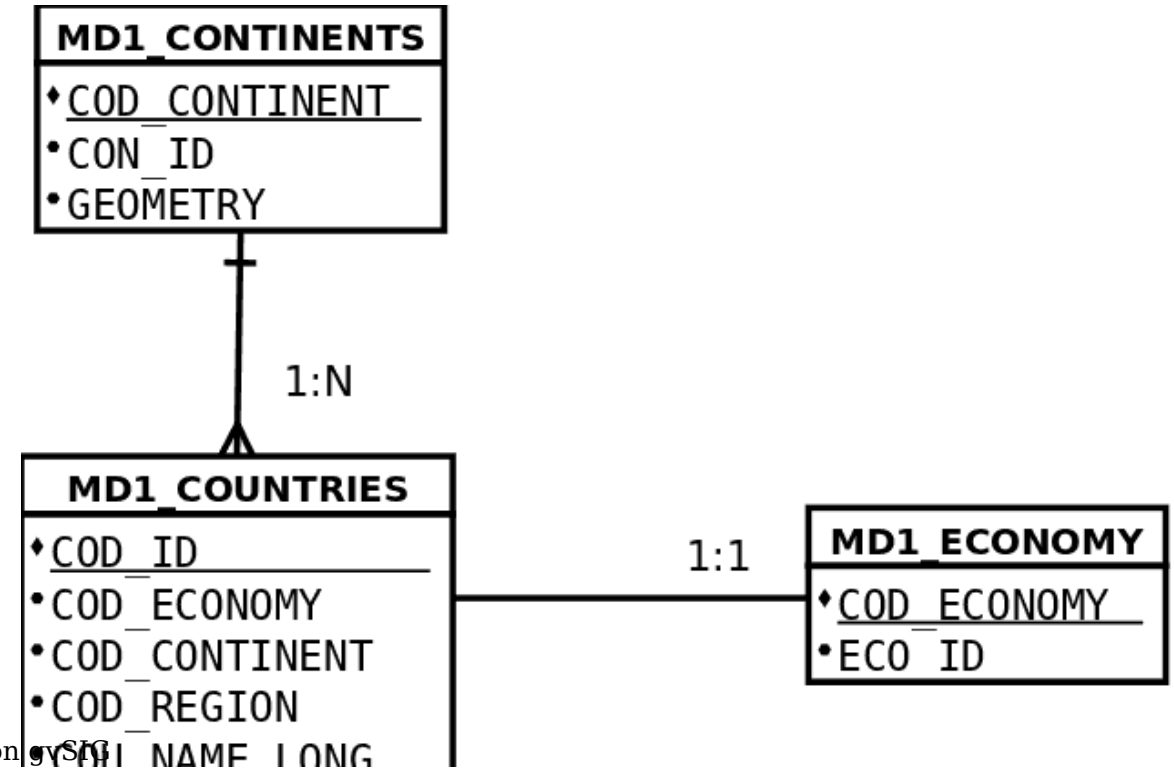
A través del *gestor de columnas* se declaran que relaciones existen entre las tablas. Por ejemplo, si se tiene una tabla de *facturas* y otra de *líneas de factura*, se define una relación entre ambas, indicando como obtener las líneas de una factura, o como obtener la factura asociada a una línea de factura. *GvSIG Desktop* permite declarar este tipo de relaciones entre las distintas tablas o entidades del modelo de datos, y luego cuando se precisa presentar esta información a traves de formularios, adapta la presentación para que se pueda navegar entre las relaciones declaradas. Además, es posible usar estas relaciones para poder realizar búsquedas entre las distintas tablas del modelo de forma sencilla y más o menos transparente para el usuario.

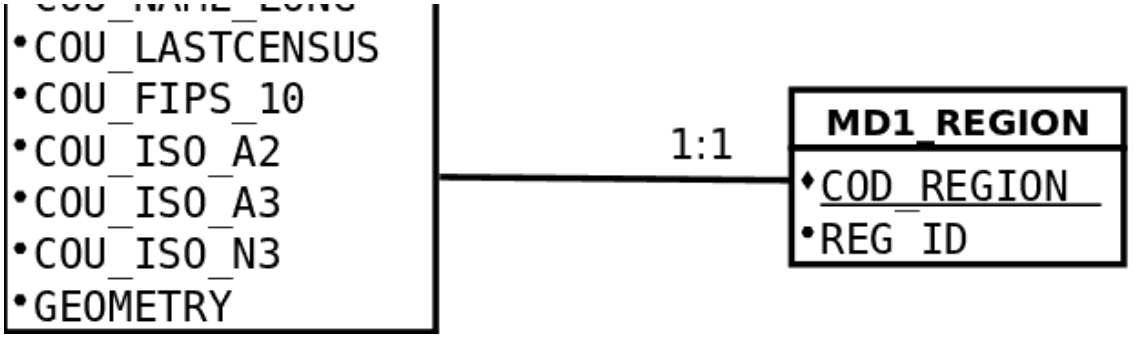
Pero para poder “navegar” entre las distintas tablas de un modelo de datos, no solo es necesario declarar las relaciones que existen entre ellas; también se debe declarar como se puede acceder a cada una de las tablas que componen el modelo. Para esto hay que definir un *repositorio de datos*, en el que se indica donde esta almacenada cada tabla del modelo y como acceder a ella.

