

CAPÍTULO 6

CONSULTA, MANIPULAÇÃO E APRESENTAÇÃO DE DADOS GEOGRÁFICOS EM LEGAL

“Inventar uma linguagem poderia significar: inventar, com base em leis naturais (ou, em concordância com elas), uma aparelhagem para uma determinada finalidade; tem, porém., um outro sentido também, análogo aquele em que falamos na invenção de um jogo.”

L. Wittgenstein (Investigações Filosóficas).

6.1 APRESENTAÇÃO

O modelo de dados e as operações geográficas definidas anteriormente servem de base para a proposta de uma linguagem de consulta e manipulação espacial, denominada LEGAL (**L**inguagem **E**spacial para **G**eoprocessamento **A**lgébrico). O objetivo de LEGAL é prover um ambiente geral para análise geográfica, incluindo operações sobre geo-campos e geo-objetos.

O que se fará neste capítulo é apresentar um esboço da sintaxe dos operadores desta linguagem e mostrar alguns exemplos de seu uso, deixando o detalhamento para trabalhos posteriores. Os operadores e palavras reservadas da linguagem estão apresentados em COURIER MAIÚSCULAS.

6.2 BREVE REVISÃO DA LITERATURA

Conforme observado anteriormente, a discussão das operações de manipulação (dita álgebra de mapas, cf. Tomlin, 1990) é feita, na literatura, de forma separada da análise das operações de consulta e apresentação (Egenhofer, 1989). Na maioria dos sistemas comerciais para Geoprocessamento, estas operações são implementadas em pacotes distintos.

6.2.1 ÁLGEBRA DE MAPAS DE TOMLIN

Para realizar operações de modelagem espacial em geo-campos, a abordagem mais corrente em sistemas de Geoprocessamento é utilizar a linguagem *MAP* (Map Analysis Package), proposta por Dana Tomlin (1990). Esta proposta serve de base para muitos SIGs comerciais.

Em seu trabalho, Tomlin apresenta um conjunto de operações sobre mapas temáticos e modelos numéricos de terreno, definidos de forma a operar em geometrias matriciais, que incluem operações pontuais, de vizinhança (por ele chamadas de *focais*) e zonais.

A implementação mais difundida das idéias de Tomlin é a linguagem GRID, disponível no sistema ARC/INFO. Esta linguagem permite aos usuários realizar operações como¹:

<pre> OUTGRID = INGRID1 + INGRID2 OUTGRID = INGRID1 XOR 5 OUTGRID = SIN(INGRID1) * 4 / LOG(INGRID2) </pre>
--

A linguagem GRID, embora muito flexível, tem o sério inconveniente de não distinguir entre os diferentes tipos de operandos. Isto decorre do fato de ser fracamente tipada. Nos exemplos acima, se INGRID1 for do tipo MNT, e INGRID2 do tipo TEMÁTICO, o resultado pode não ter sentido. Em GRID, um mapa temático

¹Utilizaremos o fonte tipográfico “courier” toda vez que nos referirmos a comandos a ser executados por um sistema computacional.

no formato matricial é chamado de “grade de inteiros”, e um modelo numérico de terreno de “grades de ponto flutuante”, o que estabelece uma confusão entre o dado geográfico e sua representação. Veja-se a seguinte explicação, retirada do manual do sistema ARC/INFO (ESRI, 1991):

“For both local and zonal functions, if any of the input grids are floating point, the output grid will contain floating-point values. If all of the input grids are integer grids, the output grid is an integer grid, with the exception of those local and zonal functions that are statistically based.”

Em função deste tipo de problemas, optamos por fazer de LEGAL uma linguagem fortemente tipada, estando os operadores ligados a um contexto semântico, definido pelas diferentes especializações de geo-campos. Diferenças adicionais de LEGAL com relação às linguagens GRID e MAP incluem:

- disponibilidade de funções de tratamento de imagens;
- operações de classificação contínua e modelagem ambiental;
- chamada a funções externas, definidas pelo usuário.

