

CLASE
3

Bases de datos espaciales

República Argentina
Ministerio de Educación
Programa Nacional Mapa Educativo

Curso de capacitación
Bases de datos espaciales

CLASE 3
Material de lectura

Temas de esta clase:

Integridad referencial. Restricciones, Clave primaria, claves únicas y claves ajena.
Vistas.
SQL: consultas y subconsultas.
SQL: Join o juntar tablas.
Utilización de códigos comunes para entidades geográficas.
Vistas con datos espaciales.
SQL espacial: consulta con resultados geométricos.

Referencias:

A lo largo del documento encontraremos íconos y recuadros que requieren de una especial atención de los lectores:



ACTIVIDADES: son consignas de actividades para realizar la práctica con gvSIG y PGAdmin acompañando la lectura. En la presente clase hay 5 actividades para resolver.



¡IMPORTANTE! Indica una actividad que no debe omitirse para poder desarrollar correctamente la práctica de la clase.

1 Integridad referencial

La integridad referencial es una propiedad de las bases de datos relacionales garantizada por el sistema gestor de bases de datos. Esta propiedad garantiza que cada entidad almacenada en la base se relacione de manera correcta con otras entidades válidas, evitando sin repeticiones innecesarias, datos perdidos y relaciones mal resueltas.

Por ejemplo, si creamos una nueva tabla de departamentos, podemos colocar el nombre y código de cada uno de ellos en la tabla. Y si deseamos colocar un campo con el código de provincia a la que pertenece, podemos hacer que dicho campo solo albergue códigos existentes en la tabla de "provincias", evitando el ingreso de un código de provincia inexistente.

A continuación veremos algunos elementos de una base de datos en Postgres que nos garantizan la integridad referencial.

1.1 Restricciones

Las restricciones son objetos pertenecientes a las tablas de una base de datos, y que obligan a cumplir con ciertas condiciones. Se utilizan para fijar reglas dentro de la base de datos y parte de ellas son esenciales para el funcionamiento de una base de datos relacional.

Muchas restricciones son establecidas automáticamente por Postgres, pero algunas pueden ser definidas por los usuarios para asegurarse de que las tablas no admitan valores incorrectos. Ejemplo de estos valores incorrectos pueden ser un valor de latitud de signo positivo en el hemisferio sur, valores duplicados en campos que deberían tener sólo valores distintos (DNI, códigos de localidad), valores nulos para un campo preparado para almacenar un valor de superficie o un nombre, etc.

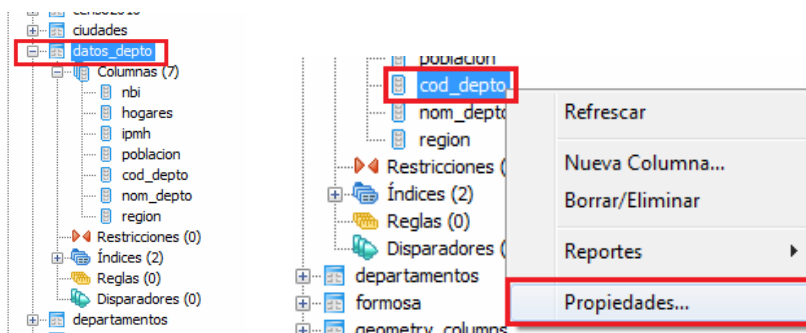
1.2 Clave primaria

Es el campo o una combinación de campos que utiliza el motor de base de datos para identificar unívocamente a cada registro de una tabla. No puede haber dos registros de una misma tabla con idéntica clave primaria.

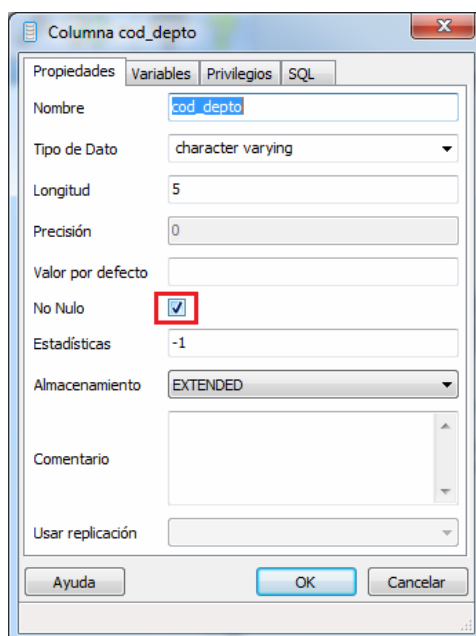
Por ejemplo, en la tabla de provincias, la clave primaria puede ser el código de la provincia. Pero en el caso de los departamentos, la clave primaria no puede ser el código de provincia, ya que este se repite tantas veces como departamentos existen en una misma provincia. Tampoco podrían utilizar el nombre como clave primaria, ya que existen nombres que se repiten en todo el país. Pero sí podrían usar los códigos de departamento, que tienen un valor único para cada registro.

	nom_prov character vari	nom_region character vari	superficie double precis	pob2001 bigint	pob2010 bigint	hogares bigint	computadora bigint	cod_prov [PK] character	densidad double precis
1	CIUDAD AUTON	CENTRO	200	2776138	2890151	1150134	789145	02	14450.755
2	BUENOS AIRES	CENTRO	307571	13827203	15625084	4789484	2308740	06	50.8015515116
3	CATAMARCA	NOA	102602	334568	367828	96001	34518	10	3.58499834311
4	CORDOBA	CENTRO	165321	3066801	3308876	1031843	510197	14	20.0148559469
5	CORRIENTES	NEA	88199	930991	992595	267797	86184	18	11.2540391614
6	CHACO	NEA	99633	984446	1055259	288422	85393	22	10.5914606606
7	CHUBUT	SUR	224686	413237	509108	157166	89422	26	2.26586436182
8	ENTRE RIOS	CENTRO	78781	1158147	1235994	375121	164184	30	15.6889859230
9	FORMOSA	NEA	72066	486559	530162	140303	36426	34	7.35661754502
10	JUJUY	NOA	53219	611888	673307	174630	59114	38	12.6516281779
11	LA PAMPA	SUR	143440	299294	318951	107674	51344	42	2.22358477412
12	LA RIOJA	CUYO	89680	289983	333642	91097	38013	46	3.72036128456
13	MENDOZA	CUYO	148827	1579651	1738929	494841	214868	50	11.6842306839
14	MISIONES	NEA	29801	965522	1101593	302953	86162	54	36.9649676185
15	NEUQUEN	SUR	94078	474155	551266	170057	90178	58	5.85966963583
16	RÍO NEGRO	SUR	203013	552822	638645	199189	97439	62	3.14583302547
17	SALTA	NOA	155488	1079051	1214441	299794	97607	66	7.81051270837
18	SAN JUAN	CUYO	89651	620023	681055	177155	65274	70	7.59673623272
19	SAN LUIS	CUYO	76748	367933	432310	126922	71376	74	5.63285036743
20	SANTA CRUZ	SUR	243943	196958	273964	81796	52176	78	1.12306563418
21	SANTA FE	CENTRO	133007	3000701	3194537	1023777	487158	82	24.0178110926
22	SANTIAGO DEL NOA	NOA	136351	804457	874006	218025	51099	86	6.40997132400
23	TUCUMAN	NOA	22524	1338523	1448188	368538	124454	90	64.2953294263

El o los campos seleccionados como claves primarias, no pueden tener valores repetidos ni nulos. Es decir, que las celdas de dichos campos no pueden estar vacías. Para asegurarnos de que un campo no pueda almacenar valores nulos, podemos entrar a sus propiedades y configurar la propiedad “No nulo”. Para ellos accedemos al campo desplegando las columnas con el botón “+” situado a la izquierda del ícono de la tabla, y luego hacemos clic con el botón derecho sobre la columna correspondiente y seleccionamos la opción “Propiedades”.



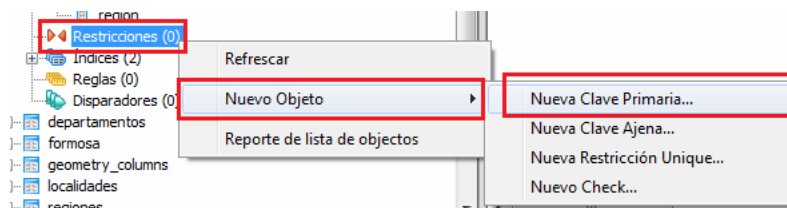
Una vez en la ventana de propiedades de la columna o campo, activamos el check box de la propiedad “No nulo” y luego apretamos “OK”. Si la opción ya está marcada, simplemente cerramos la ventana.



Para crear una clave primaria, en primer lugar es necesario identificar el campo o el conjunto de campos que tienen (y siempre tendrán) valores diferentes para cada registro. En caso de las entidades geográficas es muy recomendable utilizar los códigos normalizados que se hayan establecido. (Código de departamento, localidad, provincia, región, barrio, catastro, etc.)

	nbi real	hogares integer	ipmh real	poblacion integer	cod_depto character var	nom_depto text	region text
1	36	2193	78.8	8970	18098	MBURUCUYA	NEA
2	17.5	62937	48.8	243485	06260	ESTEBAN ECHE	CENTRO
3	35.5	4883	75.5	21435	18126	SALADAS	NEA
4	52.7	4070	83.5	18390	18028	CONCEPCION	NEA
5	36.3	4079	77.9	17911	18161	SAN ROQUE	NEA
6	26.4	9586	65	38931	18105	MERCEDES	NEA
7	13.4	10961	46.6	38683	30077	NOGOYA	CENTRO
8	14.6	7566	46.4	25627	30091	TALA	CENTRO
9	16	9676	46.7	33667	30105	VICTORIA	CENTRO
10	11.6	12539	38.5	43069	30021	DIAMANTE	CENTRO
11	9.4	88751	32.8	317431	30084	PARANA	CENTRO
12	22.5	29574	57.3	116006	06270	EZEIZA	CENTRO
13	17.7	80312	50.4	326765	18021	CAPITAL	NEA
14	25.7	11147	78.6	47209	54014	CAINGUAS	NEA
15	24.5	3210	66	13099	18133	SAN COSME	NEA
16	23.6	10736	65.5	41331	54070	LEANDRO N. AL	NEA
17	16.3	133787	45.8	512517	06028	ALMIRANTE BR	CENTRO
18	25	2043	72.6	8717	18077	ITATI	NEA
19	14.9	1913	36.6	6257	62028	CONESA	SUR
20	11.2	4121	30.4	13935	62063	PICHI MAHUIDA	SUR
21	16.5	8746	37.1	31816	62014	AVELLANEDA	SUR
22	15.6	77975	36.8	279817	62042	GENERAL ROCA	SUR

