Progettare un Database Geografico con UML

Claudio Rocchini (*rockini@tele2.it*)– Istituto Geografico Militare

Introduzione

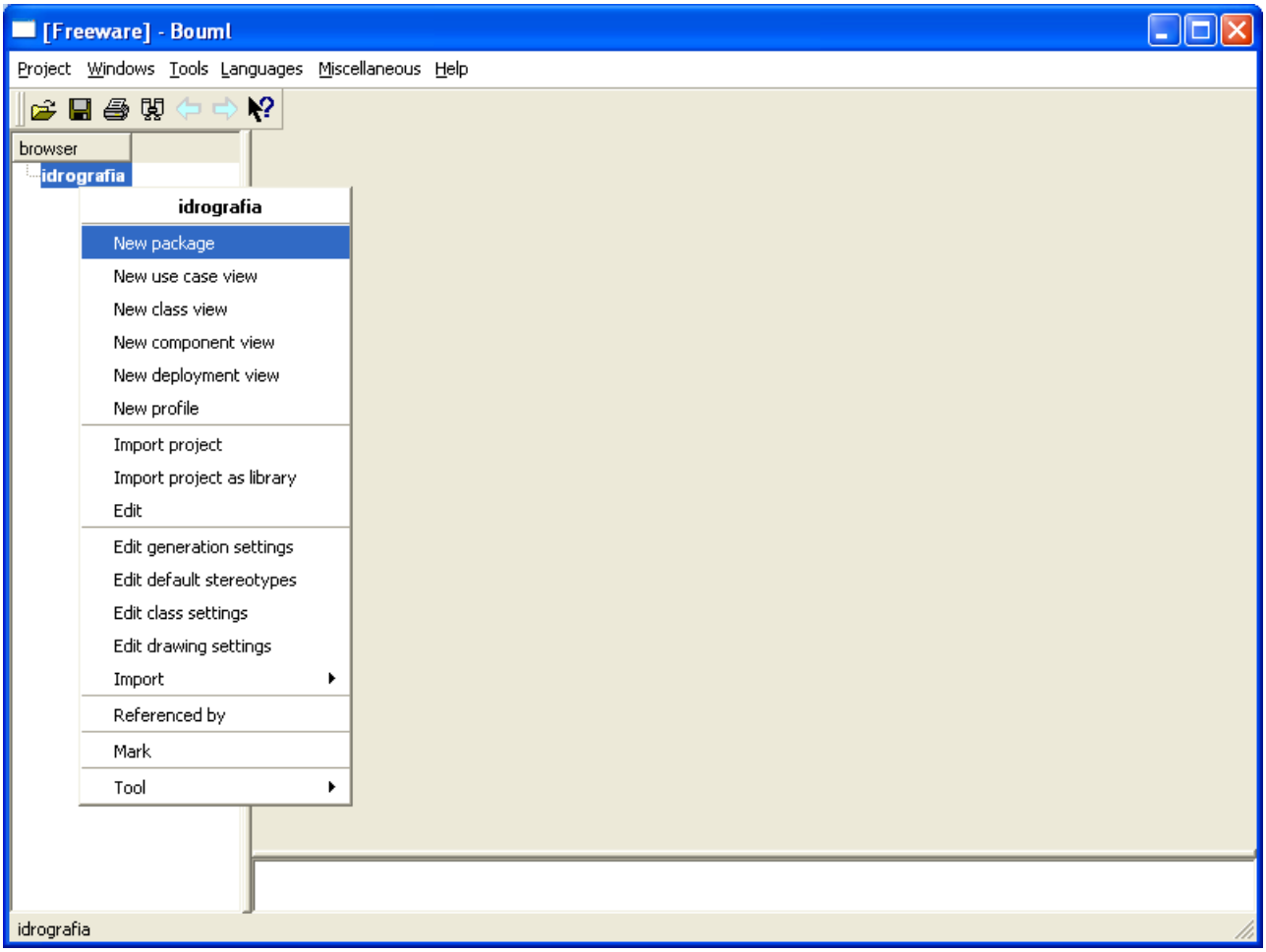
In questa breve nota si vuole introdurre i principi di progettazione tramite il linguaggio UML, con particolare riferimento alla progettazione di un database geografico. I software utilizzati in questa presentazione sono: Bouml (http://bouml.free.fr), Postgres-PostGIS (http://www.postgresql.org) ed lo strumento di esportazione *postgis* sviluppato dall'IGM. L'installazione di questi software esula dagli scopi di questo corso.

Obiettivo

Il nostro scopo è quello di modellare una piccola parte del database geografico "Catalogo dei Dati Territoriali" in uso presso le regioni italiane. Il nostro db conterrà le feature corso d'acqua (areale) ed elemento idrico (asse lineare). Per introdurre i concetti di generalizzazione e relazione (1 a n, n a n), si costruiranno due classi astratte (feature e geometria) da cui deriveremo le nostre feature concrete. Inoltre due attributi apparterranno ad un dominio rappresentato da due tabelle di dominio.

Costruzione dello schema

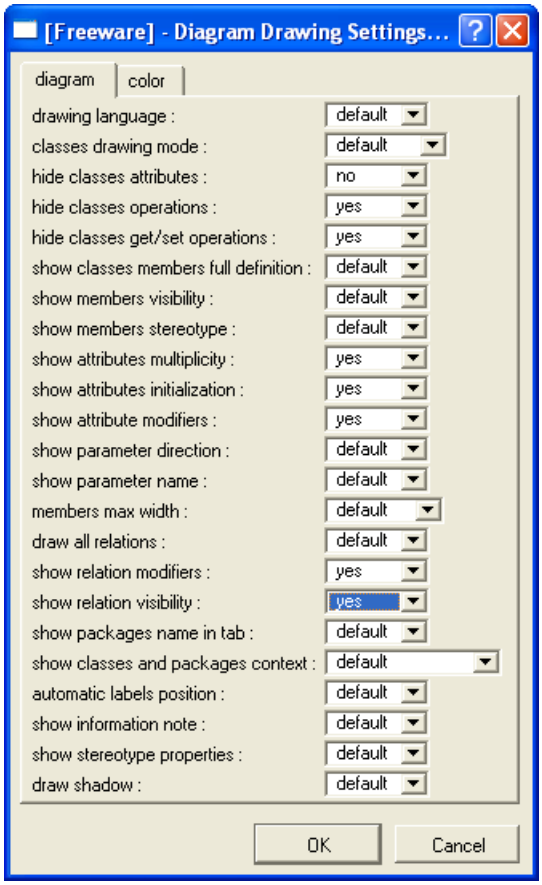
Lancia il programma BoUML, seleziona il menù Project –New e salva il progetto nella cartella desiderata con il nome di "idrografia". Nel browser, clicca col bottone destro su idrografia e seleziona "New package", quindi crea il nuovo package col nome di "public".