Centrando la atención en *VCSGis*, se ha integrado una herramienta que ofrece la posibilidad de cargar desde el repositorio del control de versionesque tanto la información relacionada con las restricciones y relaciones tablas así como en donde se encuentran cada una de las tablas del modelo. De esta manera se puede almacenar la definición del modelo de datos para tablas ya integradas en el repositorio, pudiendo conectarse a ese modelo de datos, descargarlo y registrarlo de forma cómoda para el usuario.

Para ilustrar como configurar la parte de configuración de modelos de datos en el contexto de VCSGis vamos a trabajar un modelo sencillo, con solo unas pocas tablas.

Las tablas seleccionadas para realizar el ejemplo son *countrie*, *continent*, *region* y *economy*. Estas tablas representan paises, continentes, subregión y el tipo de región segun su economía respectivamente. Las relaciones establecidas entre capas se detrallan en la siguiente ilustración:





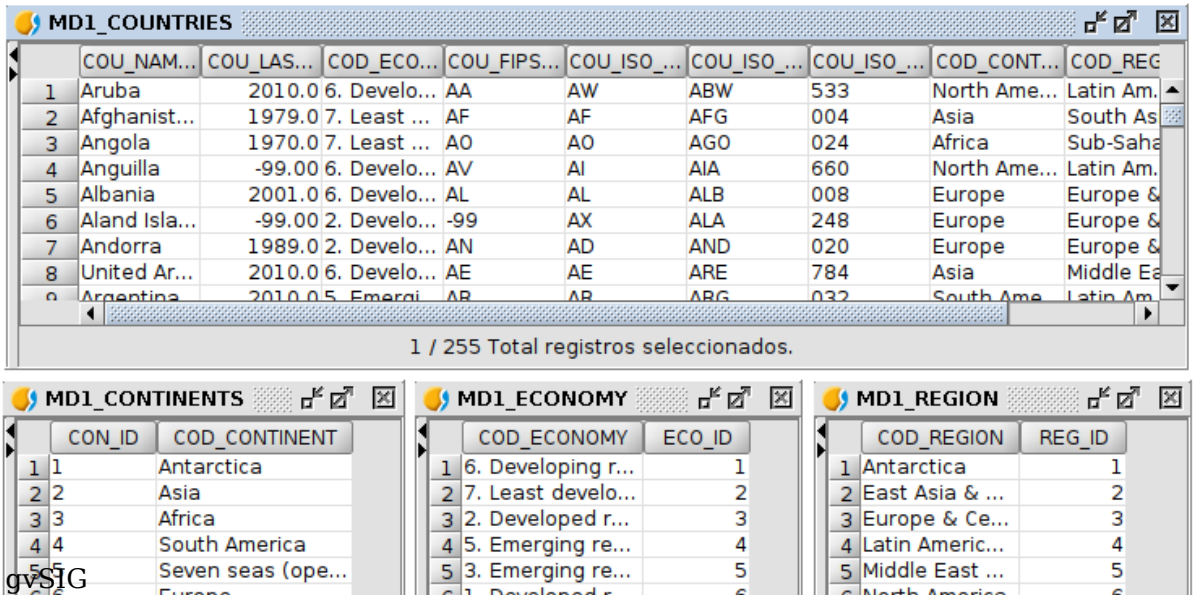
Configuración del modelo de datos en VCSGis

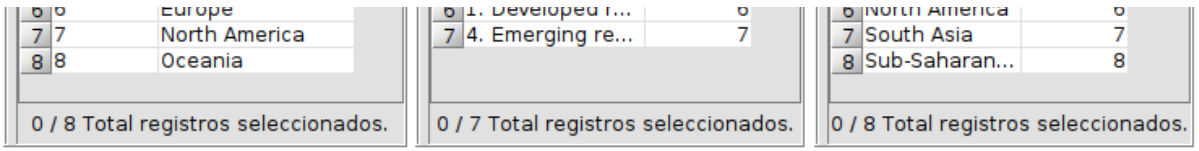
Configurar un modelo de datos en *VCSGis* requiere cierto análisis y organización. No esta pensado para crearse sin una planificación previa. Antes de añadir las tablas que forman el modelo se deben tener bien claras y declaradas las relaciones entre ellas, así como las restricciones que se buscan aplicar a los atributos de estas. Adicionalmente, también hy que declarar como se deben presentar los datos de las distintas tablas al usuario, normalmente en forma de *formularios*.

Para realizar toda esta tarea hay que apoyarse en las herramientas de gvSIG desktop de *Crear nueva tabla o capa* y el *Gestor de columnas*. Una vez declarada la información que define el modelo de datos, es posible añadir las tablas que lo componen al repositorio de la herramienta de control de versiones *VCSGis*.