6.2.2 LINGUAGENS DE CONSULTA ESPACIAL

A maior parte das implementações de *linguagem de consulta* utiliza o padrão SQL para ter acesso aos atributos dos dados geográficos. No entanto, não é possível expressar relações topológicas diretamente em SQL, o que tem dado origem a muitas propostas de linguagens de consulta espacial (Egenhofer, 1989; Alves, 1990; OOi, 1990). Estas propostas são, de uma maneira geral, uma extensão da linguagem SQL com operadores que expressam relações espaciais. Cada linguagem tem seu conjunto próprio de operadores de extensão. As conclusões destas propostas de extensão da linguagem SQL são (Frank and Mark, 1991):

- estender SQL com operadores espaciais pode ser feito sem maiores problemas, desde que a semântica destes operadores tenha sido definida formalmente;

- as consultas em Geoprocessamento incluem restrições complexas, que requerem planejamento cuidadoso ao ser traduzidas para SQL;
- estender SQL para incluir controle de apresentação e apontamento é possível, mas ainda não existe qualquer sintaxe ou fluxo de ações compatível com a requerida pelos padrões cognitivos da linguagem natural.
- aconselha-se projetar uma linguagem separada para lidar com as questões de apresentação de dados, e não incluir estes comandos na cláusula SQL padrão "select-from-where".

A proposta do novo padrão para a linguagem SQL (SQL 3) contém um conjunto de extensões para lidar com dados espaciais. SQL 3 inclui tipos de dados para representar dados espaço-temporais, dados geométricos; seus operadores topológicos estão baseados nos trabalhos de Egenhofer e Franzosa (1991) e seus tipos de dados no padrão SAIF (Surveys and Resource Mapping Branch, 1993).

Para definir as operações de consulta espacial em LEGAL, partimos das idéias apresentadas por Egenhofer (1989), Egenhofer (1992), Egenhofer (1994), e Hemerly (1993a) e do padrão ODMG-93 (Cattell, 1994).

6.3 CARACTERÍSTICAS DE LEGAL

As principais características de LEGAL são:

- LEGAL é orientada-a-objetos; o resultado das operações é um conjunto de objetos ou um conjunto de valores.
- LEGAL é definida de forma semelhante ao padrão ODMG-93 (Cattell, 1994). Seguindo este padrão, as operações de consulta são implementadas utilizando uma sintaxe baseada em SQL.
- Tanto operações de consulta espacial quanto operações de manipulação fazem parte da sintaxe da linguagem.
- As operações sobre campos, as operações combinadas campos-objetos e a linguagem de apresentação são implementadas por operadores do mesmo nível semântico da linguagem SQL.

LEGAL é *fortemente tipada*. Cada dado pertence a uma dos seguintes classes:

1. Classes básicas do modelo de dados, como segue:

- GEOOBJECT para instâncias de GEOOBJETO;
- NONSPATIAL para instâncias de NÃOESPACIAL;
- NETWORK para instâncias de REDE;
- CADASTRAL para instâncias de CADASTRAL;
- THEMATIC para instâncias de TEMÁTICO;
- IMAGE para instâncias de DADO SENSOR REMOTO;
- DTM para instâncias de MODELO NUMÉRICO DE TERRENO.

Estes dados tem os atributos básicos indicados na seção 4.5.5. No que segue, estes atributos básicos serão sempre expressos em maiúsculas.

2. Especializações das classes básicas, criadas através dos mecanismos de definição do esquema conceitual.
3. Classes auxiliares:
 - PROJECT como partição geográfica do banco de dados, com os atributos básicos: PROJECTION (indica a projeção cartográfica), DATUM, LATLONG_BOX (indica as coordenadas geodésicas do retângulo envolvente), PROJ_BOX (indica as coordenadas de projeção do retângulo envolvente).
 - TABLE para as tabelas de transformação entre geo-campos.
 - MASK para máscaras aplicáveis a filtros.
 - RECTANGLE e CIRCLE, utilizadas para definir vizinhanças.
4. Conjunto de objetos recuperados do banco:
 - COLLECTION: agrupamento de objetos de classes diferentes.
5. Tipos de dados tradicionais INT, FLOAT, CHARACTER e BOOLEAN.
6. Tipo de dado MULTIMIDIA, usado para armazenar arquivos de texto, imagens, áudio e vídeo².

²Numa primeira aproximação, este tipo de dados poderia ser apenas um nome de arquivo contendo uma apresentação multimídia, que seria decodificado por seu conteúdo.

6.4 DEFINIÇÃO DE DADOS EM LEGAL

A parte de definição de dados em LEGAL inclui:

- criar e definir bancos de dados geográficos;
- definir o esquema do banco de dados;
- criar instâncias das classes deste esquema;

6.4.1 DEFINIÇÃO DE ESQUEMA CONCEITUAL

Em LEGAL, o esquema do banco de dados é criado pela especialização das classes básicas do modelo e das classes derivadas das classes básicas. A sintaxe

```
CREATE <especialização> (<atributos gerais>)
IS_A <classe básica> | <especialização>
      (atributos específicos)
```

será utilizada para especializar o modelo, como ilustrado nos exemplos seguintes:

- Criar a classe “Uso do Solo”, especialização de mapa temático, com os temas “Floresta Primária”, “Cerrado” e “Floresta Secundária”, e com os atributos “área” e “anos de regeneração”³.

```
CREATE Uso_do_Solo
      ( ATTRIBUTE nome          STRING;
        ATTRIBUTE ano          INT; }
IS_A THEMATIC
      (THEMES = { "Floresta Primária",
                 "Cerrado", "Floresta Secundária" });
```

³O atributos indicados em maiúsculas indicam atributos herdados da classe básica.