I *packages* del nostro schema corrisponderanno a *schemi* della base di dati, in particolare questo primo package conterrà i dati pubblici, comuni a tutte le sezioni di dati geografici. Ripeti l'operazione e crea un secondo package col nome di "sezione", questo secondo package-schema conterrà i dati geografici di una particolare sezione di cartografia.

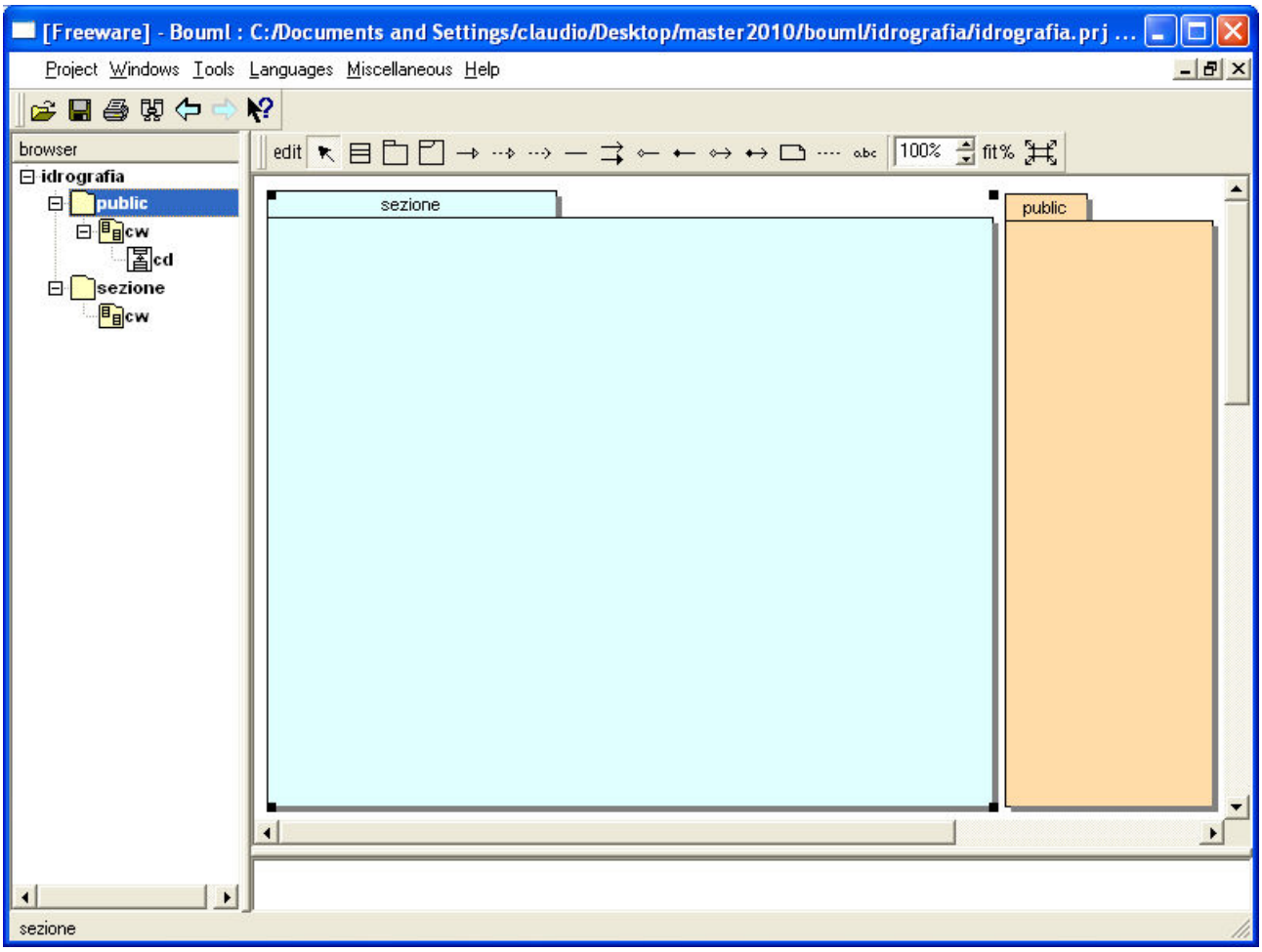
Sempre tramite il menù del bottone destro crea all'interno di ogni package una "class view" (menù "New Class View"), i nomi non sono importanti (es. mettere "cw"); con questa operazione indichiamo il fatto che realizzeremo una schema di classi in UML.

All'interno della prima class view crea un "class diagram" (menù "New Class Diagram"), anche in questo caso il nome non è importante (es. mettete "cd"). Il diagramma sarà la nostra rappresentazione grafica dello schermo. Dopo aver creato il diagramma cliccaci col bottone destro e seleziona la voce "show" in modo da visualizzare il diagramma stesso nella finestra a destra, quindi seleziona il menù "Edit drawing settings" e: attiva tutte le opzioni di visualizzazione della classe, degli attributi e delle relazioni, disattiva inoltre la visualizzazione degli operatori (sia normali che get/set) dato che le entità del database non hanno operatori. Queste impostazioni di visualizzazione ci mostreranno graficamente tutti i dati salienti del nostro schema.