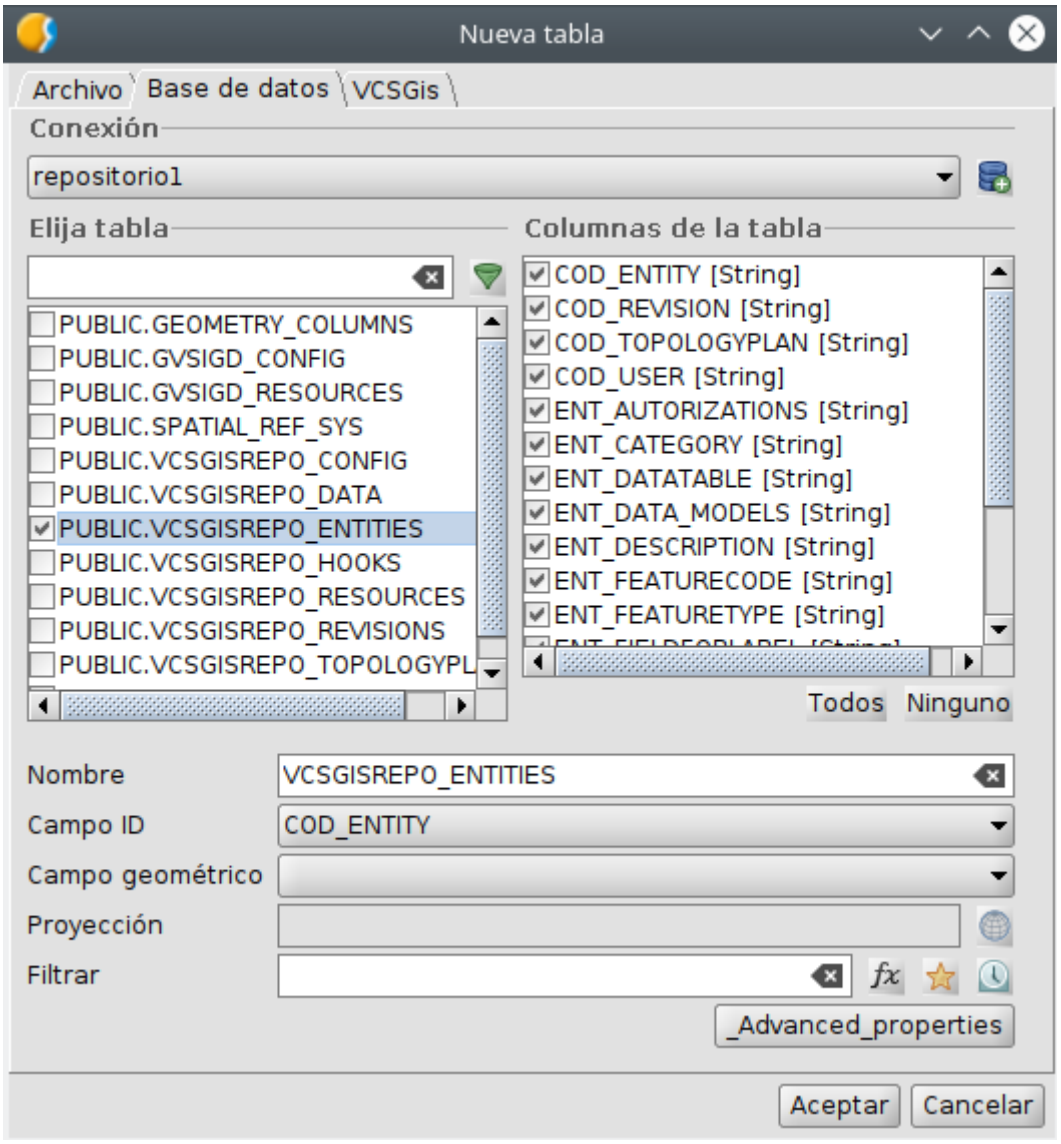
Ahora solo queda indicar a *VCSGis* que tablas de las que hay en el repositorio forman parte de un modelo de datos. Para ello se necesita disponer de acceso de escritura a la base de datos en la que reside el repositorio donde estan las tablas que forman el modelo. Cargaremos en el proyecto de *gvSIG desktop* la tabla *PUBLIC.VCSGISREPO_ENTITIES* y, bien directamente sobre la tabla o en el formulario asociado a ella, indicaremos para cada “entidad” a que modelo de datos pertenece.

Para facilitar la compresión de la configuación de modelos de datos en la herramienta de control de versiones *VCSGis* de *gvSIG Desktop* se utiliza el siguiente ejemplo. El ejemplo se basa en 4 tablas; **MD1_CONTINENTS**, **MD1_COUNTRIES**, **MD1_ECONOMY** y **MD1_REGION**, siendo el conjunto de ellas el modelo de datos **MD1**.