- Criar a classe “Hospital”, especialização de `geo_objeto`, e com os atributos “nome”, “número de leitos”, “receita do SUS”.

```
CREATE Hospital
(ATTRIBUTE nome          STRING;
 ATTRIBUTE num_leitos    INT;
 ATTRIBUTE receita_SUS   FLOAT;)
IS_A GEOOBJECT;
```

- Criar a classe “Mapa Municipal”, especialização de `CADASTRAL`, com os atributos: número da planta, data do levantamento e empresa responsável.

```
CREATE Mapa_Municipal
(ATTRIBUTE nome          STRING;
 ATTRIBUTE num_planta    INT;
 ATTRIBUTE data_levantamento DATE;
 ATTRIBUTE empresa_respons  STRING)
IS_A CADASTRAL;
```

A definição de classe pode admitir um valor por omissão (`DEFAULT`) que serve para inicializar as instâncias de forma conveniente. Vejamos um exemplo da criação de uma classe de imagens, que por omissão serão sempre instanciadas com resolução de 30m.

```
CREATE Landsat_TM
(ATTRIBUTE nome          STRING;
 ATTRIBUTE resolucao_h = 30 INT;
 ATTRIBUTE resolucao_v = 30 INT;
 ATTRIBUTE banda_espectral INT)
IS_A IMAGE;
```

6.4.2 CRIAÇÃO DE GEO-OBJETOS E PLANOS DE INFORMAÇÃO

A criação explícita de instâncias de cada classe é obtida pelo comando `NEW`. Para o caso de geo-campos e de objetos cadastrais, será preciso ainda indicar qual a representação associada (matricial ou vetorial) e os parâmetros associados a cada representação. Tomemos os exemplos abaixo.

- Criar uma instância da classe `Uso_do_Solo` denotada por `acao_antropica_1990`, cuja representação matricial será uma grade de 100 x 100 metros de resolução, na escala 1:250.000.

```
VARIABLE acao_antropica_1990 Uso_do_Solo;  
  
acao_antropica_1990 = NEW  
(nome = "ação antrópica 1990",  
ano = 1990)  
REPRESENTED_BY RASTER  
(RESOLUCAO_H = 100 m,  
RESOLUCAO_V = 100 m,  
ESCALA = 1/250.000);
```

- Criar uma instância da classe Hospital, denominada hospital_SJC:

```
VARIABLE hospital_SJC Hospital;  
  
hospital_SJC = NEW  
  
(nome = "Hospital Municipal SJC",  
num_leitos = 300);
```

- Criar uma instância da classe MAPA_MUNICIPAL, para conter o mapa da cidade de São José dos Campos, cuja representação vetorial estará na escala 1:100.000, na projeção UTM e cujo retângulo envolvente possui coordenadas geográficas (-27°30", -45°00", -27°00", -44°30"),

```
VARIABLE mapa_SJC Mapa_Municipal;  
  
mapa_SJC = NEW  
  
(nome = "mapa SJCampos",  
num_planta = 25,  
data_levantamento = 1990,  
empresa_resp = "Aerodata")  
  
REPRESENTED_BY VECTOR  
  
(ESCALA = 100.000,  
PROJECAO = UTM,  
LAT_LONG_BOX= (-27°30", -45°00", -27°00", -44°30"));
```

6.5 CONSULTA ESPACIAL EM LEGAL

A consulta espacial em LEGAL tem dois componentes: uma *expressão de busca* expressa em SQL estendido e uma *resposta de consulta* que pode ser objeto de manipulação posterior.

A consulta espacial é aplicável a geo-objetos e geo-campos. No caso de geo-campos, dada a sua natureza contínua, os operadores disponíveis em LEGAL permitem apenas a recuperação baseada em seus atributos⁴. O mesmo procedimento vale para *geo-objetos complexos* (objetos cadastrais e de rede). Para os geo-objetos, LEGAL oferece mecanismos para recuperação baseada em atributos e em restrições espaciais.