Una vez identificados los campos, desplegamos los objetos dependientes de la tabla, y vemos que aparece el ícono de restricciones. Hacemos clic con el botón derecho sobre el objeto “Restricciones” y seleccionamos la opción “Nuevo Objeto”, y luego “Nueva clave primaria”

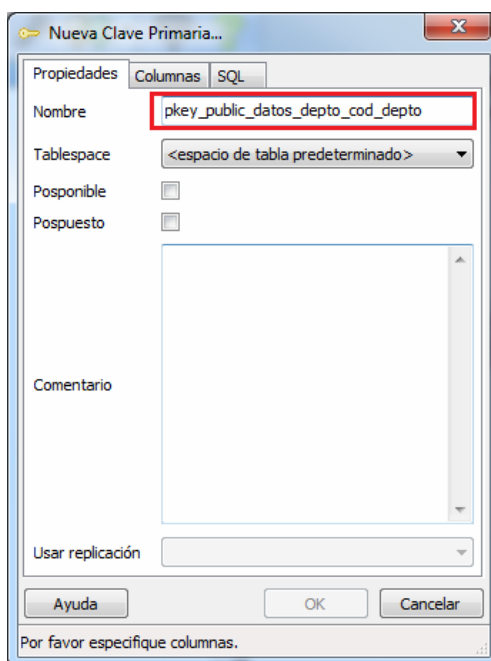


Se abrirá una ventana en donde daremos atributos a la nueva clave primaria. En este caso solo colocaremos un nombre. Como propuesta se puede utilizar la siguiente fórmula para nombrar los objetos “clave primaria” de la base:

pkey + esquema + tabla + campo/s

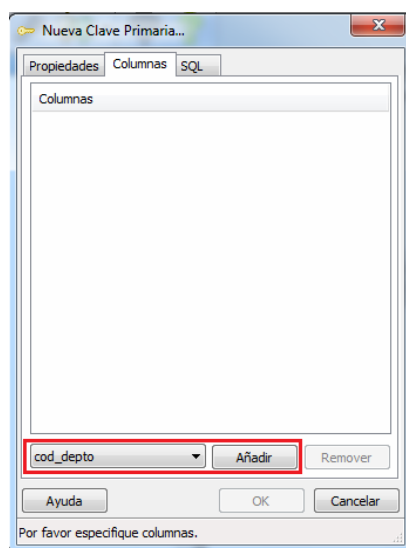
En este caso, siguiendo la fórmula, el nombre de la clave primaria sería:

“pkey_public_datos_depto_cod_depto”

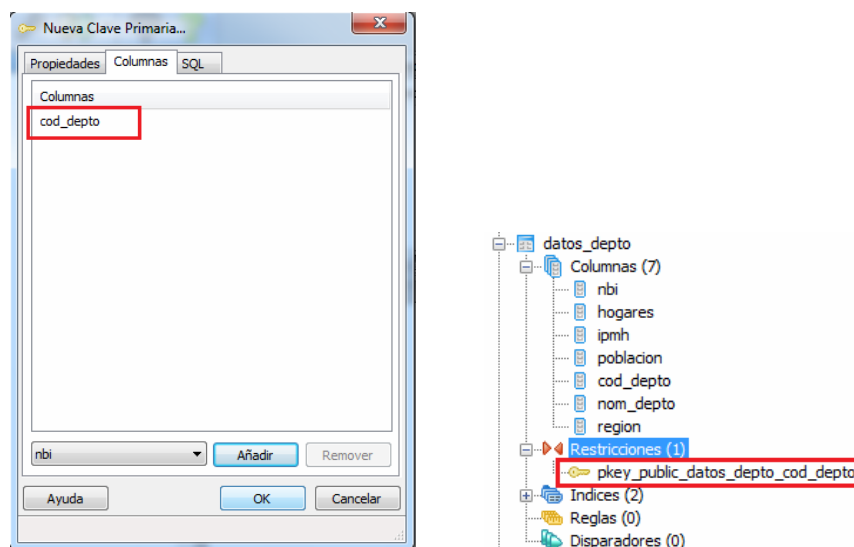


Es importante que en el entorno de cada institución se utilicen nombres consensuados para los objetos de la base de datos. De esta manera se facilita el acceso a los datos por parte de todos los usuarios, independientemente de quién haya creado cada objeto.

Luego de dar el nombre a la clave primaria, pasamos a la solapa "Columnas" en donde se selecciona la o las columnas que formarán la clave primaria. En este caso seleccionamos la columna "cod_depto" y apretamos el botón "Añadir".



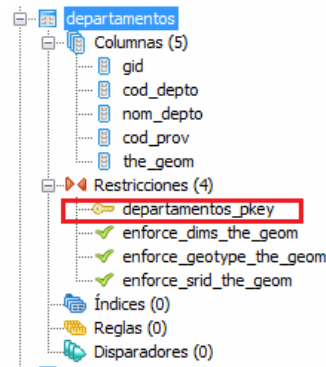
Si existiera otro campo a concatenar para formar la clave, simplemente lo seleccionamos y volvemos a apretar "Añadir". Finalmente, apretamos "OK" para aceptar los cambios y cerrar la ventana.



Hay operaciones que solo se pueden realizar con tablas que tienen clave primaria, y por lo tanto el sistema puede identificar a cada registro. Un ejemplo de estas operaciones es la edición de tablas desde la interface de PGAdmin. Si abrimos la tabla `datos_depto` con el botón *Ver los datos del objeto seleccionado*, comprobaremos que no podemos modificar el contenido de las celdas. Pero si asignamos la función de clave primaria a uno de sus campos: “`cod_depto`”, veremos que cuando la volvemos a abrir, el sistema nos permite realizar modificaciones y guardarlas.

	nbi real	hogares integer	ipmh real	poblacion integer	cod_depto [PK] caracte	nom_depto text	region text
1	6	97971	13.9	226999	02001	I	CENTRO
2	5.6	98934	11.3	229780	02002	II	CENTRO
3	18.7	51163	14.1	119847	02003	III	CENTRO
4	18.6	30513	19.4	83474	02004	IV	CENTRO
5	18.9	30284	26.8	93120	02005	V	CENTRO
6	11.5	57095	12.2	148006	02006	VI	CENTRO
7	3.9	61019	9.7	156316	02007	VII	CENTRO
8	3.9	51772	9.3	133996	02008	VIII	CENTRO
9	4	95082	9.9	232807	02009	MODIFICAR	CENTRO
10	2.5	83579	8.7	212870	02010		CENTRO
11	5.6	35773	11.8	101297	02011	XI	CENTRO
12	6.7	38714	11.8	106825	02012	XII	CENTRO
13	6.1	36476	18	112217	02013	XIII	CENTRO

En el caso de la migración de capas en formato shapefile desde gvSIG o PGAdmin; la definición de la clave primaria se realiza en forma automática. Por ejemplo, la tabla de departamentos creada en la clase anterior, ya tiene definida su clave primaria. Podemos comprobarlo al abrir los objetos de dicha tabla.



ACTIVIDAD 1

Crear un documento de texto para luego subir al campus.

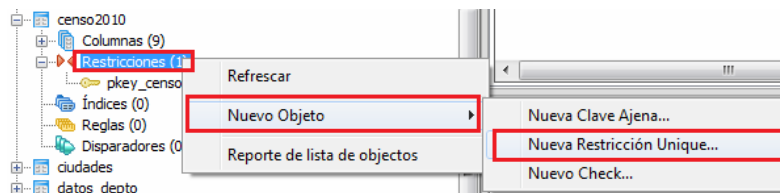
Agregar una clave primaria a la tabla "datos_depto" sobre el campo "cod_depto"

Visualizar la tabla luego de crear la clave primaria, imprimir pantalla y agregar al documento de texto.

1.3 Clave única

La clave única es similar a la clave primaria, ya que también restringe la posibilidad de que existan datos repetidos en una tabla. Pero la diferencia es que el sistema no la utiliza para identificar unívocamente a cada registro, y además la clave única admite la existencia de un registro vacío. Esto último significa que no es necesario que el o los campos que componen la clave única tengan habilitados la propiedad "No nulo" en sus propiedades.

Para crear la restricción de clave única, nos dirigimos al objeto "restricciones" de la tabla, hacemos clic con el botón derecho y seleccionamos la opción "Nueva restricción unique".

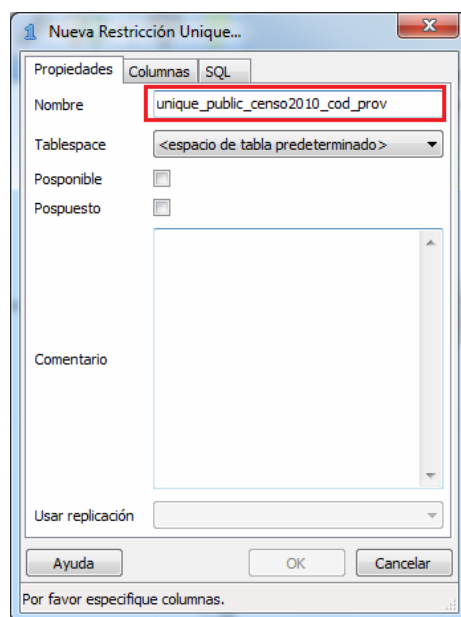


Aquí se nos abre la ventana de propiedades, donde podremos colocar un nombre. En este caso, la fórmula propuesta para componer el nombre es:

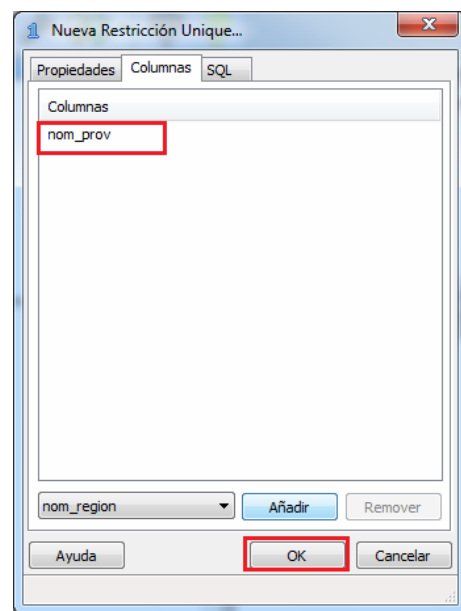
unique + esquema + tabla + campo/s

Que en este caso podría ser:

unique_public_censo2010_cod_prov

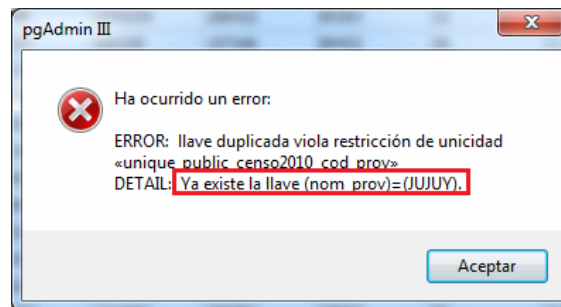


Cuando definimos el nombre, podemos movernos a la solapa “Columnas” y seleccionar el o los campos que formarán la clave única. En este caso seleccionamos “nom_prov”. Y luego presionamos “OK”.