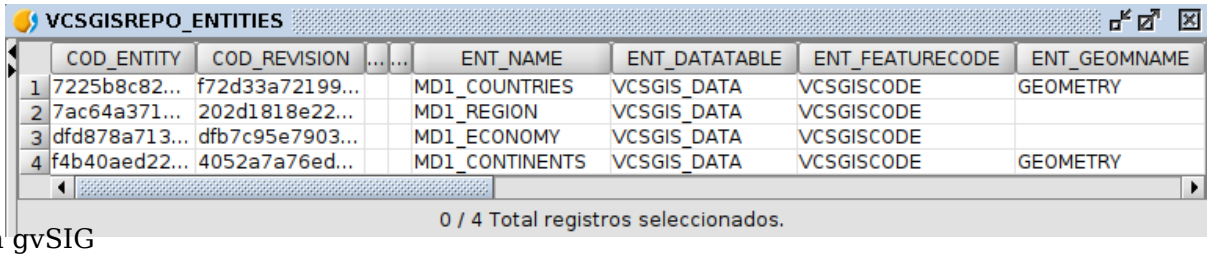




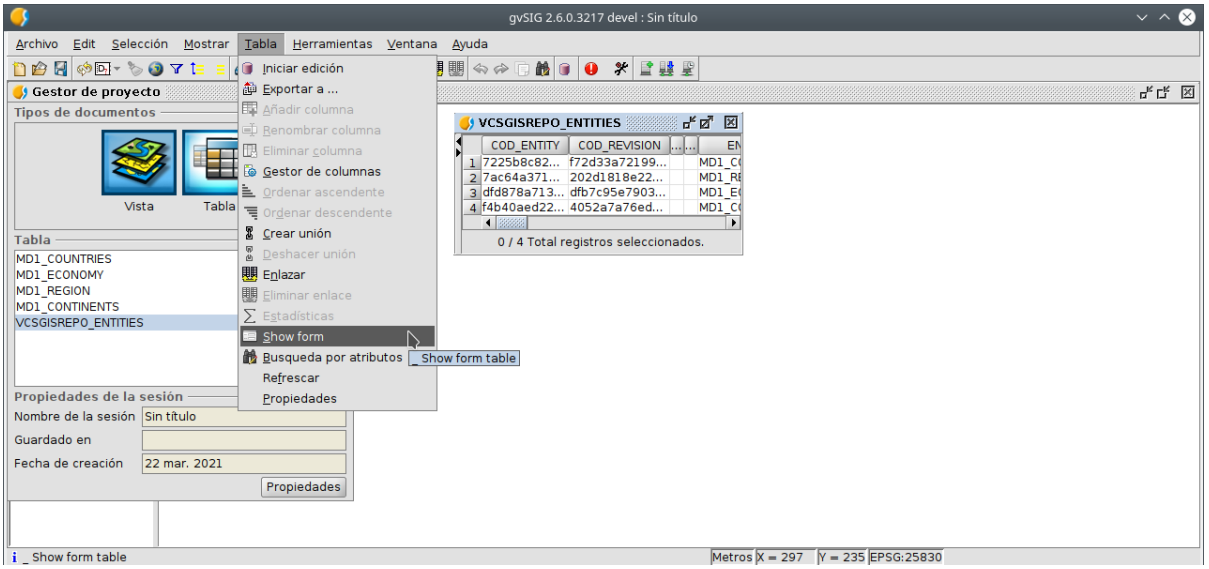
Para empezar la configuración hay que abrir la tabla del repositorio *PUBLIC.VCSGISREPO_ENTITIES* desde el *Gestor de proyectos* situado en el menú *Mostrar* de *gvSIG Desktop*. El proceso de abrir una tabla es el genérico a abrir cualquier archivo, primero se selecciona *Tabla* como tipo de datos a abrir, se selecciona la opción de *Nuevo*, lo que habilita una ventana donde se tiene que seleccionar la pestaña *Base de datos*. Esa pestaña muestra en su zona superior un desplegable donde hay que especificar la base de datos donde se encuentra la tabla. Una vez seleccionada la base de datos, en la lista de tablas de esta hay que marcar la tabla en cuestión y pulsa el botón *Aceptar*.



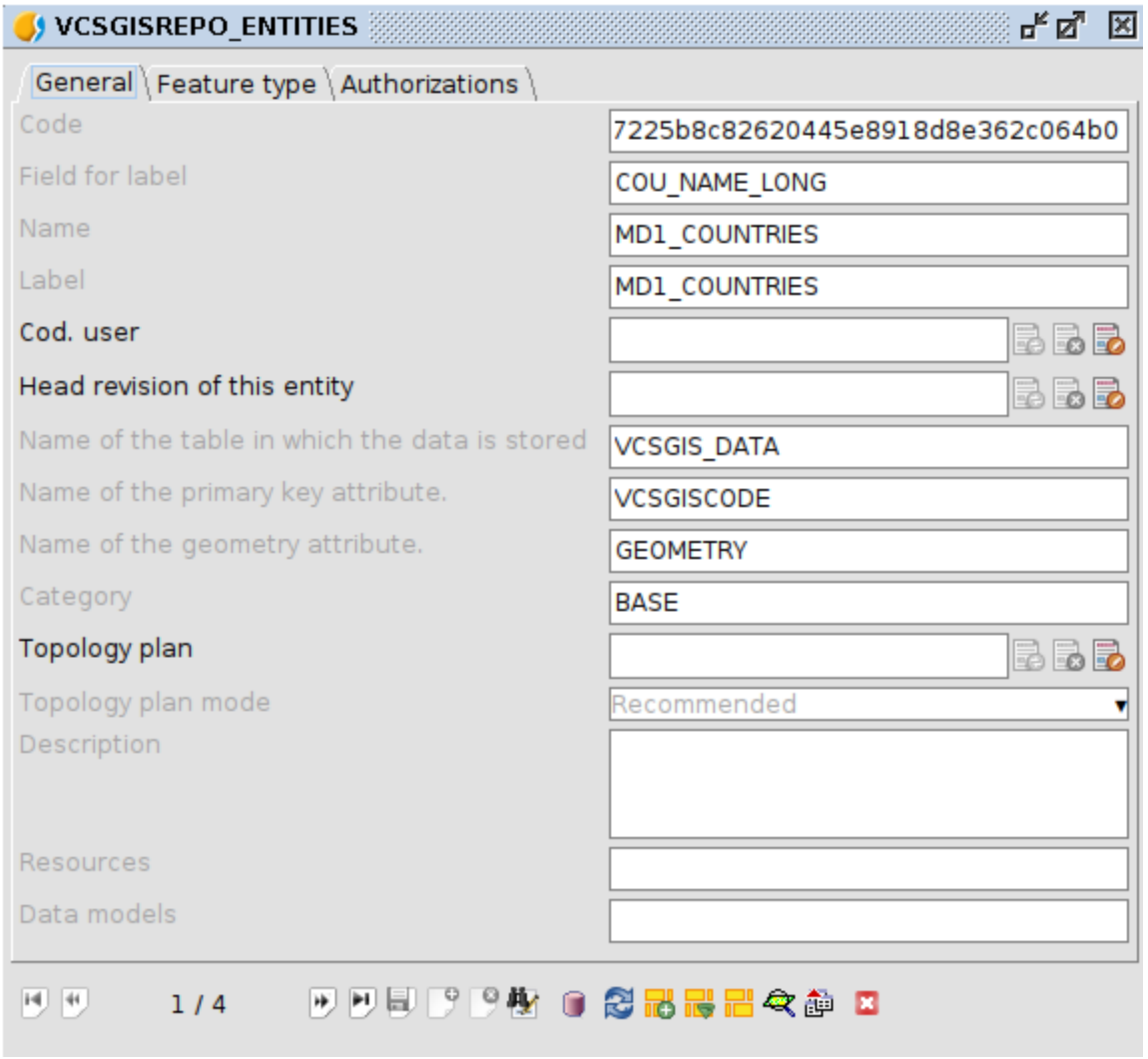
De lo anterior se obtiene la siguiente tabla.