6.5.1 EXPRESSÃO DE BUSCA

Em LEGAL, uma expressão de busca é construída a partir da linguagem OQL, com as seguintes extensões:

1. As restrições espaciais sobre geo-objetos serão sempre computadas a partir de suas representações geométricas num mapa de geo-objetos.
2. A cláusula `SELECT` pode permitir a recuperação de instâncias de classes em LEGAL e de valores de atributos destas instâncias.
3. A cláusula `FROM` contém a indicação das classes de objetos sobre as quais se realiza a consulta e do objeto cadastral (ou de rede) sobre o qual será computada a restrição espacial. Este objeto cadastral será indicado pelo qualificativo `ON MAP`, que pode ser omitido, no caso de haver apenas uma geometria presente para o geo-objeto.
4. A cláusula `WHERE` pode conter as restrições topológicas `INSIDE`, `TOUCH`, `CROSS`, `OVERLAP`, `DISJOINT` e a restrição métrica `DISTANCE`.

⁴Uma possível extensão incluiria aspectos de recuperação de geo-campos baseada em conteúdo, de forma análoga aos trabalhos recentes na área de “content-based image retrieval”.

5. A formulação geral da expressão de busca em LEGAL será:

```
SELECT <valor > | <objeto> | <lista de objetos e
valores>
```

```
FROM <objeto> IN <classe> ON MAP <cadastral> | <rede>
```

```
WHERE <where_clause>
```

Tome-se o seguinte exemplo de consulta espacial em LEGAL:

- “Selecione todas as regiões da França a menos de 50 km da cidade de Toulouse”. Esta consulta está ilustrada na figura 6.1.

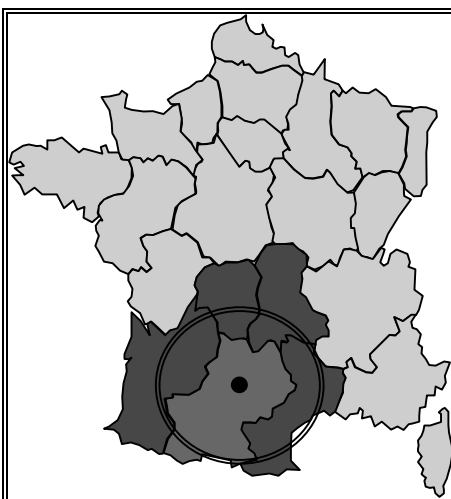


Figura 6.1 - Ilustração de consulta espacial.

A expressão de busca correspondente será:

```
SELECT regioao
FROM regioao IN Regioes_França ON MAP mapa_França,
      cidade IN Cidades_França ON MAP mapa_França
WHERE cidade.nome = "Toulouse" AND
      distance (regiao, cidade) < 50 km
```

6.5.2 RESPOSTA DE CONSULTA

Um aspecto que diferencia LEGAL da linguagem SQL tradicional é o procedimento para manipular os objetos resultantes de uma consulta. O resultado de uma consulta em SQL é uma tabela; para ter acesso aos itens resultantes de uma consulta, é utilizada a noção de “cursor”.

Este esquema de recuperação apresenta sérios inconvenientes para tratar dados espaciais. Em primeiro lugar, a manipulação posterior dos geo-objetos fica prejudicada e exige uma sintaxe não-intuitiva para a linguagem. A recuperação de respostas sob a forma de tuplas de valores também não permite tratar o caso de objetos complexos, como no caso de uma hierarquia de objetos.

Como visto no capítulo anterior, a resposta à consulta “selecione todas as reservas indígenas a menos de 30 km das represas da Eletronorte na Amazônia” é um objeto complexo composto dos pares de geo-objetos {(reserva, represa)}, passível de manipulação posterior.

Em resumo, *num bancos de dados geográficos é necessário recuperar geo-objetos e geo-campos, e não tabelas*. Para tanto, é preciso substituir o modelo de resposta de consultas implícito em SQL (“tabelas”) por um mecanismo orientado-a-objetos, em coerência com o modelo de dados proposto no capítulo anterior.

O modelo de resposta de consulta adotado em LEGAL é baseado nos conceitos do ODMG - Object Data Management Group (Catell, 1994). O modelo de dados do ODMG contempla a idéia de *coleções*, que contém um número arbitrário de elementos de um mesmo tipo. A recuperação de um elemento dentro de uma coleção é feita por posição absoluta dentro da coleção ou a partir do elemento corrente.