Luego de haber definido la clave única, podremos comprobar su funcionamiento. Por ejemplo, podemos intentar cambiar el nombre de una provincia para que se repita con el valor existente en otro registro. Si intentamos renombrar la provincia de Salta con el valor “JUJUY”, de inmediato nos saltará un aviso que nos informa que estamos violando la restricción de la clave única, ya que no pueden existir dos registros con el mismo valor.

	nom_prov character vai	nom_region character vai	superficie double precis	pob2001 bigint	pob2010 bigint	hogares bigint	computadora bigint	cod_prov [PK] caracte	densidad double precis
1	CIUDAD AUTON	CENTRO	200	2776138	2890151	1150134	789145	02	14450.755
2	BUENOS AIRES	CENTRO	307571	13827203	15625084	4789484	2308740	06	50.8015515116
3	CATAMARCA	NOA	102602	334568	367828	96001	34518	10	3.58499834311
4	CORDOBA	CENTRO	165321	3066801	3308876	1031843	510197	14	20.0148559469
5	CORRIENTES	NEA	88199	930991	992595	267797	86184	18	11.2540391614
6	CHACO	NEA	99633	984446	1055259	288422	85393	22	10.5914606606
7	CHUBUT	SUR	224686	413237	509108	157166	89422	26	2.26586436182
8	ENTRE RIOS	CENTRO	78781	1158147	1235994	375121	164184	30	15.6889859230
9	FORMOSA	NEA	72066	486559	530162	140303	36426	34	7.35661754502
10	JUJUY	NOA	53219	611888	673307	174630	59114	38	12.6516281779
11	LA PAMPA	SUR	143440	299294	318951	107674	51344	42	2.22358477412
12	LA RIOJA	CUYO	89680	289983	333642	91097	38013	46	3.72036128456
13	MENDOZA	CUYO	148827	1579651	1738929	494841	214868	50	11.6842306839
14	MISIONES	NEA	29801	965522	1101593	302953	86162	54	36.9649676185
15	NEUQUEN	SUR	94078	474155	551266	170057	90178	58	5.85966963583
16	RÍO NEGRO	SUR	203013	552822	638645	199189	97439	62	3.14583302547
17	JUJUY	NOA	155488	1079051	1214441	299794	97607	66	7.81051270837
18		CUYO	89651	620023	681055	177155	65274	70	7.59673623272



1.4 Claves ajena

Las claves ajenas o foráneas son restricciones que prohíben la incorporación de valores no permitidos en un campo determinado. El conjunto de valores permitidos se define en función de los datos existentes en otra tabla, y es por esto que se denomina clave *foránea*. El campo de la tabla que tiene los valores permitidos debe ser del mismo tipo de dato que la columna a la que se le aplica la restricción. Además, este campo de referencia debe ser “no nulo” y poseer la restricción de clave única.

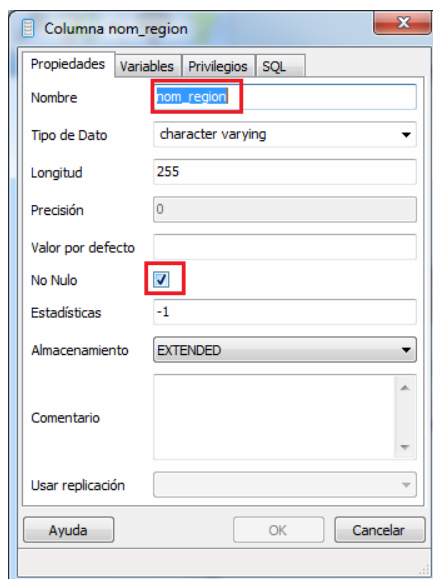
Por ejemplo, en la tabla “datos_deptos” tenemos el campo “region”. Para asegurarnos de que nadie agregue en este campo un valor incorrecto, podemos crearle una clave foránea para que solo admita los valores almacenados en el campo “nom_region” de la tabla “region”.

	nbi real	hogares integer	ipmh real	poblacion integer	cod_depto [PK] caracte	nom_depto text	region text
456	23.8	3090	50.2	10952	74042	GOBERNADOR	CUYO
457	11.7	5742	38.1	19656	74049	JUNIN	CUYO
458	10.5	46284	34.9	167682	74056	LA CAPITAL	CUYO
459	47.5	1484	81.2	5146	74063	LIBERTADOR	CUYO
460	5.4	2248	14.4	7634	78007	CORPEN AIG	SUR
461	12	19890	25	72174	78014	DESEADO	SUR
462	9.2	25096	18.2	91419	78021	GUER AIG	SUR
463	11.2	1982	19.4	6580	78028	LAGO ARGENT	SUR
464	10.6	1792	29.1	6059	78035	LAGO BUENOS	SUR
465	6.9	1943	20.3	6139	78042	MAGALLANES	SUR
466	12.6	883	23.9	2846	78049	RIO CHICO	SUR
467	9	12486	30.4	41258	82007	BELGRANO	CENTRO
468	8.4	24771	25.5	78441	82014	CASEROS	CENTRO
469	9.1	47672	29.8	161200	82021	CASTELLANOS	CENTRO
470	11.3	24499	36.2	82710	82028	CONSTITUCION	CENTRO
471	25.9	4946	64.9	19869	82035	GARAY	CENTRO
472	9.9	56701	32.8	180796	82042	GENERAL LOPEZ	CENTRO

	nom_region character vari	cod_region smallint
1	CENTRO	1
2	CUYO	2
3	NEA	3
4	NOA	4
5	SUR	5

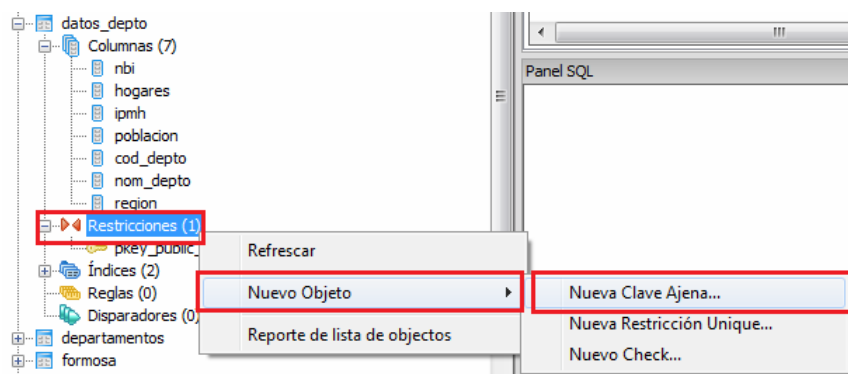
CLAVE AJENA

Para lograr esto, primero debemos asegurarnos de que el campo “nom_region” de la tabla “regiones” no contenga valores repetidos. Luego, debemos habilitar la propiedad “no nulo” de dicho campo, haciendo clic con el botón derecho sobre la columna, y luego marcando con el tilde en el casillero de “no nulo”.



Una vez preparado el campo de referencia, podemos crear la restricción de clave ajena sobre el campo “region” de la tabla “datos_depto”. Desplegamos los objetos dependientes de la tabla “datos_depto” y hacemos clic con el botón derecho sobre las restricciones.

Del menú desplegable, seleccionamos las opciones “Nuevo objeto” y “Nueva clave ajena”.



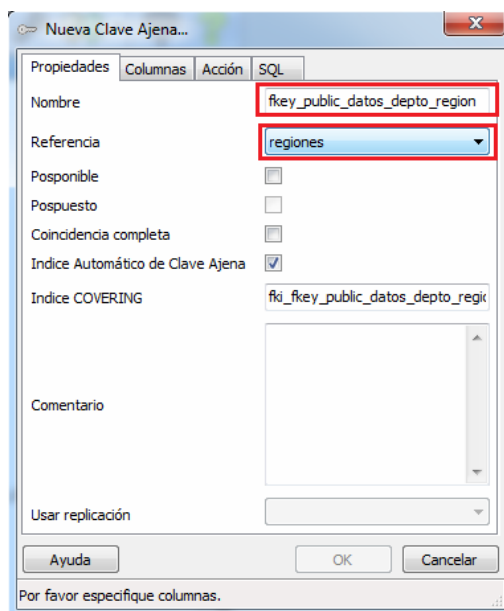
En la ventana, colocamos el nombre usando la fórmula:

fkey + esquema + tabla + campo

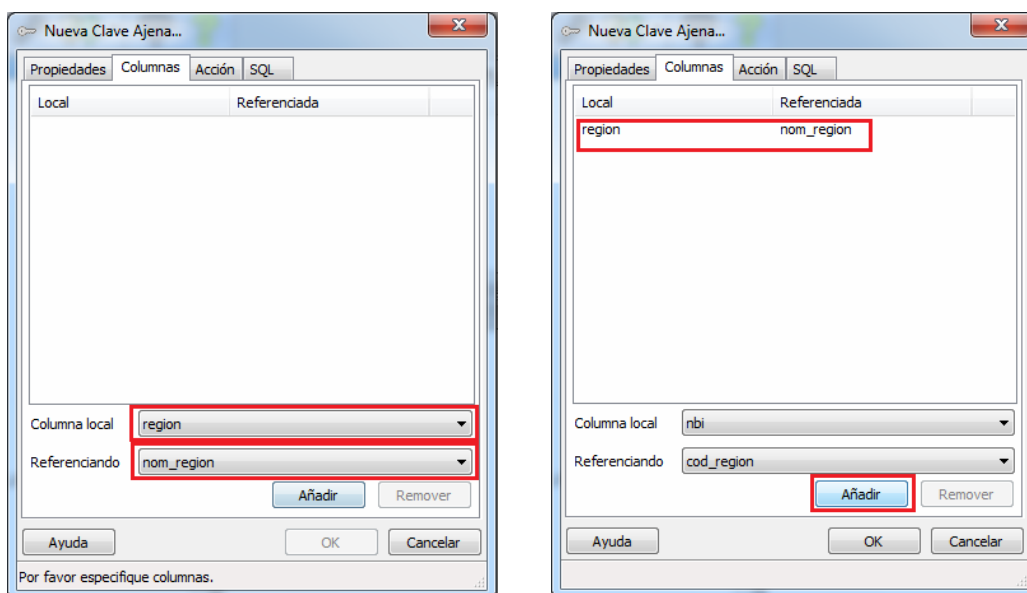
De la que resulta el nombre:

fkey_public_datos_depto_region

Y en “Referencia” seleccionamos el nombre de la tabla que contiene el campo de referencia. En este caso el nombre de la tabla es “regiones”.