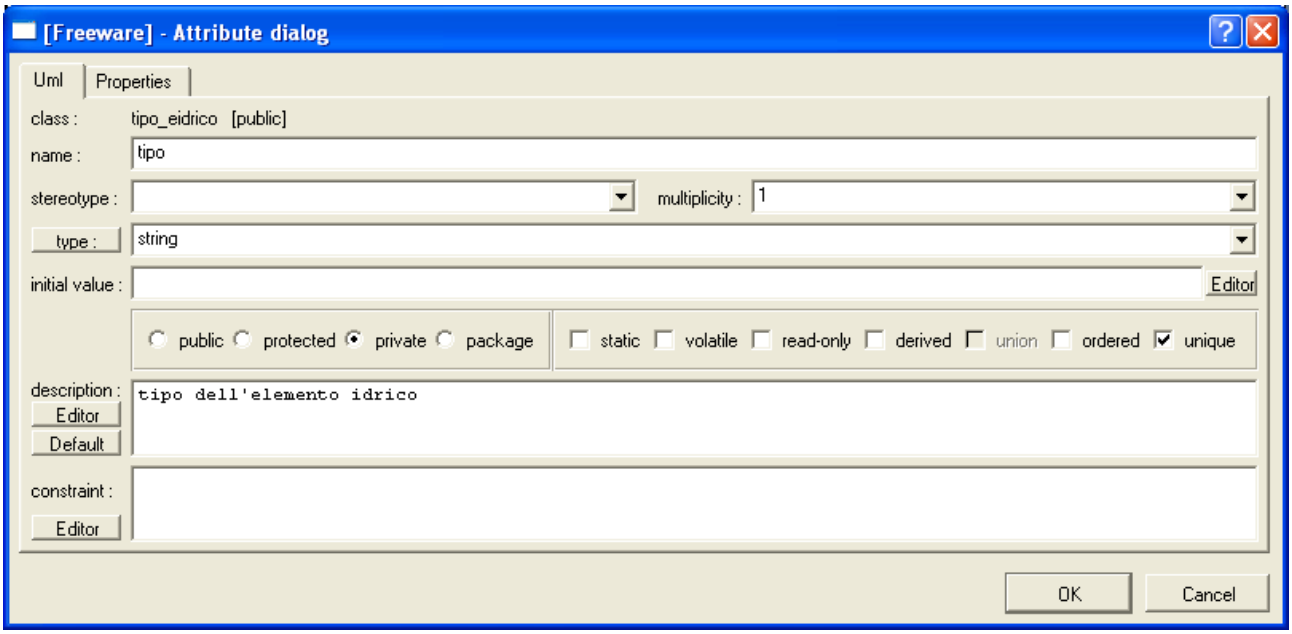


Adesso trascina i due package nello schema e allargali in modo da occupare tutta le finestra. Il package "public" avrà bisogno di meno spazio.

Entriamo nel merito dei contenuti e creiamo la nostra prima classe (entità, ovvero tabella di db), all'interno del package (schema) "public". La nostra prima tabella rappresenterà il dominio dell'attributo tipo dell'elemento idrico. Clicca col bottone destro sulla class view (cw) dentro il package "public" e selezionate "New class" per creare una nuova classe. Chiama la classe "tipo_eidrico", quindi trascinate la nuova classe nello schema grafico e posizionala dentro il package "public". Abbiamo creato la nostra prima classe. Il passo successivo prevede l'aggiunta degli attributi.