Seguindo esta formulação, utilizaremos instâncias da classe COLLECTION para armazenar (de forma não-persistente) as respostas de uma consulta, com a sintaxe:

```

COLLECTION <colecão>
    OF <classes_OBJETOS> | <classes_atributos>

```

Na linguagem LEGAL, a resposta a uma consulta pode devolver uma coleção de geo-objetos e de valores. Os valores correspondem a atributos de geo-objetos ou a propriedades destes objetos, computadas a partir de operadores espaciais como medidas de *distância*.

Para ilustrar a idéia, tome-se a consulta: "Recupere os postos de saúde num raio de 5 km do hospital municipal de S.J.Campos, indicando ainda sua distância ao hospital".

Esta consulta será implementada como os resultado de duas consultas. No exemplo, supomos que as classes Hospital e Postos_Saude foram definidas previamente, como especializações de OBJECT e a classe Cadastro_Urbano é uma especialização de CADASTRAL.

```

COLLECTION postos_dist (Postos_Saude, FLOAT);
VARIABLE cad_SJC Cadastro_Urbano;
cad_SJC = ( SELECT cad FROM cad IN Cadastro_Urbano
            WHERE cad.nome = "Mapa SJCampos" );
postos_dist =
    (SELECT postos, distance (postos, hospital)
     FROM postos IN Postos_Saude ON MAP cad_SJC,
          hospital IN Hospital ON MAP cad_SJC,
     WHERE hospital.nome = "Municipal"
     AND distance (postos, hospital) < 5 km );

```

6.5.3 EXEMPLOS DE CONSULTAS ESPACIAIS

Apresentamos a seguir exemplos de consultas espaciais em LEGAL. Consideremos inicialmente a consulta: “Selecione todas as fazendas da família Magalhães cruzadas pela Linha Verde que liga Salvador a Aracaju, adjacentes a esta, ou a menos de 5 km dela”. Suporemos a existência das classes Fazenda, Estrada, Rede_Estradas, Cadastro_Fazendas.

```
VARIABLE estradas_BA           Rede_Estradas;
VARIABLE cad_faz_BA            Cadastro_Fazendas;
COLLECTION fazendas_Toninho    Fazendas;

// Recupere o mapa das fazendas
cad_faz_BA= (SELECT cad FROM cad IN Cadastro_Fazendas
              WHERE cad.nome = "mapa_faz_BA");

// Recupere as estradas da Bahia
estradas_BA= (SELECT rede FROM rede IN Rede_Estradas
              WHERE rede.nome = "mapa_estr_BA");

// Agora, as fazendas de Painho !!
fazendas_Toninho= (SELECT fazenda
                   FROM fazenda IN Fazendas ON MAP cad_faz_BA,
                   estrada IN Estradas ON MAP estradas_BA,
                   WHERE (estrada.name = "Linha_Verde" AND
                           fazenda.proprietario = "Magalhães" AND
                           (estrada TOUCH fazenda OR
                            estrada CROSS fazenda OR
                            distance(fazenda, estrada) <= 5km)));
```

O próximo exemplo ilustra o caso de uma ligação entre duas tabelas. Considere a consulta: “Selecione todas as fazendas cadastradas pelo INCRA na região do Pontal do Paranapanema, com mais de 1000 ha, e cujos proprietários pagam mais de 10.000 reais de ITR”.

Neste caso, a consulta deve combinar os objetos da classe FAZENDAS com o objeto não-espacial cadastro da classe CADASTRO_INCRA, onde está arrolado o cadastro alfanumérico de propriedades. A figura 6.2 ilustra esta situação, aonde os atributos dos geo-objetos e dos objetos não-espaciais foram implementados na forma de relações. A chave de ligação entre as duas relações é o número do cadastro INCRA (chave primária da segunda relação).