Luego, vamos a la solapa “Columnas” y seleccionamos las columnas a relacionar. En “Columna local” seleccionamos la columna de la tabla local que deseamos restringir: “region”.

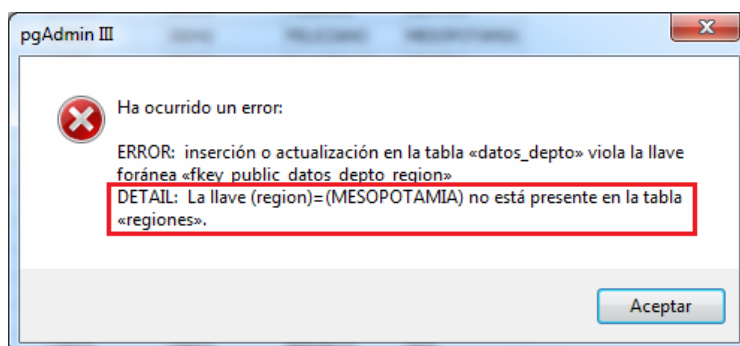


Y en la opción “Referenciando” seleccionamos la columna de referencia de la tabla ajena: “nom_region”. Apretamos el botón “Añadir” y vemos que se agrega la relación de referencia entre la columna local y la ajena.

Para terminar colocamos “OK”. Ahora podemos probar como funciona la restricción. Intentemos agregar en un valor no permitido en la “region” de la tabla “datos_depto”. Por ejemplo, podemos reemplazar el valor “CENTRO” por el valor “MESOPOTAMIA” en una celda.

	nbi real	hogares integer	ipmh real	poblacion integer	cod_depto [PK] caracte	nom_depto text	region text
264	19.4	12934	54.8	48159	30013		CENTRO
265	21.4	40334	49.1	156054	30015	CONCORDIA	CENTRO
266	11.6	12539	38.5	43069	30021	DIAMANTE	CENTRO
267	18.5	15449	46	59946	30028	FEDERACION	CENTRO
268	25.2	6218	62.6	24779	30035	FEDERAL	CENTRO
269	28.8	3383	70.6	14519	30042	FELICIANO	MESOPOTAMIA
270	16.1	13627	42.4	47791	30049	GUALEGUAY	
271	11.3	28671	39	100641	30056	GUALEGUAYCHIL	CENTRO
272	36.3	2970	62.5	11427	30063	ISLAS DEL IBICU	CENTRO
273	24	15986	60.9	65889	30070	LA PAZ	CENTRO
274	13.4	10961	46.6	38683	30077	NOGOYA	CENTRO
275	9.4	88751	32.8	317431	30084	PARANA	CENTRO
276	17.6	4470	51.9	16089	30088	SAN SALVADOR	CENTRO
277	14.6	7566	46.4	25627	30091	TALA	CENTRO

Luego de escribir, al salir de la celda o al presionar Enter, comprobaremos que nos salta la restricción correspondiente.



Si en algún momento fuera necesario agregar una nueva región en esta tabla, deberíamos agregarlo, en primer lugar, en la columna de referencia ajena (tabla regiones), y luego recién estaremos habilitados a usar dicho valor en la campo con la restricción.

2 Vistas

Las vistas son objetos de la base de datos definidas por una consulta de selección. Se comportan como “tablas virtuales”, ya que tienen campos, registros, y pueden ser consultadas, pero en realidad no almacenan datos, sino que los “traen” desde las tablas de origen en el momento en que son solicitados.

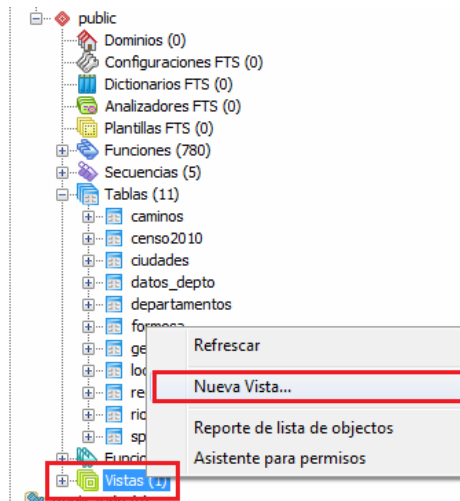
Por ejemplo, podemos tener una tabla con los datos de todos los departamentos del país, y a partir de ella crear vistas con los datos de cada región. Para crear una vista con los datos de la región “CUYO”, en primer lugar definimos la consulta que trae dichos registros:

```
SELECT * FROM datos_depto WHERE region = 'CUYO'
```

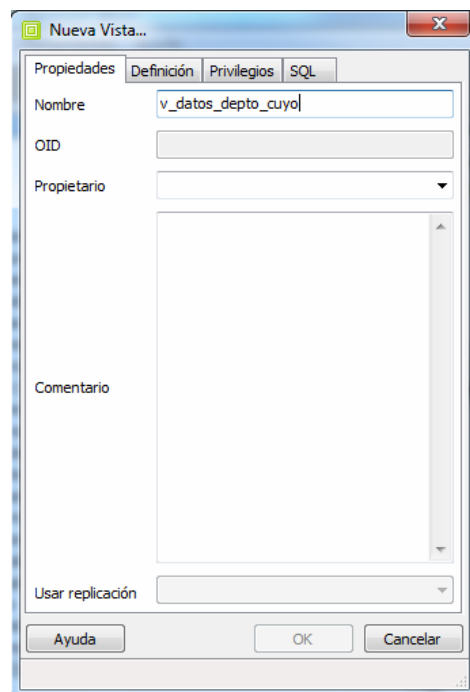
Esta consulta nos devuelve los registros de Cuyo.

SELECT * FROM datos_depto WHERE region = 'CUYO'							
Panel de Salida							
Salida de datos							
	nbi	hogares	ipmh	poblacion	cod_depto	nom_depto	region
	real	integer	real	integer	character varying(5)	text	text
1	23.3	1817	77.2	8073	70021	CALINGASTA	CUYO
2	26.3	4114	74.3	19068	70105	SARMIENTO	CUYO
3	27.8	3250	70.9	15193	70126	25 DE MAYO	CUYO
4	29.8	1542	64.5	6768	70119	VALLE FERTIL	CUYO
5	20.5	1715	73	7563	70014	ANGACO	CUYO
6	22.1	4495	73.6	20269	70007	ALBARDON	CUYO
7	47.5	1484	81.2	5146	74063	LIBERTADOR	CUYO
8	11.4	10650	48.7	43524	70098	SANTA LUCIA	CUYO
9	7.2	31970	31	111723	70028	CAPITAL	CUYO
10	19.1	3683	51.6	12469	74021	CORONEL PRI	CUYO

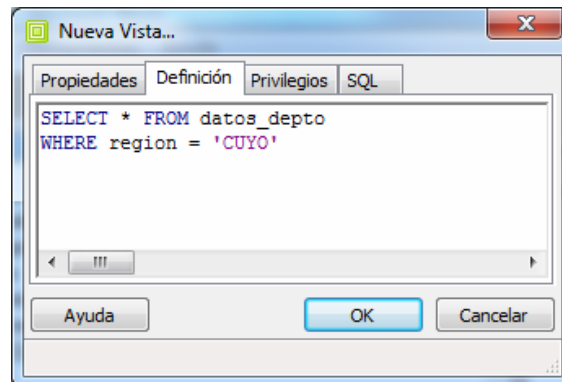
Luego, creamos un nuevo objeto Vista y le damos como definición esta consulta. Para crear una nueva vista, buscamos el objeto “Vistas” que cuelga del esquema “Public”. Hacemos un clic con el botón derecho sobre “Vistas” y seleccionamos la opción “Nueva vista”.



En la primera solapa, solo colocaremos el nombre de la nueva vista. En este caso, proponemos que el nuevo nombre utilice como formato la letra “v” antes del nombre, para diferenciarlo de las tablas. Por ejemplo, la nueva vista se podría llamar “v_datos_depto_cuyo”.





Una vez colocado el nombre, vamos a la solapa “Definición”. En el gran espacio en blanco escribimos o pegamos la consulta que probamos previamente en el constructor de consultas.



Es muy recomendable comprobar el correcto funcionamiento de la consulta antes de colocarlo en la definición. Dicha comprobación se puede hacer en el

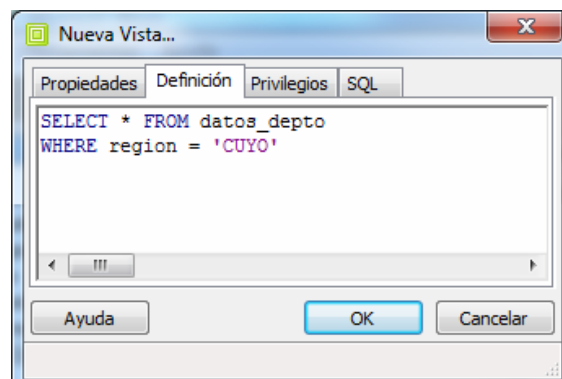


“Constructor de consultas arbitrarias”. Una vez colocada la definición de la vista, presionamos “OK”, y vemos como se agregar un nuevo elemento al árbol de objetos de PGAdmin.

Ahora podemos usar la vista de la misma manera que lo haríamos con una tabla. Podemos visualizarla con el botón , o podemos consultar sus datos desde el Constructor de consultas  colocando:

```
SELECT * FROM v_datos_depto_cuyo;
```

Podríamos generar todas las vistas que quisiéramos, sabiendo que no estamos generando nuevos datos, sino que los estamos poniendo a disposición de los usuarios de maneras diferentes. Además, como los datos provienen de una sola tabla, al actualizar los datos de dicha tabla, esta actualización repercute en todas las vistas relacionadas.