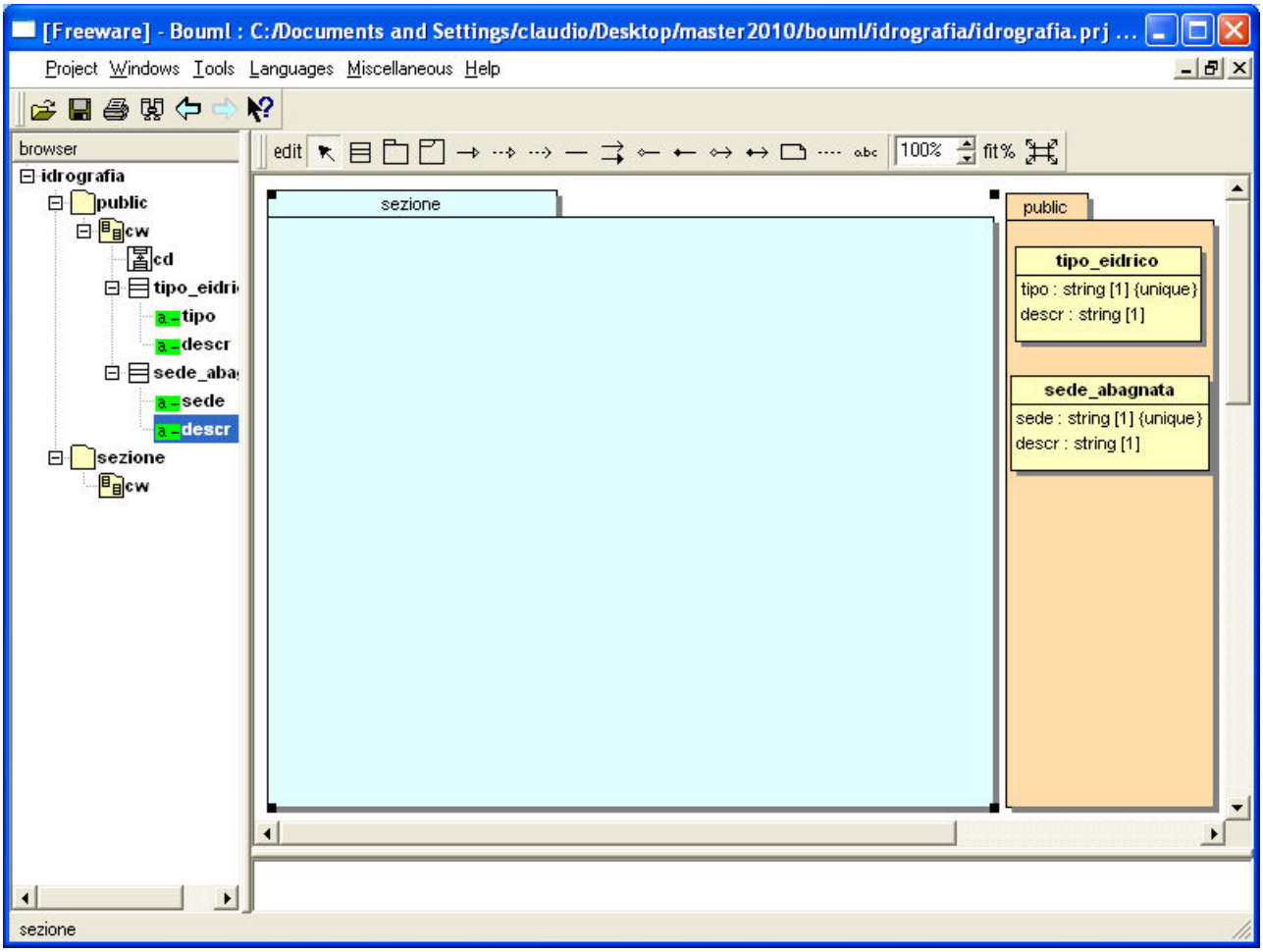


Clicca col bottone destro sulla classe appena creata (si può fare sia sullo schema grafico che sul browser degli elementi a sinistra). Scegli il menù “Add attribute” e aggiungi l’attributo con nome “tipo”.



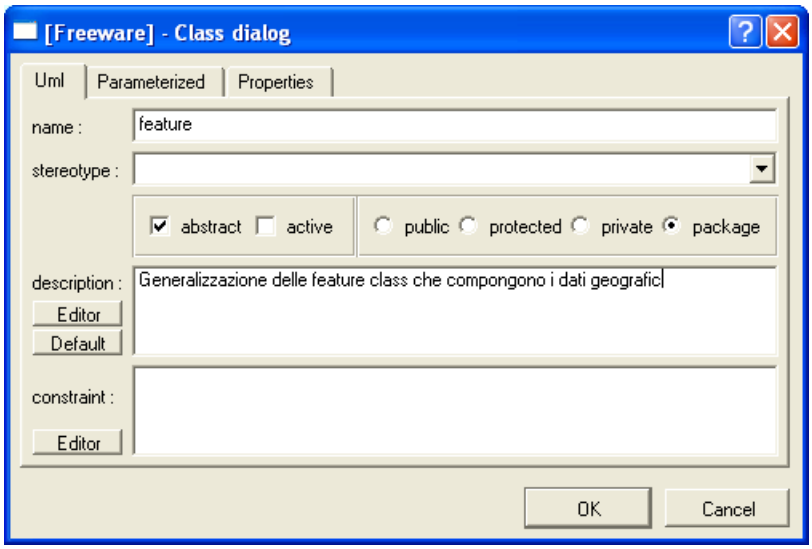
Dopo la creazione dell’attributo si apre il dialogo per i dettagli sull’attributo stesso. Seleziona come multiplicity il valore “1”, questo vuol dire che l’attributo è obbligatorio e non può essere nullo. L’altra

possibilità per gli attributi delle basi di dati è l'opzione "0..1", la quale significa che l'attributo è facoltativo.



Le altre opzioni non hanno senso in un database relazionale, dato che non sono previsti attributi multi valore. In seguito seleziona "varchar" come type, dato che i valori di tipo sono stringhe. Infine seleziona l'opzione "unique", dato che questo attributo corrisponde alla chiave primaria dell'entità; quindi premi "OK" dato che il nostro attributo è completo.

Adesso ripetiamo l'operazione di creazione di attributi, per creare il secondo attributo: nome "descr" (descrizione del tipo), sempre type "varchar", sempre multiplicity "1", ma questa volta l'opzione "unique" non va selezionata: questo attributo non fa parte della chiave primaria.



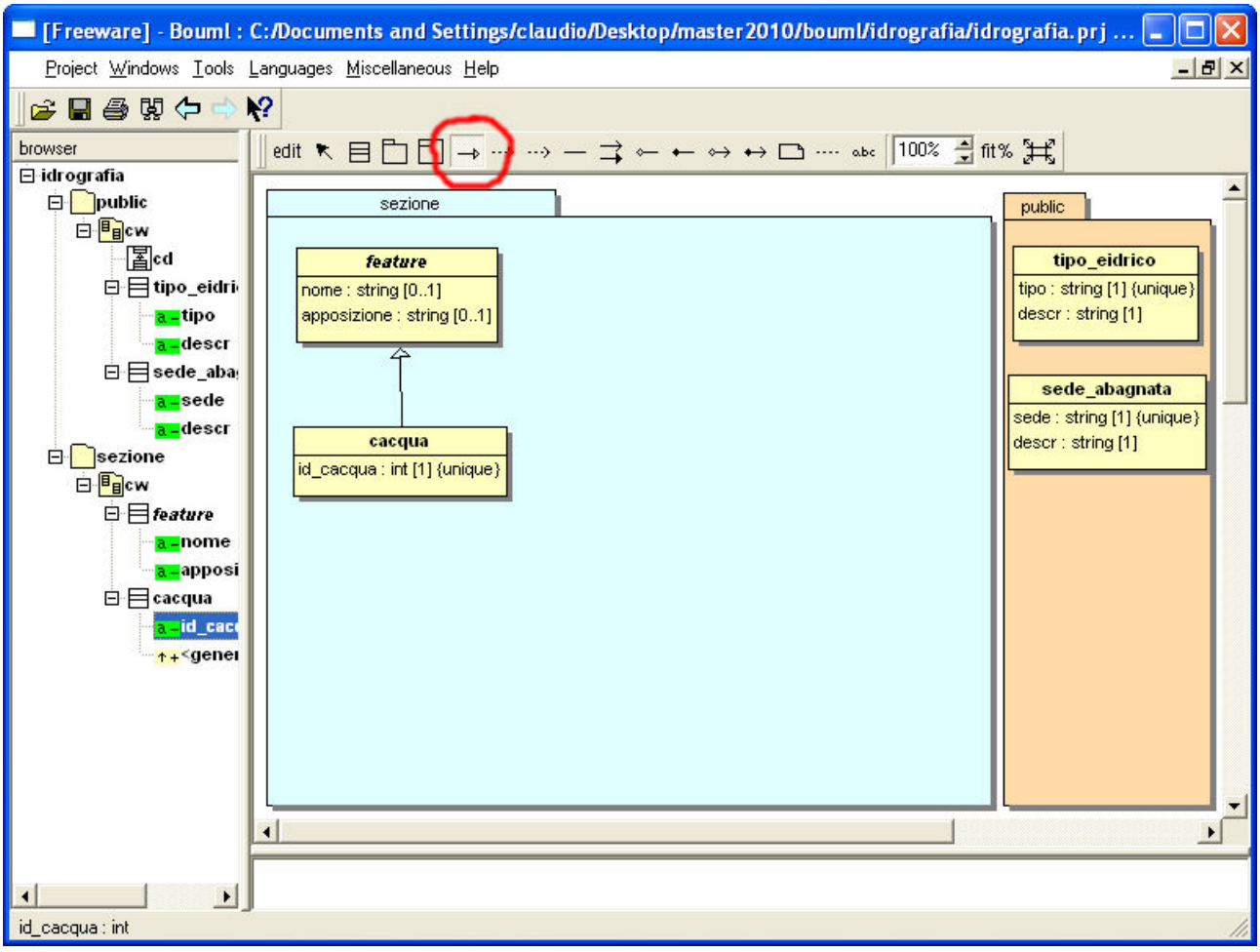
La stessa sequenza di operazioni deve essere ora ripetuta per creare una seconda classe, che rappresenta il dominio dell'attributo sede di area bagnata. Adesso siamo bravi e possiamo creare una nuova classe del package "public" che si chiama "sede_abagnata", con gli attributi sede (chiave primaria) e descr. A questo punto il package "public" è completo.