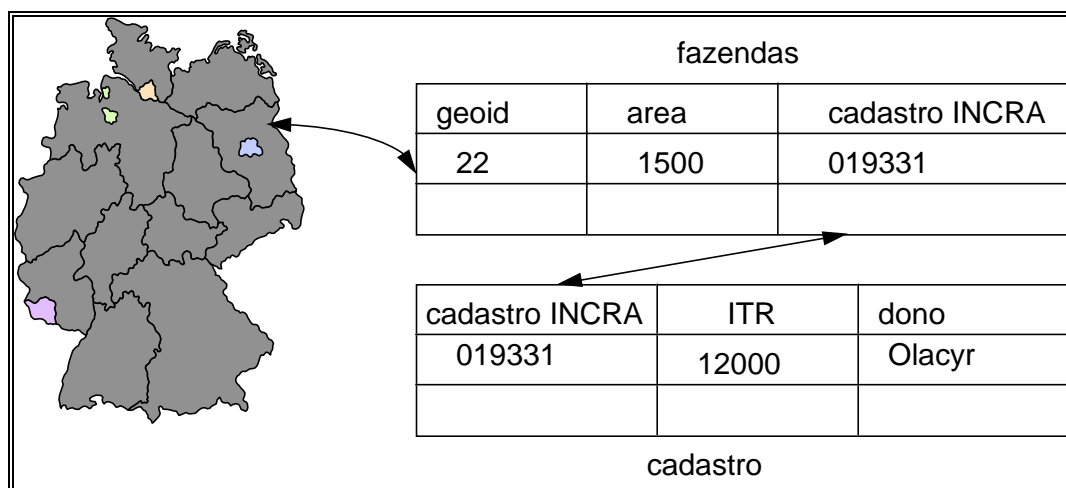


Figura 6.2 - Exemplo de consulta entre geo-objetos e objetos não-espaciais.

```
// Dados básicos existentes no Banco
VARIABLE mapa_faz_SP          Mapa_Fazendas;
VARIABLE mapa_reg_SP          Mapa_Regioes;

// Dados recuperados por selecao espacial
COLLECTION fazendas_PP        Fazenda;

// Recuperacao do mapa de fazendas
mapa_faz_SP= (SELECT mapa FROM mapa IN Mapa_Fazendas
              WHERE mapa.nome = "mapa_faz_SP");

// Recuperacao do mapa de regioes
mapa_reg_SP= (SELECT mapa FROM mapa IN Mapa_Regioes
              WHERE mapa.nome = "mapa_regioes_SP");

// Recuperacao do objeto "regiao_pontal"
// Selecao das fazendas
fazendas_PP= ( SELECT fazenda
              FROM fazenda IN Fazenda ON MAP mapa_faz_SP,
              regiao IN Regioes ON MAP mapa_reg_SP,
              cadastro IN Cadastro_INCRA
              WHERE (fazenda.num = cadastro.num AND
                    cadastro.ITR >= 10.000 AND
                    fazenda.area >= 1.000 AND
                    regiao.nome = "Pontal do Paranapanema" AND
                    fazenda INSIDE regiao));
```

6.5.4 OPERAÇÕES DE JUNÇÃO ESPACIAL

A operação usual de junção espacial compara dois conjuntos de geo-objetos, computando um predicado espacial sobre suas representações em mapas de objetos. Sua sintaxe é análoga à junção convencional, como ilustram os exemplos a seguir.

Consideramos também um caso especial de junção espacial, que ocorre quando comparamos um conjunto de geo-objetos com os valores de um geo-campo.

Junção Espacial sobre Conjuntos de Geo-Objetos

No primeiro caso, consideramos a consulta: “Mostre os aeroportos que estão dentro das reservas indígenas da Amazônia ou a menos de 50 km de uma reserva”. Suporemos a existências das classes Aeroporto, Reserva, Mapa_Reserva e Mapa_Aeroportos.

```
// definicao dos dados
VARIABLE mapa_reserv          Mapa_Reserva;
VARIABLE mapa_aerop          Mapa_Aeroportos;
// definições da coleção
COLLECTION aero_res (Aeroporto, Reserva, FLOAT);
// recuperacao dos mapas
mapa_reserv= (SELECT mapa FROM mapa IN Mapa_Reserva
              WHERE mapa.nome = "mapa_res_AMZ");
mapa_aerop=(SELECT mapa FROM mapa IN Mapa_Aeroporto
             WHERE mapa.nome = "mapa_aerop_AMZ");
// consulta sobre geo-objetos
aero_res= ( SELECT aeroporto,reserva,
            distance(aeroporto, reserva)
            FROM aeroporto IN AEROPORTO ON MAP mapa_aerop,
            reserva IN RESERVA ON MAP mapa_reservas
            WHERE (aeroporto INSIDE reserva
            OR distance (aeroporto, reserva) < 50 km));
```

Junção Espacial entre Geo-Objetos e Geo-Campo

Neste caso, o predicado espacial a ser computado sobre um mapa de geo-objetos e um geo-campo, como na consultas indicadas a seguir. Para permitir o cômputo destas operações, definimos o operador *region*, que aplicado a um geo-campo, retorna o conjunto de pontos que possui um dado valor (temático ou numérico).