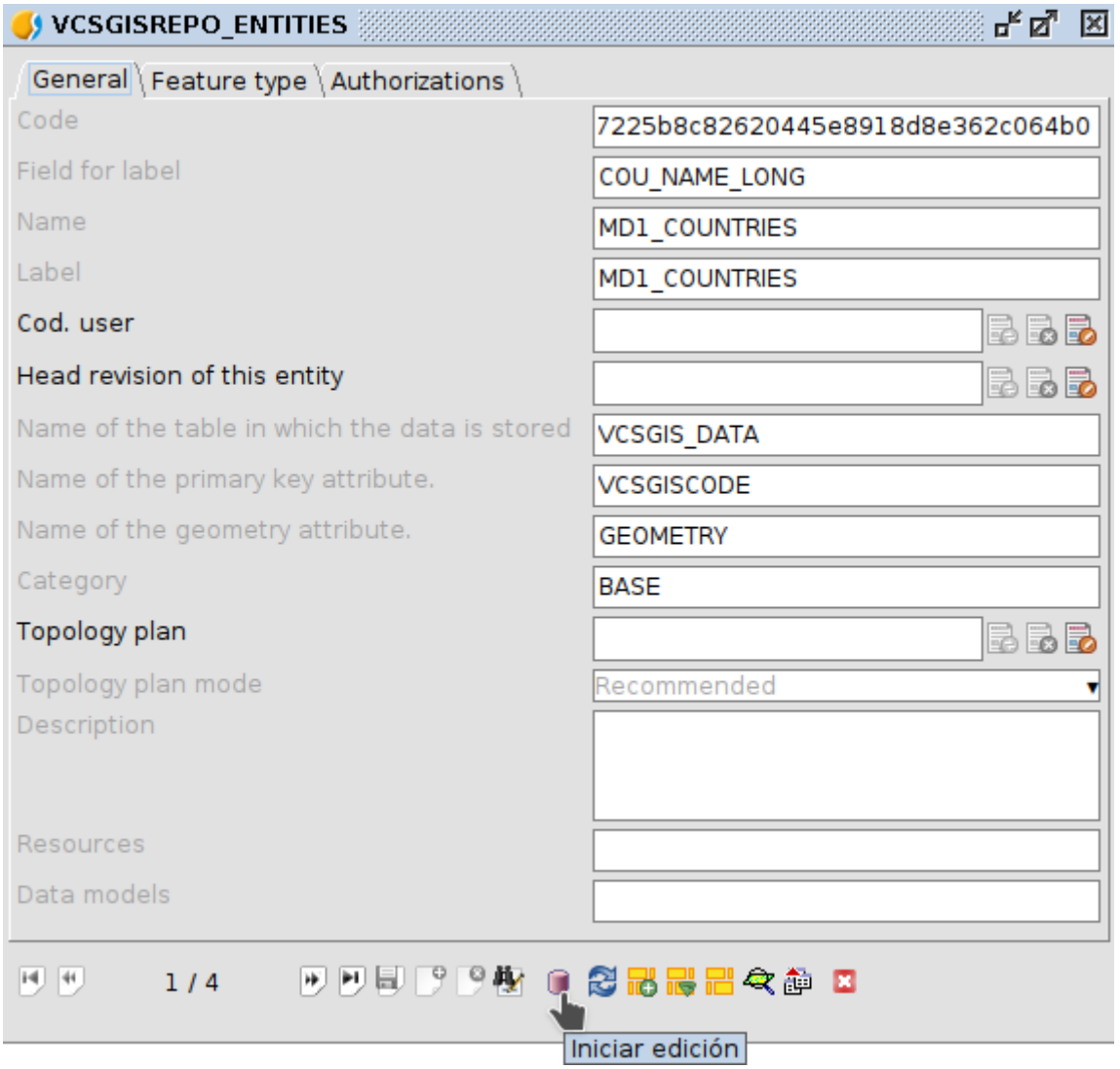
En la tabla se muestran las cuatro tablas anteriores cargadas, pero hay que especificar, que aún no conforman un modelo de datos. Para definirlo se tiene que modificar y añadir en ellas a que modelo de datos pertenecen. Para realizar lo anterior hay que editar cada tabla de manera manual o mediante su fomulario. Para obtener el formulario se ejecuta la opción *Show form* situada en el menú *Tabla* de *gvSIG Desktop* siempre y cuando la tabla este abierta y seleccionada.