Passiamo allo schema “sezione”. La prima classe che creiamo sarà una classe astratta, che generalizza tutte le feature class contenenti nel db geografico. Clicca sul package “sezione” e seleziona “New class”. Scrivi “feature” come nome, quindi trascina la nuova classe nello schema grafico, dentro il suo package. Adesso clicca sulla classe col bottone destro e seleziona il menù “Edit class”: si aprirà il dialogo con le proprietà della classe. Seleziona l’opzione “abstract”, dato che questa è una classe astratta (vale a dire che non può esistere di per sé, ma deve essere concretizzata in classi più definite). Il fatto che una classe sia astratta è visualizzato dal nome della classe scritto in corsivo. Ovviamente in ogni oggetto del nostro schema è possibile definire il campo “description”.

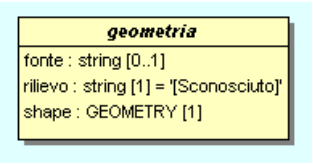
Una volta create la classe “feature”, aggiungiamo due attributi alla classe: “nome” e “apposizione” entrambi di tipo “varchar” con multiplicity “0..1”, vale a dire entrambi opzionali. Questi attributi saranno comuni a tutte le classi feature derivate.

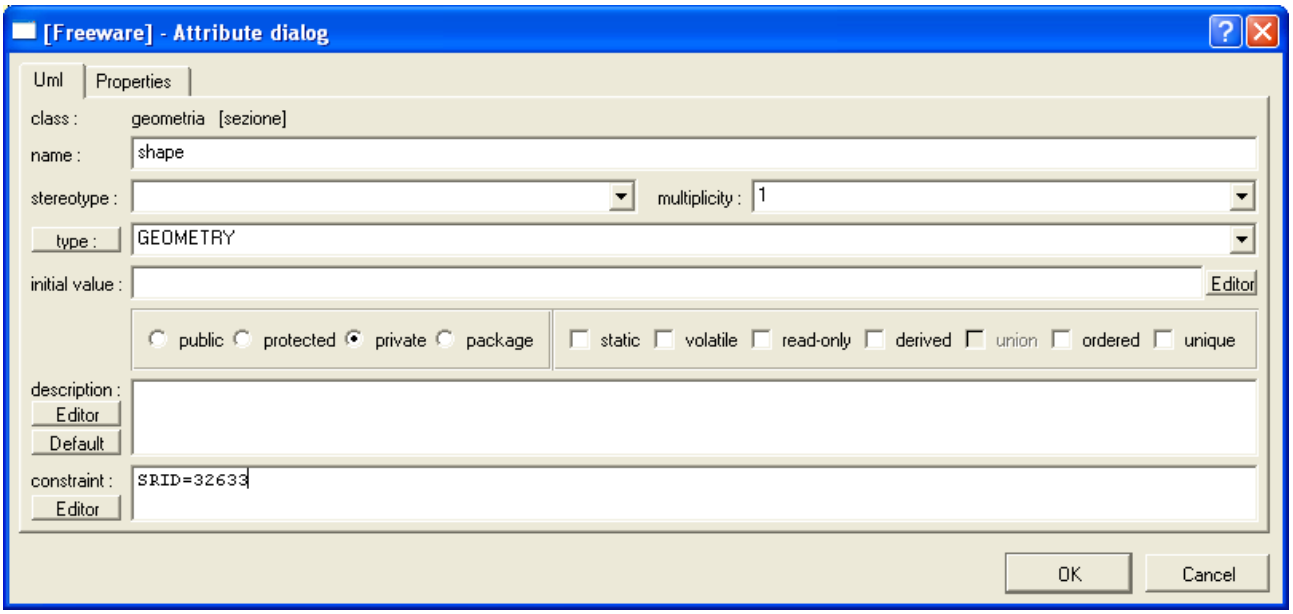
Selezionando sempre il package “sezione” aggiungiamo adesso la classe “cacqua”, la trasciniamo nello schema ed aggiungiamo l’unico attributo proprio “id_cacqua” di tipo “int”, multiplicity “1” e unico (vale a dire chiave primaria). Questa classe rappresenta i corsi d’acqua naturali (fiumi, torrenti, etc.).

Adesso dobbiamo rendere la classe “cacqua” figlia della classe feature. Per farlo, seleziona il tool “generalization” (simbolizzato da una freccia con linea intera a testa bianca), quindi clicca sulla classe “cacqua” e trascina la relazione su feature. Se tutto va bene appare la freccia di relazione fra le due classi. La freccia a testa bianca indica che la classe figlia (cacqua) discende dalla classe “feature” e quindi ne eredita tutte le caratteristiche, per cui ad esempio la classe “cacqua” contiene anche gli attributi “nome” e “apposizione”.