1. “Dado um mapa de solos e um mapa de rios do Paraná, indique todos os rios que cruzam áreas com solos podzólicos”

```
// definições dos dados
VARIABLE mapa_solos      Mapa_Solo;
VARIABLE rede_rios      Hidrografia;
// definição da coleção
COLLECTION rios_podz     Rios;
// recuperacao dos mapas
rede_rios= (SELECT rede FROM rede IN Hidrografia
            WHERE rede.nome = "mapa_rios_PR");
mapa_solos= (SELECT mapa FROM mapa IN Mapa_Solo
            WHERE mapa.nome = "mapa_solos_PR");
// consulta sobre geo-objetos
rios_podz= (SELECT rio
            FROM rio IN RIOS ON MAP rede_rios,
            WHERE rio CROSS region(mapa_solos= "podzólico"));
```

2. “Indique todos os lotes de Cubatão que estão em áreas com alta declividade”.

```
// definicao dos dados
VARIABLE campo_decl      Declividade;
VARIABLE mapa_lotes      Mapa_Lotes;
// definições da coleção
COLLECTION lotes_risco   Lotes;
// recuperacao dos mapas
campo_decl= (SELECT gc FROM gc IN Declividade
             WHERE gc.nome = "declivid_CBT");
mapa_lotes= (SELECT mapa FROM mapa IN Mapa_Lotes
            WHERE mapa.nome = "mapa_lotes_CBT");
// juncao espacial sobre geo-objetos e geo-campo
lotes_risco= ( SELECT lotes
              FROM lotes IN LOTES ON MAP mapa_lotes
              WHERE
                lotes INSIDE region (campo_decl = "Alta")
              OR
                lotes OVERLAP region (campo_decl = "Alta"));
```

6.5.5 PREDICADO ESPACIAL PARA REDES

No caso de redes, motivados pela topologia arco-nó, é conveniente introduzir o predicado espacial LINKED_TO, que indica a ligação topológica entre dois objetos em uma rede. Trata-se de um predicado a ser utilizado na “where-clause” da operação de seleção espacial, computado a partir da sintaxe:

```
<geo_objeto> LINKED_TO <geo_objeto>.
```

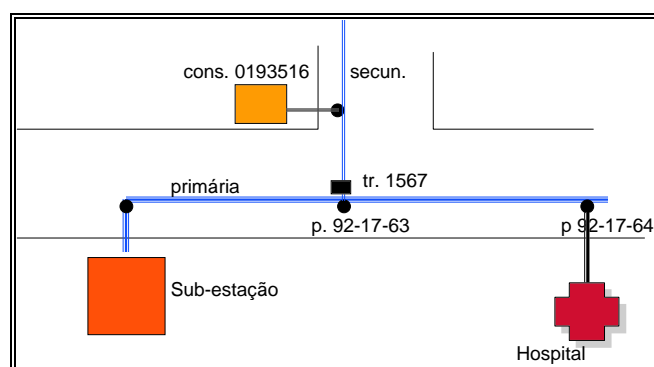


Figura 6.3 - Exemplo de rede elétrica.

Como exemplo, considere-se o caso ilustrado na figura 6.3, onde queremos saber “Quais os hospitais ligados à sub-estação do Jardim Botânico, que são da rede elétrica do bairro da Lagoa”. O programa em LEGAL seria:

```
COLLECTION cons      Consumidor;
VARIABLE rede_Rio    Rede_Eletrica;
rede_Rio= (SELECT rede FROM rede IN Rede_Eletrica
           WHERE rede.nome = "Rede_Rio");
hospitais= (SELECT cons
            FROM cons IN Consumidor ON MAP rede_Rio,
            sub_estacao IN Sub_Estacao ON MAP rede_Rio
            WHERE sub_estacao.nome = "Jardim Botânico"
            AND  cons.type = "hospital"
            AND  cons LINKED_TO sub_estacao );
```

6.6 MANIPULAÇÃO ESPACIAL EM LEGAL

Após a recuperação dos elementos de um banco de dados geográficos, LEGAL permite a aplicação de operações de geo-campos e geo-objetos, definidas no capítulo 5, e que incluem:

- Transformações pontuais entre geo-campos.
- Operações booleanas entre geo-campos.
- Operadores matemáticos pontuais, tais como funções aritméticas e trigonométricas (aplicáveis a IMAGE e a DTMs).
- Métodos de classificação contínua, utilizando a função FUZZY, que permite gerar um mapa nebuloso (variando no intervalo [0..1]).
- Operações de vizinhança.
- Operações zonais entre geo-campos.
- Operações zonais entre geo-campos e geo-objetos.

6.6.1 OPERADORES DE TRANSFORMAÇÃO SOBRE GEO-CAMPOS

Os operadores de transformação realizam o mapeamento entre as várias classes de geo-campos, a saber:

- **WEIGHT**: transforma um **THEMATIC** em um **DTM**;
- **SLICE**: transforma um **DTM** ou uma **IMAGE** em um **THEMATIC**;
- **RECLASSIFY**: transforma um **THEMATIC** em outro **THEMATIC** de classe distinta.