También se puede crear vistas utilizando sintaxis SQL en lugar de crearlas desde la interfaz visual de PGAdmin. Para el ejemplo que estuvimos viendo, la consulta sería:

```
CREATE VIEW v_datos_depto_cuyo AS  
SELECT * FROM datos_depto WHERE region = 'CUYO';
```

Es decir, que se coloca los comandos “CREATE VIEW”, luego el nombre de la vista a crear, y luego el comando “AS” seguido de la definición de la consulta.



ACTIVIDAD 2

Crear una vista “v_computadoras” con la consulta que devuelve la cantidad de computadoras por hogar en las provincias.

```
SELECT nom_prov, computadoras / hogares :: float AS compu_hogar
FROM censo2010;
```

Visualizar la vista, imprimir pantalla y agregar al documento.

3 SQL: consultas y subconsultas.

El lenguaje de consulta SQL nos permite introducir parámetros en la consulta que, a su vez, son el resultado de otra consulta.

Por ejemplo, en la consulta:

```
SELECT * FROM datos_depto where region = 'CENTRO'
```

El parámetro 'CENTRO' puede ser reemplazado por la consulta:

```
SELECT nom_region FROM regiones WHERE cod_region = 1
```

Y de esta manera, la consulta quedaría:

```
SELECT * FROM datos_depto where region =
(SELECT nom_region FROM regiones WHERE cod_region = 1)
```

El beneficio, en este caso, es que no necesitamos conocer exactamente como están escritas las regiones. Simplemente vamos cambiando de código y generando las consultas para cada región. Por ejemplo, para la región “CUYO”, solo reemplazamos el número de región en la consulta.

```
SELECT * FROM datos_depto where region =
(SELECT nom_region FROM regiones WHERE cod_region = 2)
```

También es útil usar el resultado de una consulta para generar un conjunto de valores que formarán parte de la cláusula “WHERE” de la consulta. Por ejemplo, si deseo obtener las rutas de la región Cuyana, como no disponemos del dato de región en la tabla “rios”, entonces necesitaremos enumerar cada una de las provincias que la componen en la condición “WHERE”:

```
SELECT nombre FROM rios WHERE provincia = 'SAN JUAN'
OR provincia = 'SAN LUIS'
OR provincia = 'MENDOZA'
OR provincia = 'LA RIOJA';
```

Colocamos “IN” y luego la enumeración de valores para el mismo campo en vez de repetir la condición con respecto al mismo campo:


```
SELECT nombre FROM rios WHERE provincia IN ('SAN JUAN',
'MENDOZA', 'SAN LUIS', 'LA RIOJA');
```

El listado de provincias de la región también lo podemos obtener de la consulta:

```
SELECT nom_prov FROM censo2010 where nom_region = 'CUYO';
```

```
SELECT nom_prov FROM censo2010 where nom_region = 'CUYO';
```

Salida de datos	
Comentar Mensajes Historial	
nom_prov character varying(255)	
1 LA RIOJA	
2 MENDOZA	
3 SAN JUAN	
4 SAN LUIS	

Y de esta manera, reemplazamos la enumeración de provincias, por el resultado de la subconsulta:

```
SELECT nombre FROM rios WHERE provincia IN
( SELECT nom_prov FROM censo2010 where nom_region = 'CUYO');
```

Salida de datos	
Comentar Mensajes Historial	
nombre text	
22 SAN JUAN	
23 SAN JUAN	
24 SAN JUAN	
25 DE LOS PATOS	
26 SALADO	
27 DE LA CIENAGA	
28 SAN JUAN	
29 SAN GUILLERMO	
30 LECHUZO	
31 SALADO	
32 BLANCO	
33 ATUEL	
34 ATUEL	
35 ATUTIA	
36 OLTA	

4 SQL: Join o juntar tablas.

El procedimiento de juntar tablas en PostgreSQL es similar al utilizado desde cualquier SIG de escritorio en donde juntamos los datos de una tabla independiente con los de la tabla de atributos de una capa. Se necesitan dos tablas que posean campos que puedan ser utilizados para establecer las correspondencias. Y para esto, los campos necesitan ser del mismo tipo de dato y representar la misma entidad en ambas tablas.

Por ejemplo, si deseamos calcular la densidad de población para los departamentos de La Pampa, necesitamos contar en la misma tabla con el dato de

superficie calculado a partir de la geometría del campo “the_geom” de la tabla “departamentos” y el dato de población de la tabla “datos_deptos”.

	gid [PK] serial	cod_depto text	nom_depto text	cod_prov text	the_geom geometry		nbi real	hogares integer	ipmh real	poblacion integer	cod_depto [PK] character	nom_depto text	region text
1	1	42133	REALICO	42	0106000020A75	306	9.9	626	42.2	2070	42014	CALEU CALEU	SUR
2	2	42147	TRENEL	42	0106000020A75	307	7.6	29300	25	95973	42021	CAPITAL	SUR
3	3	42049	CHALILEO	42	0106000020A75	308	10	2051	30.8	6707	42028	CATRILO	SUR
4	4	42098	LOVENTUE	42	0106000020A75	309	8.6	4737	27.9	14521	42035	CONHELO	SUR
5	5	42140	TOAY	42	0106000020A75	310	22.6	279	53.8	845	42042	CURACO	SUR
6	6	42021	CAPITAL	42	0106000020A75	311	20.7	692	48.3	2414	42049	CHALILEO	SUR
7	7	42091	LIMAY MAHUID	42	0106000020A75	312	8.1	3369	25.8	10748	42056	CHAPALEUFU	SUR
8	8	42154	UTRACAN	42	0106000020A75	313	39	392	67.6	1451	42063	CHICAL CO	SUR
9	9	42056	CHAPALEUFU	42	0106000020A75	314	13.3	3050	28.2	9249	42070	GUATRACHE	SUR
10	10	42105	MARACO	42	0106000020A75	315	7.4	2695	27.7	7729	42077	HUCAL	SUR
11	11	42119	QUEMU QUEMU	42	0106000020A75	316	10.9	201	36.8	520	42084	LIMAY MAHUID	SUR
12	12	42028	CATRILO	42	0106000020A75	317	31.4	156	59	475	42091	LOVENTUE	SUR
13	13	42007	ATREUCO	42	0106000020A75	318	13.4	2551	39.9	8489	42098	LOVENTUE	SUR
14	14	42077	HUCAL	42	0106000020A75	319	7.9	16665	25.3	54235	42105	MARACO	SUR
						320	16.8	2008	45.3	7623	42112	PUELEN	SUR
						321	6.4	2883	23.8	8661	42119	QUEMU QUEMU	SUR
						322	14.1	3255	34.2	10571	42126	RANCOL	SUR
						323	7.4	4853	22.9	15168	42133	REALICO	SUR

Vemos que en ambas tablas se nos presenta el código de departamento que está almacenado como texto en los dos casos. Ésos serán los campos que utilizaremos para juntar las tablas.

Entonces, la consulta puede empezar con el pedido de campos de densidad y de población:

```
SELECT nom_depto, AREA(the_geom), poblacion ...
```

En este caso, estamos extrayendo el dato de superficie aplicando la función “AREA” al campo geométrico. Ahora agregamos los nombres de las tablas que contienen estos campos:

```
SELECT nom_depto, AREA (the_geom), poblacion ...
FROM departamentos JOIN datos_depto
```

Aquí hemos agregado los nombres de las tablas, y el comando “JOIN” que indica la operación de juntar tablas. Pero vemos que al enumerar los campos, pueden surgir ambigüedades, ya que, por ejemplo, las dos tablas contienen un campo “nom_depto”. Por eso, para especificar a qué tabla pertenece cada campo que solicitamos, colocaremos alias para renombrar a cada tabla.

```
SELECT a.nom_depto, AREA (a.the_geom), b.poblacion ...
FROM departamentos a JOIN datos_depto b...
```

Entonces, le hemos asignado un alias a cada tabla (“a” y “b”) y hemos utilizado este alias para indicar a qué tabla pertenece cada campo: “a.nom_depto” pertenece a “departamentos” que lleva el alias “a”; y “b.poblacion” pertenece a “datos_depto” que tiene el alias “b”.

Una vez definido los campos solicitados, las tablas de origen y sus alias, es necesario establecer la correspondencia que hará posible el proceso de juntar las tablas. En este caso, habíamos identificado que la correspondencia se deba por los campos “cod_depto” de ambas tablas. Entonces la consulta se completa con:

```
SELECT a.nom_depto, AREA (a.the_geom), b.poblacion
FROM departamentos a JOIN datos_depto b
ON a.cod_depto = b.cod_depto;
```

```
SELECT a.nom_depto, AREA (a.the_geom), b.poblacion
FROM departamentos a JOIN datos_depto b
ON a.cod_depto = b.cod_depto;
```

Panel de Salida

Salida de datos Comentar Mensajes Historial

	nom_depto text	area double precision	poblacion integer
1	REALICO	2543746803.76172	15168
2	TRENEL	1977869406.39648	5266
3	CHALILEO	8851365279.14258	2414
4	LOVENTUE	9081798801.16797	8489
5	TOAY	5102215923.37305	9107
6	CAPITAL	2520744305.57617	95973
7	LIMAY MAHUI	10150513600.666	475
8	UTRACAN	12802666196.9414	14240
9	CHAPALEUFU	2567198212.54883	10748
10	MARACO	2509397557.57031	54235

El comando “ON” especifica la relación mediante la cual se produce la correspondencia. En este caso, es la igualdad entre el campo “cod_depto” de la tabla “departamentos” y el campo “cod_depto” de la tabla “datos_depto”.

Ahora que tenemos los campos necesarios, podemos finalizar la consulta realizando la operación entre los campos para conseguir la densidad de población:

```
SELECT a.nom_depto, b.poblacion / (AREA (a.the_geom)/1000000) as
densidad
FROM departamentos a JOIN datos_depto b
ON a.cod_depto = b.cod_depto;
```

```
SELECT a.nom_depto, b.poblacion / (AREA (a.the_geom)/1000000) as densidad
FROM departamentos a JOIN datos_depto b
ON a.cod_depto = b.cod_depto;
```

Panel de Salida

Salida de datos Comentar Mensajes Historial

	nom_depto text	densidad double precision
1	REALICO	5.96285761521917
2	TRENEL	2.66246092030627
3	CHALILEO	0.27272628841658
4	LOVENTUE	0.93472671943671
5	TOAY	1.78491074011219
6	CAPITAL	38.0732785105165
7	LIMAY MAHUI	0.04679566164699
8	UTRACAN	1.11226831825093
9	CHAPALEUFU	4.18666542671394
10	MARACO	21.6127571481787
11	QUEMU QUEM	3.48248049381636
12	CATRILO	2.7254977705743
13	ATREUCO	2.85478098259199

Para finalizar, hemos realizado la transformación de metros cuadrados de la superficie a kilómetros cuadrados (división / 1.000.000) y el resultado lo usamos como denominador para calcular la densidad. El campo de resultado fue renombrado como “densidad”.