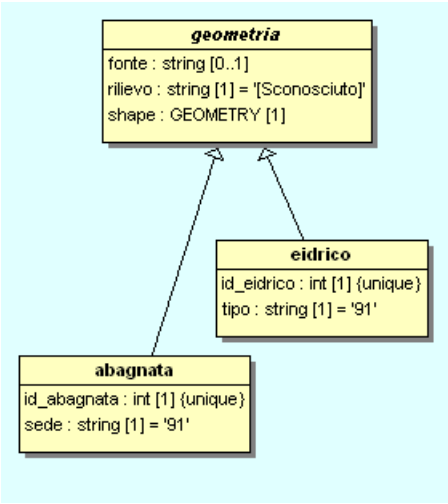


Il prossimo passo prevede la creazione della classe astratta che rappresenta la generalizzazione di tutte le classi geometriche. Chiameremo questa classe “geometria”, anche questa volta la editiamo e la segniamo come astratta. La nostra classe astratta sarà per prima cosa dotata degli attributi “rilievo” (varchar non obbligatorio) e rilievo (varchar obbligatorio). Il secondo attributo è obbligatorio ma non è detto che questo valore sia riempito in prima istanza.





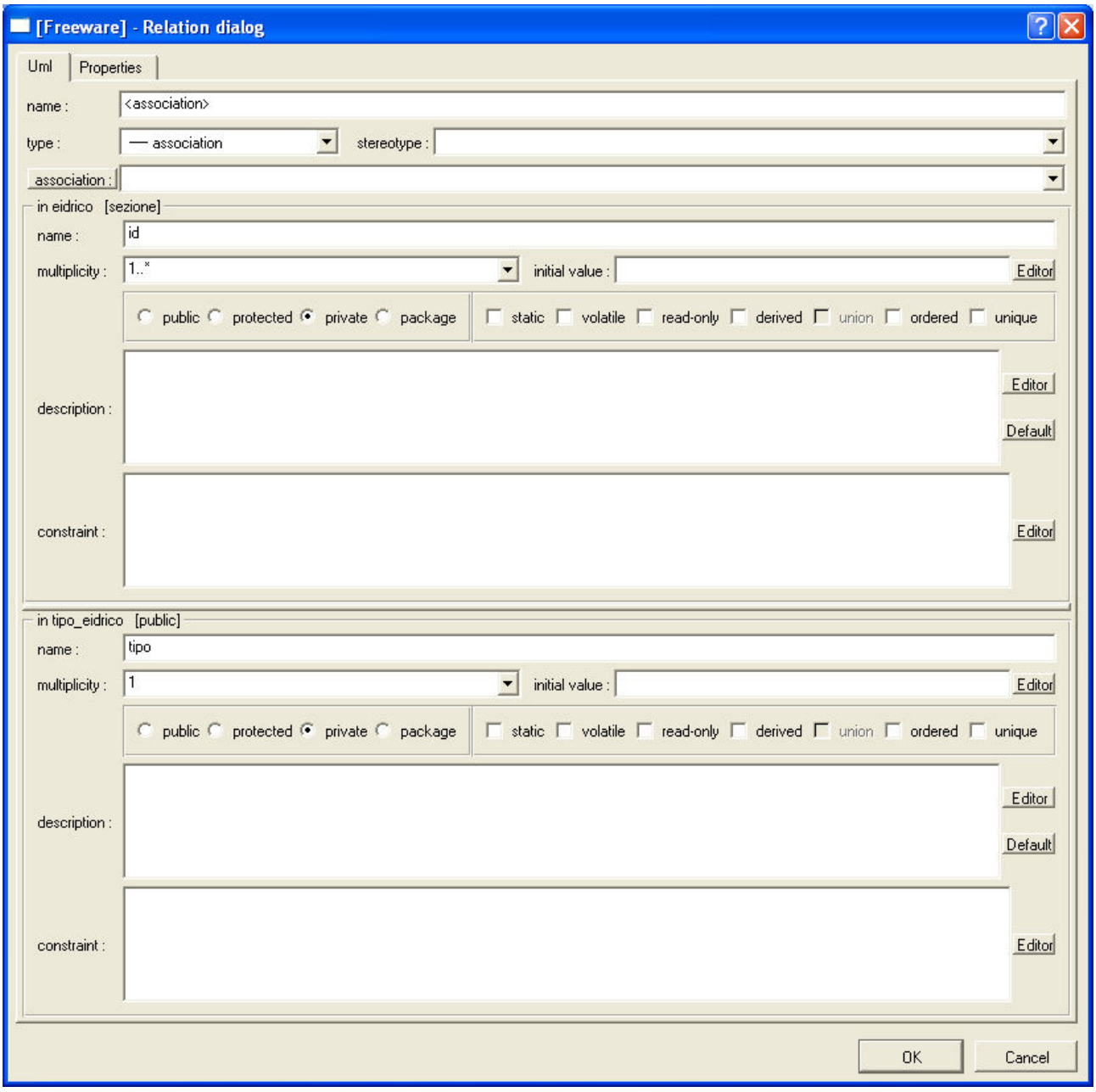
Per questo motivo definiamo (nel dialogo di definizione dell’attributo) il campo “initial value” a “[Sconosciuto]”, vale a dire che questo sarà il valore di default nel caso in cui non venga inserito esplicitamente. L’ultimo attributo da aggiungere è quello fondamentale, vale a dire l’attributo geometrico. Chiameremo questo attributo “shape” e porremo semplicemente il tipo a “GEOMETRY” (scritto maiuscolo); questo fa si che la classe associata diventi una classe spaziale, vale a dire che contiene dati geografici. L’attributo GEOMETRY ha bisogno inoltre di altre specificazioni: nel campo “constraint”, indichiamo la dicitura “SRID=32633”, questo vuol dire che il sistema di riferimento associato a questo campo spaziale è il codice EPSG 32633, vale a dire UTM-WGS84 fuso 33 Nord.



Una volta definita la classe astratta, definiamo due classi geometriche concrete: area bagnata “abagnata” ed elemento idrico “eidrico”. Entrambe le classi saranno figlie di “geometria”. La prima classe contiene gli attributi propri “id_abagnata” (intero, obbligatorio ed unico) e l’attributo “sede” (varchar, obbligatorio, con valore di default ‘91’ = non assegnato), che corrisponde al dominio definito dalla classe “sede_abagnata”. La seconda classe contiene gli attributi propri “id_eidrico” (intero, obbligatorio e unico), e l’attributo “tipo” (varchar, obbligatorio, con valore di default ‘91’).