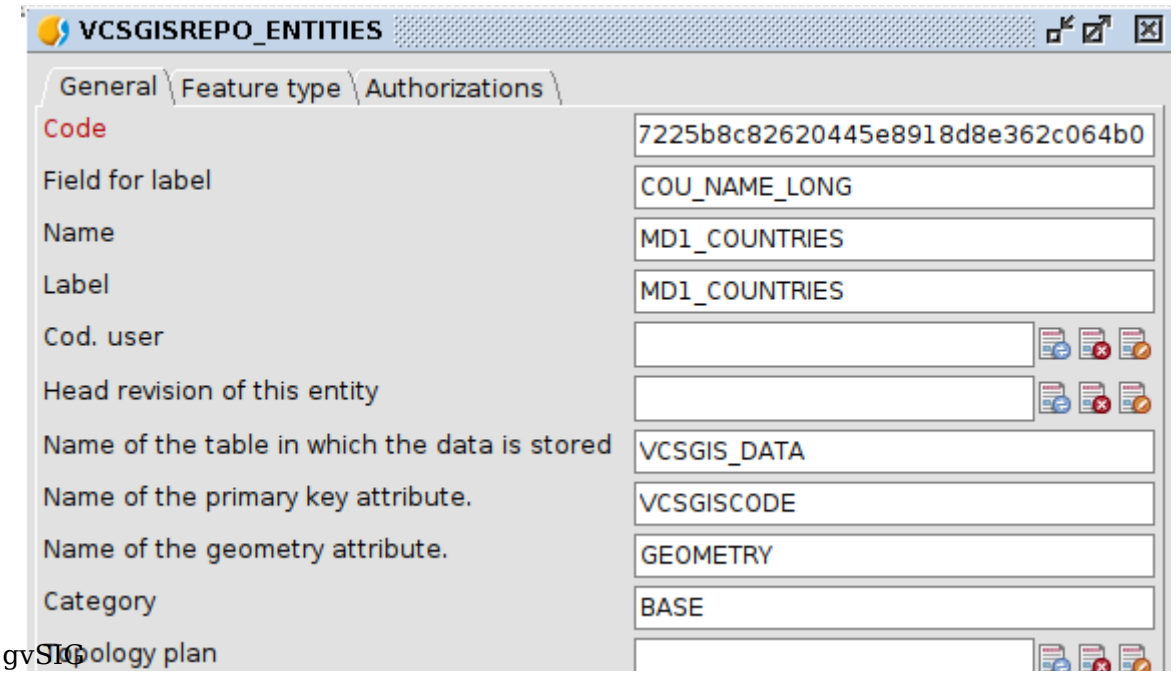
El formulario de la tabla *PUBLIC.VCSGISREPO_ENTITIES* es el siguiente.

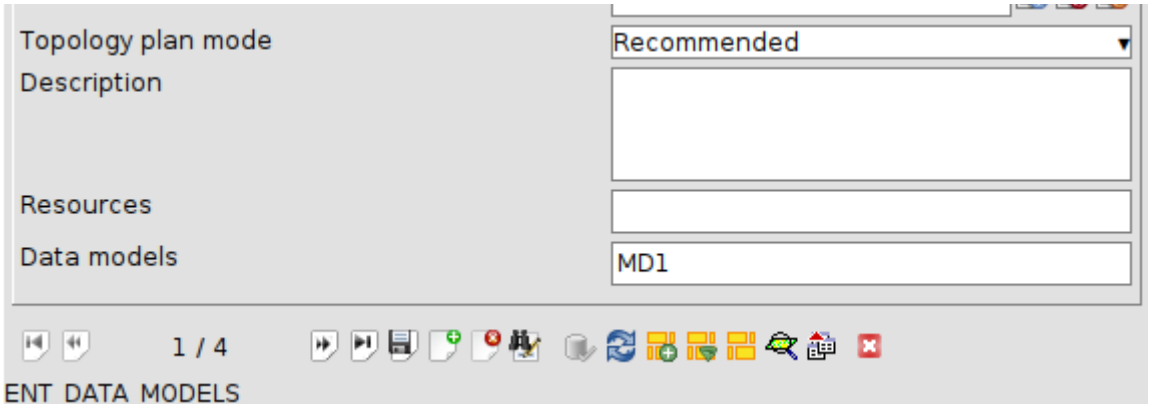


Para introducir un valor en la tabla hay que poner esta en edición ya sea desde el mismo desplegable que se mencionó anteriormente para obtener el formulario, o desde el mismo formulario utilizando el botón *Comenzar edición*.