Otro ejemplo. Deseamos averiguar en qué región se encuentran los ríos navegables. El dato de la región no se encuentra en la tabla “rios”, pero sí en la tabla “censo2010”. Y entran dos tablas, comparten el nombre de la provincia (“provincia” y

“nom_prov” respectivamente) como atributo en común. Entonces, la consulta resultante será:

```
SELECT a.nombre, b.nom_region
FROM rios a JOIN censo2010 b
ON a.provincia = b.nom_prov
WHERE a.navegabilidad = 'SI'
```

<pre>SELECT a.nombre, b.nom_region FROM rios a JOIN censo2010 b ON a.provincia = b.nom_prov</pre>		
Panel de Salida		
Salida de datos Comentar Mensajes Historial		
	nombre text	nom_region character varying(255)
558	URUGUAY	NEA
559	URUGUAY	NEA
560	PARANA MINI	CENTRO
561	DE ACCESO	CENTRO
562	DE ACCESO	CENTRO
563	PARACAO	CENTRO
564	PARACAO	CENTRO
565	PARANA VIEJO	CENTRO
566	PARANA VIEJO	CENTRO
567	PARANA VIEJO	CENTRO
568	PARANA VIEJO	CENTRO
569	PARANA DE LAS PALMAS	CENTRO

En la parte final agregamos la condición “WHERE”, ya que solo deseamos obtener el dato del subuniverso de ríos que son navegables.



ACTIVIDAD 3

Escribir la consulta en el documento de texto, que devuelve los siguientes campos: el nombre del departamento, el perímetro en kilómetros y la cantidad de hogares.

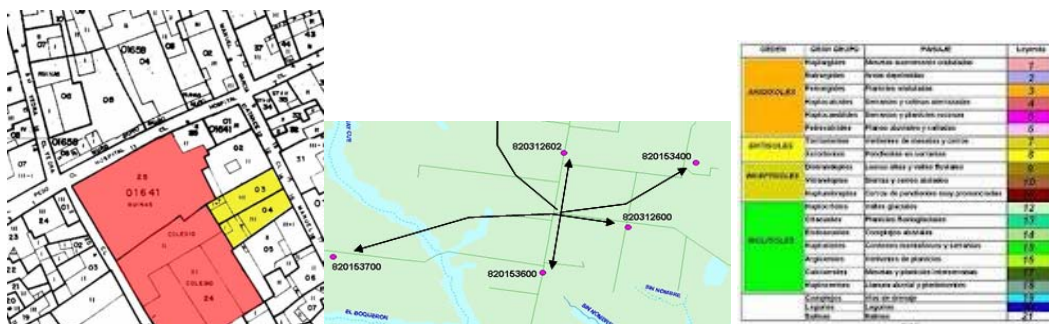
5 Utilización de códigos comunes para entidades geográficas

Como vimos anteriormente en los ejercicios de juntar tablas, la utilización de códigos para referirnos a entidades geográficas es muy útil para facilitar el manejo de la información. Los nombres de las entidades pueden variar con el tiempo, o pueden ser escritas de distintas formas y esto termina siendo un impedimento a la hora de relacionar distintas tablas. Es por esto que es muy recomendable incluir en las bases de datos geográficas, y de todo tipo, la codificación establecida por las instituciones competentes.

En nuestro país, el INDEC es uno de los organismos oficiales que más ha avanzado en la codificación de entidades geográficas. Por ejemplo, los códigos utilizados para provincias, departamentos y localidades que se utilizan en este curso, provienen del INDEC, y además son utilizados por un número creciente de instituciones.

	cod_prov character(2)	nom_prov character varying(255)		cod_depto character va	nom_depto text		cod_aglo text	nom_loc text
4	22	CHACO	1	18098	MBURUCUYA	924	1394	SARGENTO VIDAL
5	26	CHUBUT	2	06260	ESTEBAN ECHEVERRIA	925	1404	FERRI
6	14	CORDOBA	3	18126	SALADAS	926	1417	PASO CORDOVA
7	18	CORRIENTES	4	18028	CONCEPCION	927	1479	VILLA ALBERDI
8	30	ENTRE RIOS	5	18161	SAN ROQUE	928	1525	BARRIO CHACRA MONTE
9	34	FORMOSA	6	18105	MERCEDES	929	1236	CONTRALMIRANTE CORDERO
10	38	JUJUY	7	30077	NOGOYA	930	0142	ALLEN
11	42	LA PAMPA	8	30091	TALA	931	1513	BARRIO MOSCONI
12	46	LA RIOJA	9	30105	VICTORIA	932	0499	GENERAL FERNANDEZ ORO
13	50	MENDOZA	10	30021	DIAMANTE	933	0709	GENERAL ENRIQUE GODOY
14	54	MISIONES	11	30084	PARANA	934	1027	MAINQUE
15	58	NEUQUEN	12	06270	EZEIZA	935	3601	VILLA SAN ISIDRO
16	62	RIO NEGRO	13	18021	CAPITAL	936	6310	PENINSULA RUCA CO
			14	54014	CAINGUAS			

Otro ejemplo es el catastro, cuya unidad de análisis, la parcela catastral, también posee un código único normalizado que puede ser utilizado para una infinidad de propósitos. Por ejemplo, se puede geolocalizar todo tipo de fenómeno que ocurra en una parcela invocando el código único de cada una, como el domicilio de pacientes que requieren atención médica especial, establecimientos industriales, parcelas sin edificar, edificaciones precarias, etc.



De la misma manera, en cada jurisdicción, existen codificaciones referidas a establecimientos sanitarios, educativos, barrios, áreas protegidas, códigos para los distintos usos del suelo, tipo de cultivo, etc. que es necesario tener en cuenta a la hora de conformar una base de datos espaciales.



ACTIVIDAD 4

Responder en el documento de texto:

¿Cuales son las entidades geográficas que suelen representarse en su ámbito de trabajo?

¿Existen códigos o alguna normalización que se utilice para identificar dichas entidades?

6 Vistas con datos espaciales.

Hemos visto anteriormente como generar vistas a partir de consultas de selección. Estas vistas, cuando incluyen datos geométricos en la respuesta, pueden ser visualizadas desde gvSIG o cualquier cliente de bases de datos espaciales. Pero, al igual que las tablas de datos geométricos, necesitan cumplir algunos requisitos:

- Tener un campo geométrico
- Tener un campo que funcione como clave primaria.
- Estar registrados en la tabla geometry_columns.

La clave primaria, para una tabla geométrica, debe ser un campo numérico, de valores no nulos, y con valores únicos. Este requisito se logra fácilmente llamando al campo “gid” de la tabla de datos geométricos.

El registro de la columna geométrica en la tabla “geometry_columns” no es automático para las vistas, por lo cual es necesario hacerlo escribiendo directamente sobre la tabla, o bien a través de una consulta SQL.

```
INSERT INTO geometry_columns VALUES ('', 'esquema',  
'vista', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```

Por ejemplo, si creamos una vista que solo contenga el departamento “REALICO”, la definición de la vista “v_realico” será:

```
SELECT * FROM departamentos WHERE nom_depto = 'REALICO':
```

Recordemos que también podemos crear la vista corriendo la consulta:


```
CREATE VIEW v_realico AS  
SELECT * FROM departamentos WHERE nom_depto = 'REALICO';
```

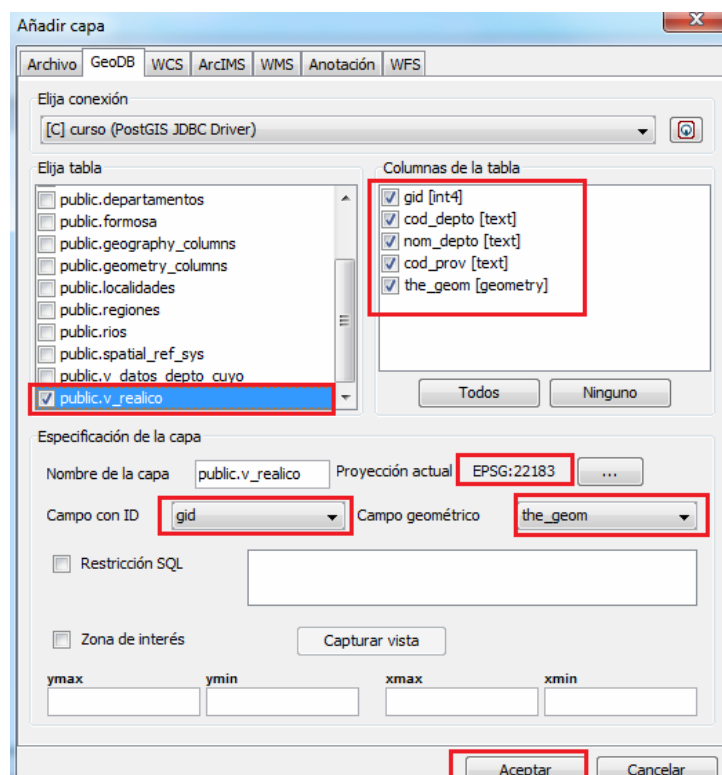
Y en este caso, cumplimos con tener el campo geométrico (the_geom), la clave primaria (gid) y finalmente creamos el registro con la consulta:

```
INSERT INTO geometry_columns VALUES ('', 'public',  
'v_realico', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```

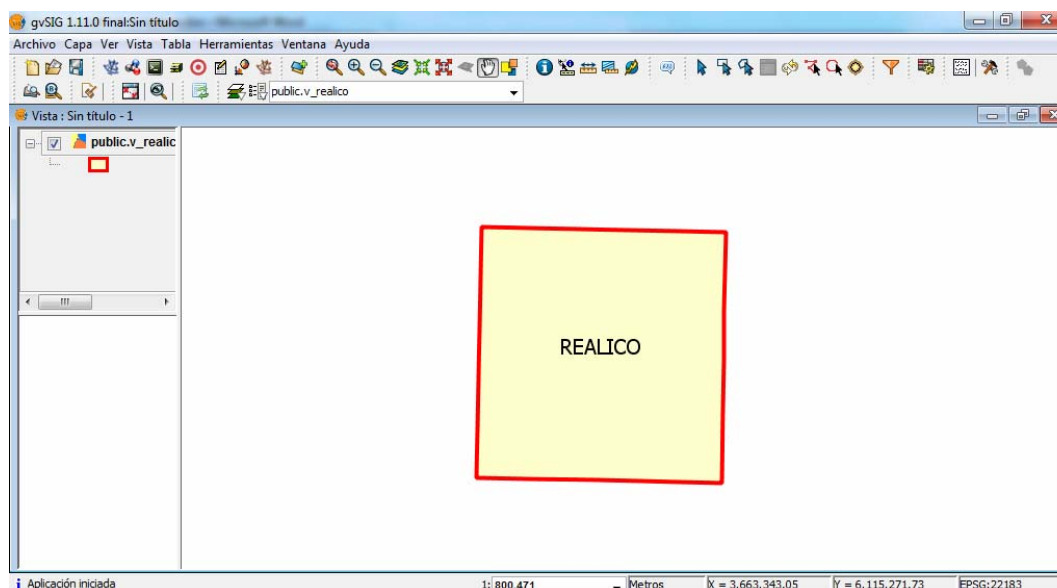
Luego de agregar el registro en “geometry_columns” podemos visualizar la vista desde gvSIG.