Entrambe le classi quindi vengono rese figlie di “geometria” (ne ereditano le caratteristiche), disegnando le due relazioni di generalizzazione.

Il prossimo passo prevede la connessione degli attributi “tipo” e “sede” ai loro domini. Per farlo selezioniamo il tool “Association” (linea intera senza testa ne coda), puntiamo su “eidrico” e trasciniamo la relazione su “tipo_eidrico”. Una volta creata l’associazione, bisogna definirne i parametri: cliccate col bottone destro sulla linea che rappresenta la relazione e scegliete il menù “edit”.



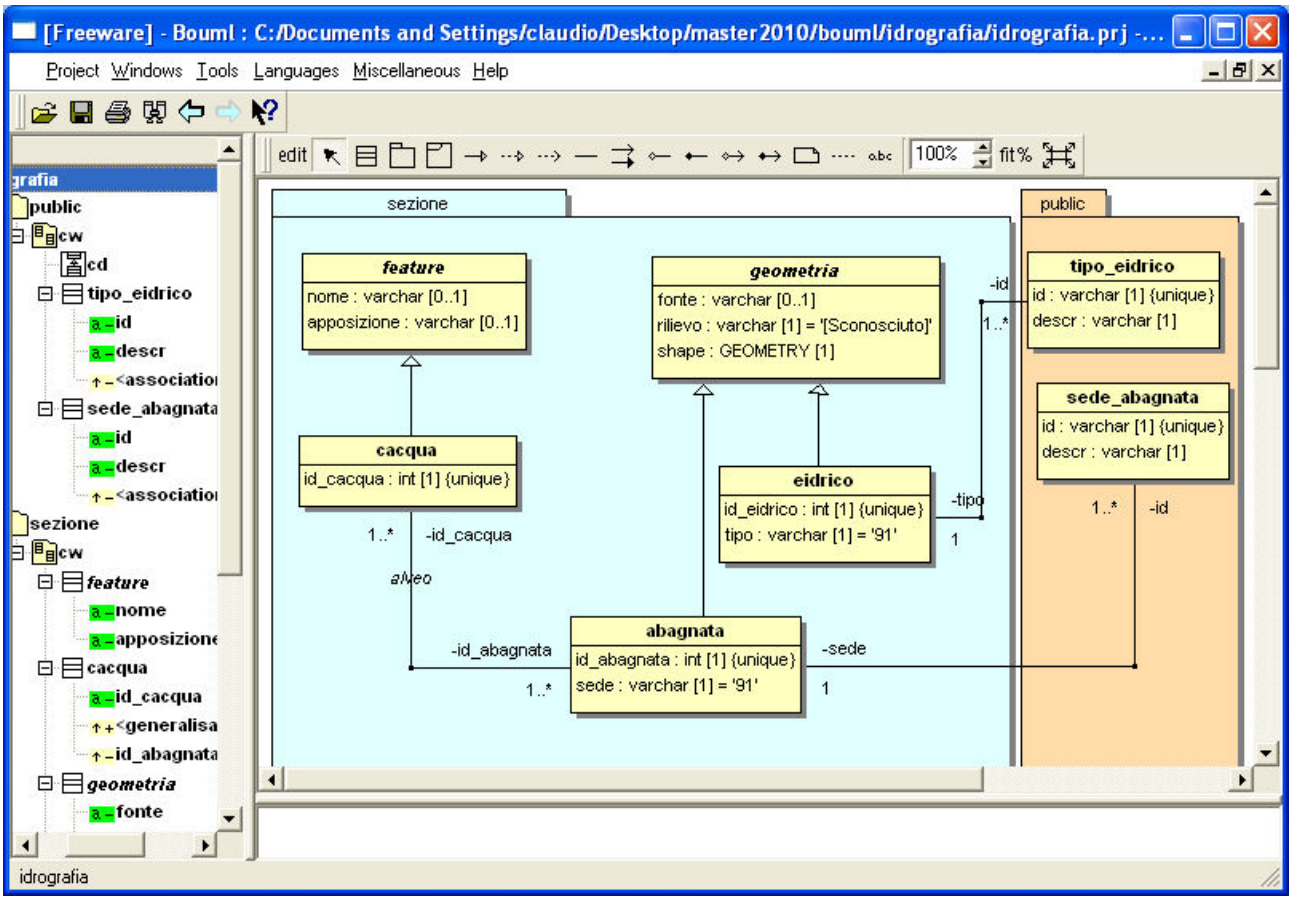
Dovete indicare due cose per ogni capo dell’associazione: il nome della parte e la molteplicità. Il nome della parte corrisponde all’attributo che forma la relazione (vi ricordate le relazioni dei db?). Nel nostro caso il nome in *eidrico* è *id* e la molteplicità è *1..*,* mentre dalla parte di *tipo_edrico* il nome è tipo e la molteplicità è *1*.

A questo punto create da voi la giusta associazione fra l’attributo *sede* di *abagnata* e l’attributo *id* di *sede_abagnata*.

Infine creiamo la relazione *n* a *n* fra *cacqua* e *abagnata*: la procedura è la stessa di sopra ma entrambi i lati della relazione hanno cardinalità 1..*.