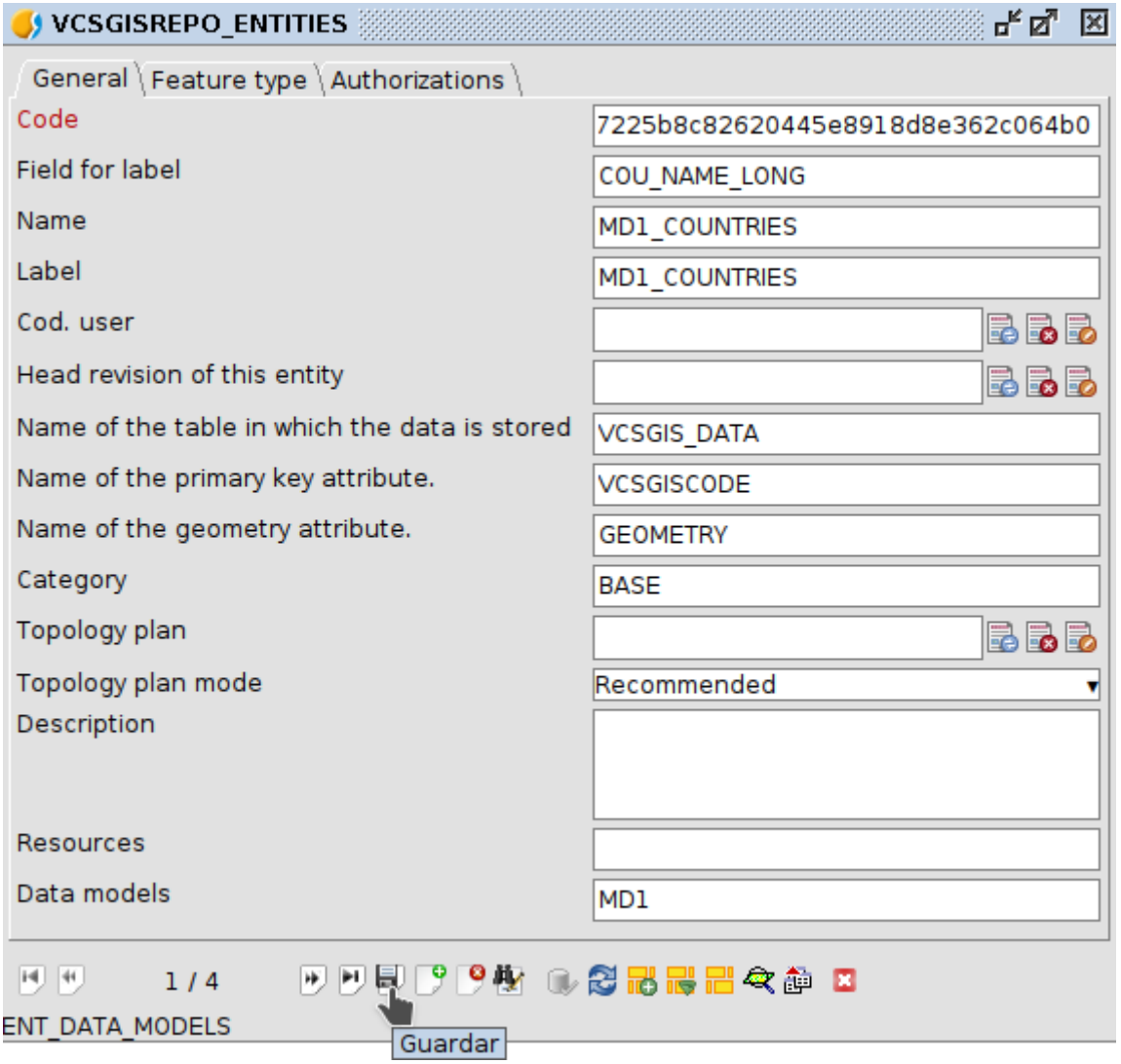


Tras esto se introduce en el campo *Data models* el modelo de datos al cual pertenece cada tabla o capa, en este caso *MD1*.





Una vez asignado el modelo de datos se guarda dicho cambio.



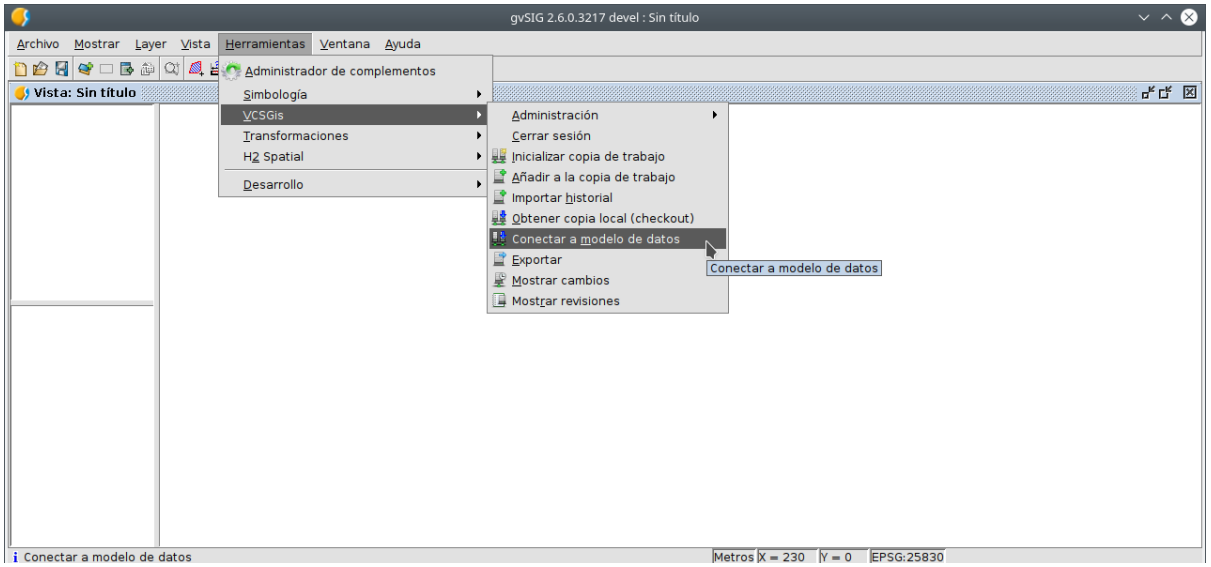
Tras lo anterior hay que repetir el proceso con todas las tablas del modelo en cuestión y finalizar la edición de la tabla *PUBLIC.VCSGISREPO_ENTITIES*