Abrimos una vista, la configuramos con el EPSG 22183, presionamos el botón de

Agregar capa  y luego nos posicionamos en la solapa de *Bases de datos espaciales*. Allí seleccionamos la conexión a la base de datos y vemos aparecer en el listado la nueva vista “v_realico”.



Al seleccionarla, vemos que aparece el nombre del campo geométrico (the_geom) y el de la clave primaria (gid). Si damos "Aceptar" vemos como la nueva capa se agrega a la vista. Y podremos comprobar que el comportamiento de una capa generada a partir de una vista es idéntico al de las capas que provienen directamente de las tablas con datos geométricos.



ACTIVIDAD

Crear una vista que junte los datos de la tabla "departamentos" con el campo "poblacion" de la tabla "datos_deptos". Recordar que para juntar estas dos tablas se usa la correspondencia "a.cod_depto = b.cod_depto".

Registrar la vista en “geometry_columns”.

Visualizar la vista desde gvSIG.

Abrir la tabla de atributos, imprimir pantalla y pegar la imagen en el documento de texto.

7 SQL espacial: consulta con resultados geométricos.

En esta clase continuaremos avanzando en el uso de SQL aplicando funciones espaciales. En la clase pasada vimos como extraer información implícita en el campo geométrico. En esta oportunidad veremos cómo generar nueva información geográfica a partir de los datos existentes.

Los resultados de estas consultas serán geometrías, por lo cual podremos visualizarlos desde gvSIG usando las vistas.

7.1 Centroid

Esta función con parámetros “centroid(geometry)” devuelve un punto que representa el centro geométrico de la figura. Puede ser aplicado tanto a polígonos, como a líneas y también a figuras múltiples.

Por ejemplo, si necesitamos representar los departamentos a través de una figura de puntos, para graficar una variable mediante símbolos de diferente tamaño, podemos convertir los mismos departamentos a puntos, sin tener que crear una nueva tabla. Para esto aplicamos la función “centroid” al campo geométrico de los departamentos:

```
SELECT gid, nom_depto, CENTROID (the_geom) as the_geom FROM departamentos;
```

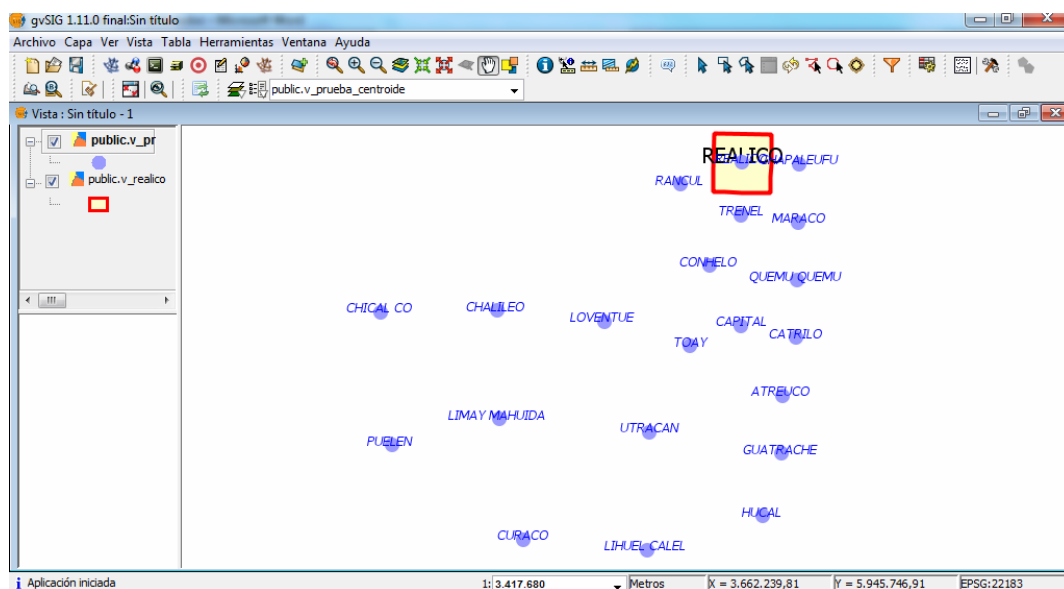
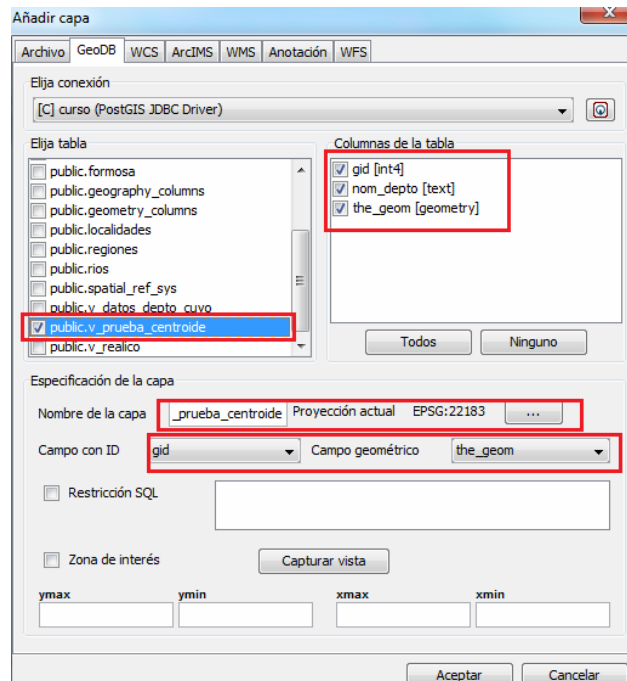
La consulta nos devuelve un campo geométrico con el alias “the_geom”, difícil de interpretar, a menos que lo visualicemos desde gvSIG. Previamente, creamos una vista y luego la registramos.

```
CREATE VIEW v_prueba_centroide AS  
SELECT gid, nom_depto, CENTROID(the_geom) as the_geom FROM departamentos;
```

```
INSERT INTO geometry_columns VALUES ('', 'public',  
'v_prueba_centroide', 'the_geom', 2, 22183, 'POINT');
```

En la consulta que inserta el registro en “geometry_columns” solo hay que tener en cuenta que la geometría resultante de “centroid” es el punto.

Con la vista creada y registrada, vamos a gvSIG, y vemos que la capa agregada es de puntos. En la tabla de atributos vemos también los datos que hemos decido tener en cuenta durante la creación de la vista.

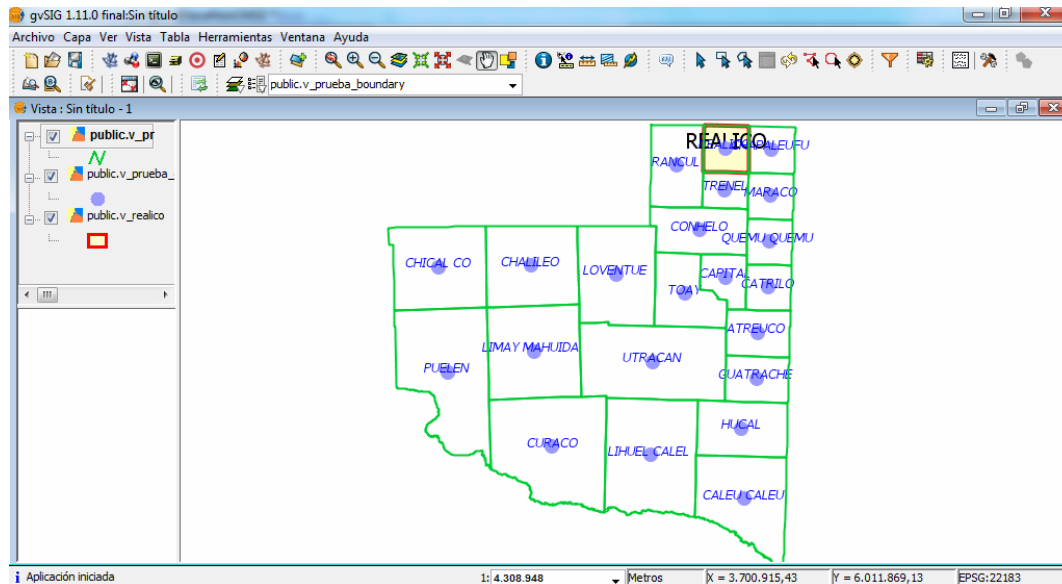


7.2 Boundary

Esta función se aplica a los polígonos, y devuelve la línea que los envuelve. Lo aplicaremos a los polígonos de departamentos, y veremos como agregar las nuevas líneas en gvSIG.

```
CREATE VIEW v_prueba_boundary AS
SELECT gid, nom_depto, BOUNDARY(the_geom) as the_geom FROM
departamentos;
```

```
INSERT INTO geometry_columns VALUES ('', 'public',
'v_prueba_boundary', 'the_geom', 2, 22183, 'MULTILINESTRING');
```