Struttura Finale

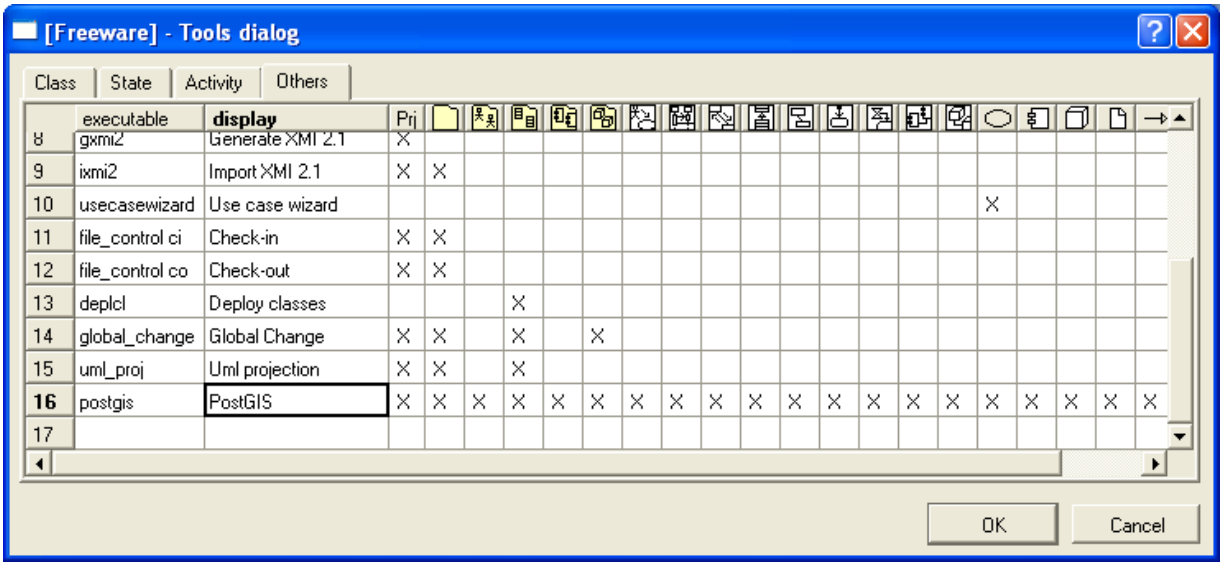
La figura di seguito rappresenta la situazione finale del nostro progetto.

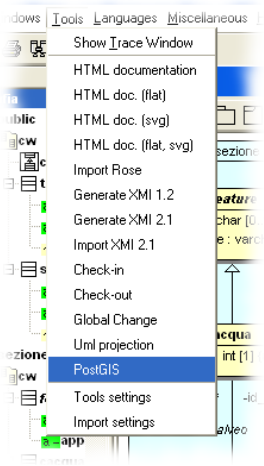


Ad esempio, la classa *caqua* è formata da insiemi di aree *abagnata* e deriva dalla classe feature. A sua volta *abagnata* è una *geometria*.

Generazione del database

Per prima cosa bisogna controllare che lo strumento *postgis* (sviluppato da IGM) sia attivo. Se la dicitura *postgis* non compare nel menù *Tool*, bisogna attivarlo. Selezionate sempre il menù *Tool* e selezionate il sotto-menù *Tools Settings*, quindi selezionate il tab *Others*.





Andata sull'ultima riga della tabella, cliccate sulla prima colonna (*executable*) e digitate *postigs*. Fate lo stesso sulla seconda colonna (*display*); quindi checcate tutti i flags della riga come nella figura soprastante. Alla fine premete ok: lo strumento è attivo.

Per generare il vostro database basta a questo punto selezionare il menù Tools – Postgis, selezionare il nome del file di salvataggio e attendere. Se non si sono fatti errori, il report di generazione riporterà l'elenco delle feature esportate. In caso di errore (es. molteplicità non prevista, tipo di associazione non prevista, etc.) il report ci darà informazioni sull'errore commesso.

Analisi del file generato

Lo strumento di esportazione genera un file di comandi SQL. Se si è seguito l'esercizio con precisione, il file generato sarà simile al testo seguente:

```
-- ***** PostGIS Generator v1.0 (C)2010 Istituto Geografico Militare *****
-- **** Schema public ****
CREATE TABLE public.tipo_eidrico
(       id varchar NOT NULL PRIMARY KEY,
        descr varchar NOT NULL
);
CREATE TABLE public.sede_abagnata
(       id varchar NOT NULL PRIMARY KEY,
        descr varchar NOT NULL
);
-- **** Schema sezione ****
CREATE SCHEMA sezione;
CREATE TABLE sezione.feature -- Generalizzazione delle feature …
(       nome varchar,
        apposizione varchar
);
CREATE TABLE sezione.cacqua
(       id_cacqua int NOT NULL PRIMARY KEY
)INHERITS (sezione.feature);
CREATE TABLE sezione.geometria
(       fonte varchar,
        rilievo varchar NOT NULL DEFAULT '[Sconosciuto]',
        shape GEOMETRY NOT NULL
);
CREATE TABLE sezione.abagnata
(       id_abagnata int NOT NULL PRIMARY KEY,
        sede varchar NOT NULL DEFAULT '91'
)INHERITS (sezione.geometria);

CREATE TABLE sezione.eidrico
(       id_eidrico int NOT NULL PRIMARY KEY,
        tipo varchar NOT NULL DEFAULT '91'
)INHERITS (sezione.geometria);

-- **** Relations ****
CREATE TABLE public.cacqua_abagnata_link
(       id_abagnata int NOT NULL REFERENCES sezione.abagnata(id_abagnata),
        id_cacqua int NOT NULL REFERENCES sezione.cacqua(id_cacqua)
);
ALTER TABLE sezione.abagnata ADD CONSTRAINT abagnata_sede_FK
        FOREIGN KEY (sede) REFERENCES public.sede_abagnata(id);
ALTER TABLE sezione.eidrico ADD CONSTRAINT eidrico_tipo_FK
        FOREIGN KEY (tipo) REFERENCES public.tipo_eidrico(id);
```

Alcune note sul file SQL generato:

- I package vengono convertiti in schemi di database
- Le classi e gli attributi vengono convertiti in tabelle con le relative colonne del tipo indicato. Le tabelle sono inserite nello schema (package) di provenienza.
- La caratteristica {unique} corrisponde ad una chiave primaria e viene convertità correttamente.
- Le cardinalità degli attributi 0..1 e 1 corrispondono rispettivamente a NULL o NOT NULL.
- Il valore di default vengono riportati così come sono in SQL.
- L'ereditarietà delle classi viene costruita con l'opzione INHERITS. Questa opzione di Postgres permette di introdurre il concetto (derivante dai db orientati agli oggetti) di ereditarietà.
- Le relazioni 1..n diventano vincoli di referenza.
- Le relazioni n..n diventano tabelle di associazione con vincoli di referenza.

Riferimenti

1. Postgres: http://www.postgresql.org
2. Postgis: http://postgis.refractions.net
3. BoUML: http://bouml.free.fr
4. UML Group: http://www.uml.org