Conectarnos a un modelo de datos

Una vez definido y cargado en el repositorio un modelo de datos, se sigue el flujo de trabajo habitual con *VCSGis*. Crear una copia de trabajo y dercargar en ellas las tablas con las que se busca trabajar. Ahora bien, se dispondrá de una herramienta que permite “conectar” a un modelo de datos. Esta herramienta presenta para una copia local determinada cuales son los modelos de datos que hay definidos en ella

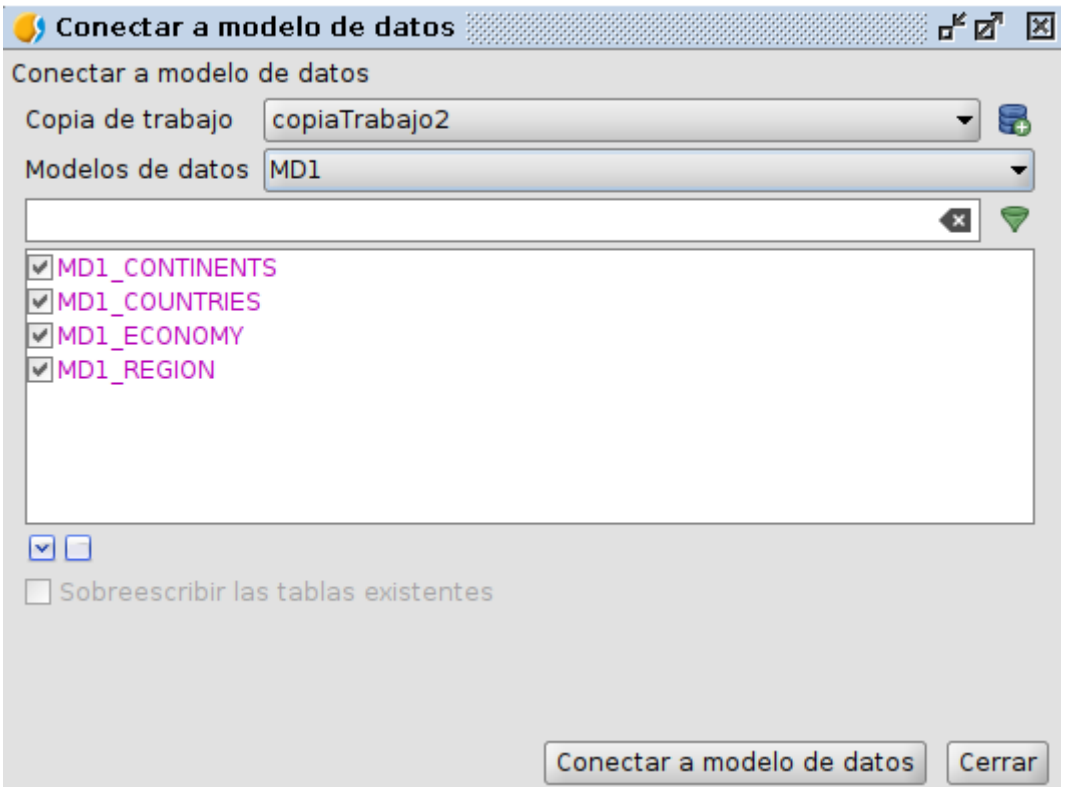
pudiendo seleccionar uno. Tras la selección se muestran las tablas que componen dicho modelo de datos, y si tras conectarse a el, se descargará las tablas del modelo y las registrará para que *gvSIG Desktop* pueda acceder a ellas aunque no esten cargadas como tablas o capas del proyecto actual.

Dicha herramienta se encuentra en el menú *Herramientas*, *submenú VCSGis*, opción *Conectar a modelo de datos*.



Esa opción despliega el siguiente cuadro de diálogo en cual tras seleccionar una copia de trabajo y el modelo de datos, nos descarga en esta las capas que conforman dicho modelo.

La siguiente ilustración muestra la ventana que tras pulsar el botón *Conectar a modelo de datos* cargará en la nueva copia de trabajo *copiaTrabajo2* las capas que conforman el modelo de datos *MD1*. La lista de capas puede verse en dicha ventana.



Consideraciones especiales sobre los modelos de datos

El sistema de “repositorio de datos” que tiene *gvSIG Desktop* para declarar donde esta cada tabla, así como la forma en que se definen las relaciones entre estas, usa un modelo “plano” de espacio de nombres. Es decir, no tiene esquemas o subespacios de nombres. Todas las tablas que se cargan en *gvSIG Desktop* comparten el mismo espacio de nombres. Esto implica que si se realiza una conexión simultanea a dos modelos de datos que contengan tablas con el mismo nombre estos colisionaran, pisandose unas tablas a otras.

Para evitar problemas de este tipo, sobre todo con modelos de datos que conviven en el mismo repositorio de *VCSGis*, es recomendable que los nombres de tablas de un modelo lleven un prefijo que identifique a ese modelo. De esta forma se pueden cargar en el control de versiones varios modelos sin que colisionen entre ellos.

Consideraciones

(**en contruccion**)

Aqui notas sobre el uso de la calculadora de campos