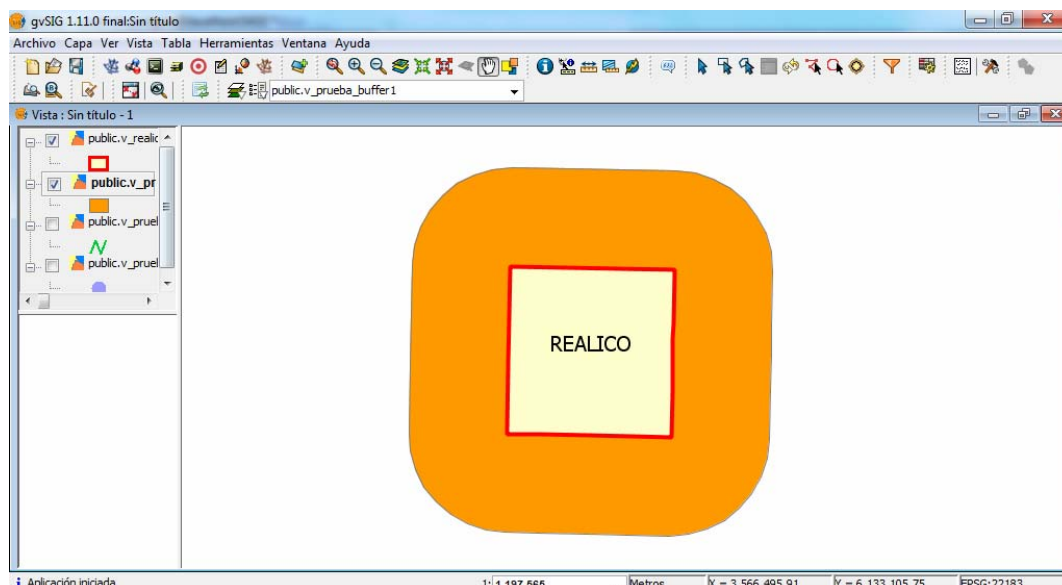
7.3 Buffer

Es el área de influencia o envolvente. Al igual que en los SIG de escritorio, existen diferentes alternativas a la hora de general un área de influencia. Puede ser a utilizando una distancia fija: `BUFFER(geometry,distancia)`

```
CREATE VIEW v_prueba_buffer1 AS
SELECT gid, BUFFER (the_geom, 30000) as the_geom
from departamentos where nom_depto = 'REALICO';

INSERT INTO geometry_columns VALUES ('', 'public',
'v_prueba_buffer1', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```

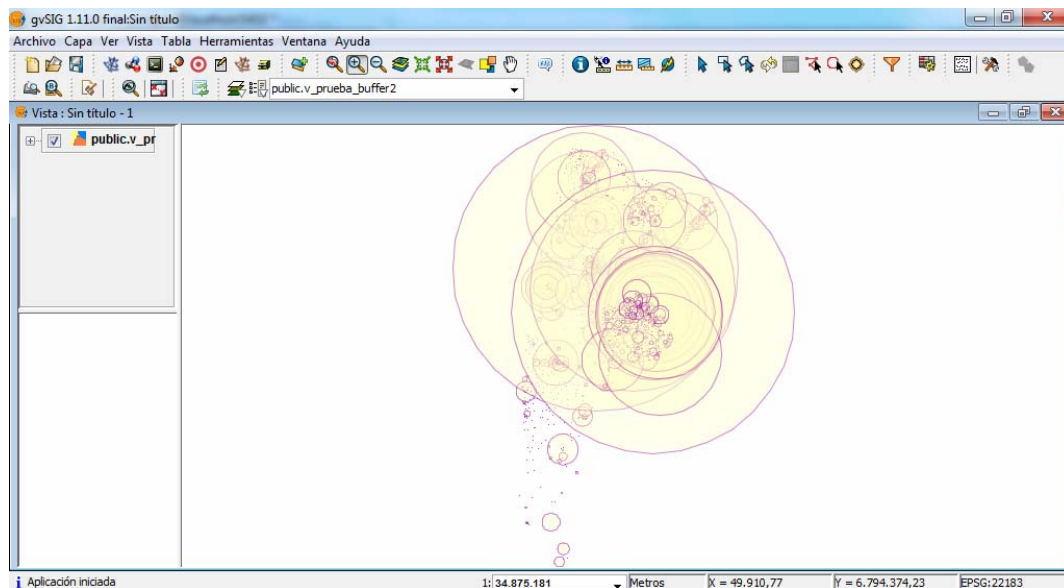
En este caso, hemos definido que el ancho del área de influencia sea de 30.000 metros (unidades de las coordenadas) o 30 km.



También se puede determinar que el ancho de la envolvente dependa de algún atributo de la propia tabla. Simplemente se reemplaza el valor fijo, por el nombre del campo a tener en cuenta. Por ejemplo, podemos crear envolvente de las localidades en función de su población.

```
CREATE VIEW v_prueba_buffer2 AS
SELECT gid, BUFFER (the_geom, poblacion) as the_geom
from localidades;

INSERT INTO geometry_columns VALUES ('', 'public',
'v_prueba_buffer2', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```

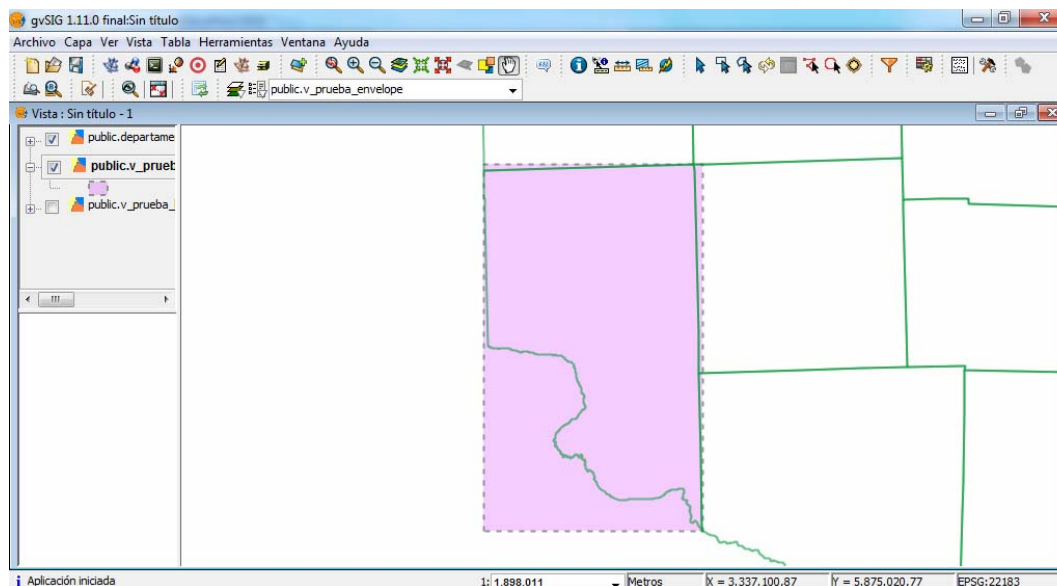


7.4 Envelope

Es un polígono rectangular o cuadrado que se crea tomando los valores máximos y mínimos de “X” e “Y” de cualquier elemento geométrico. Luego veremos que esta función es muy útil para agilizar el manejo de la información espacial, ya que localiza rápidamente la zona a la cual pertenece cada elemento. La estructura de la función es “ENVELOPE (geometry)”

```
CREATE VIEW v_prueba_envelope AS
SELECT gid, ENVELOPE (the_geom) as the_geom
from departamentos where nom_depto = 'PUELEN';

INSERT INTO geometry_columns VALUES ('', 'public',
'v_prueba_envelope', 'the_geom', 2, 22183, 'MULTIPOLYGON');
```



ACTIVIDAD 5

- Realizar una vista con centroides de la tabla “departamentos”.
- Seleccionar uno de los tipos de buffer y aplicar a una capa existente o una capa propia migrada a Postgres.
- Realizar una impresión de pantalla de los resultados y agregar al documento de texto.
- Copiar en el documento de texto las consultas que fueron utilizadas para crear las vistas y las de los registros en “geometry_columns”.

Bibliografía

OLAYA, Víctor. (2011). *Sistemas de Información Geográfica*. Versión 1.0, Rev. 24 de marzo de 2011. Proyecto “Libro Libre SIG”.

http://forge.osor.eu/docman/view.php/13/577/Libro_SIG.zip

THE POSTGIS TEAM. Manual de PostGIS 1.5.3.

<http://www.postgis.org/download/postgis-1.5.3.pdf>

OBE, Regina y HSU Leo. (2011). *PostGIS in Action*. Editorial Manning. Stamford.

Índice

1 INTEGRIDAD REFERENCIAL	1
1.1 RESTRICCIONES.....	2
1.2 CLAVE PRIMARIA.....	2
1.3 CLAVE ÚNICA.....	7
1.4 CLAVES AJENA.....	9
2 VISTAS.....	13
3 SQL: CONSULTAS Y SUBCONSULTAS.....	16
4 SQL: JOIN O JUNTAR TABLAS.....	17
5 UTILIZACIÓN DE CÓDIGOS COMUNES PARA ENTIDADES GEOGRÁFICAS	20
6 VISTAS CON DATOS ESPACIALES.....	21
7 SQL ESPACIAL: CONSULTA CON RESULTADOS GEOMÉTRICOS.....	24
7.1 CENTROID.....	24
7.2 BOUNDARY.....	25
7.3 BUFFER.....	26
7.4 ENVELOPE.....	27
BIBLIOGRAFÍA	29

Usted es libre de compartir - copiar, distribuir, ejecutar y comunicar públicamente y de hacer obras derivadas de este documento

Este documento es una obra compartida bajo la licencia Creative Commons.

Atribución-NoComercial-CompartirIgual 2.5 Argentina (CC BY-NC-SA 2.5)



Atribución — Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciante (pero no de una manera que sugiera que tiene su apoyo o que apoyan el uso que hace de su obra).



No Comercial — No puede utilizar esta obra para fines comerciales.



Compartir bajo la Misma Licencia — Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.

Aviso: Al reutilizar o distribuir la obra, tiene que dejar muy en claro los términos de la licencia de esta obra. La mejor forma de hacerlo es enlazar a la siguiente página:

<http://creativecommons.org/licenses/by-nc-sa/2.5/ar/>

Programa Nacional Mapa Educativo
Ministerio de Educación
República Argentina

Teléfono/Fax: 54 11 4129-1408
Correo electrónico: mapaedu_nac@me.gov.ar
www.mapaeducativo.edu.ar