Como regra, estes operadores requerem que o usuário defina um mapeamento entre os geo-campos de entrada e de saída. Para tal fim, a linguagem permite ao usuário definir tabelas que descrevem os mapeamentos desejados, sob forma de uma classe **TABLE** que pode ser especializado em diferentes tipos (**WEIGHT_TABLE**, **SLICE_TABLE**, **RECLASSIFY_TABLE**).

Operação de Ponderação

Uma tabela do tipo ponderação possui dois atributos:

CLASS_IN e **FUNCTION**.

Um valor de **FUNCTION** é uma lista com um tema de entrada e um peso, denotados com a seguinte sintaxe:

`<tema_entrada> : <peso>`

Ao definir uma tabela de ponderação, não precisamos indicar a categoria de saída (que será sempre uma especialização de **DTM**) pois a operação poderá ser utilizada como passo intermediário de uma operação complexa, como mostremos posteriormente.

O exemplo a seguir ilustra uma operação de ponderação, onde um mapa de solos dá origem a um mapa de solo ponderado.

```

VARIABLE uso_solo_CE          Uso_do_Solo;
VARIABLE tab_peso             WEIGHT_TABLE;
VARIABLE solo_peso_CE        Solo_Ponderado;
tab_peso= NEW
      (CLASSE_IN = USO_SOLO,
      FUNCTION = (Le: 0.2, Li : 0.4, Aq : 0.5));
solo_pond_CE = WEIGHT (uso_solo_CE, tab_peso);

```

Operação de Reclassificação

A operação de reclassificação envolve o mapeamento entre duas especializações da classe THEMATIC e é definida por uma tabela de reclassificação, que possui três atributos: CLASS_IN, CLASS_OUT e FUNCTION. A função de reclassificação é uma lista de temas de entrada com os correspondentes temas de saída, com a sintaxe:

```
<tema_entrada>: <tema_saida>;
```

Note-se que precisamos explicitar a classe dos dados de entrada e a de saída, para que, durante a execução, possamos checar a validade sintática da operação.

No exemplo a seguir, temos um mapa de cobertura do solo na Amazônia com diferentes classes {"Floresta Densa", "Floresta Várzea", "Rebrota", "Área Desmatada", "Cerrado"}. Este mapa temático será reclassificado para um novo mapa apenas com as classes {"Floresta", "Desmatamento", "Cerrado"}. Supomos a existência de duas classes: VEGETACAO e DESMATAMENTO.

```
VARIABLE cobertura_vegetal Vegetacao;
VARIABLE desmatamento          Desmatamento;
VARIABLE tab_recl               RECLASSIFY_TABLE;
tab_recl= NEW (CLASS_IN = VEGETACAO,
              CLASS_OUT = DESMATAMENTO,
              FUNCTION = ("Floresta Densa" : "Floresta",
                          "Floresta Varzea": "Floresta",
                          "Rebrota" : "Desmatamento",
                          "Area Desmatada" : "Desmatamento",
                          "Cerrado" : "Cerrado"));
// recuperação dos dados
cob_vegetal= (SELECT gc FROM gc IN Vegetacao
              WHERE gc.nome = "mapa_Amapa");
desmatamento= NEW (nome = "desmat_Amapa")
              REPRESENTED_BY RASTER
              (RES_X = 100 m, RES_Y = 100 m);
desmatamento= RECLASSIFY (cobertura, tab_recl);
```

6.6.2 OPERAÇÕES BOOLEANAS SOBRE GEO-CAMPOS

O comando `SWITCH` permite aplicar *operações booleanas* a todos os tipos de geo-campos. É necessário especificar um conjunto de condições a ser satisfeitas para cada classe de saída, conforme o exemplo abaixo, onde um mapa de Aptidão de Solos é calculado, baseado na média de chuvas, topografia e tipo de solo.

```
VARIABLE campo_solos Solo;
VARIABLE campo_topo Topografia;
VARIABLE campo_chuva Chuva;
CREATE Aptidao (ATTRIBUTE nome STRING)
  IS_A THEMATIC (THEMES = {"Boa", "Media", "Baixa"});
VARIABLE campo_aptidao Aptidao;
campo_topo= (SELECT gc FROM gc IN Topografia
             WHERE gc.nome = "Top92");
campo_chuva= (SELECT gc FROM gc IN Chuva
             WHERE gc.nome = "chuva92");
campo_solos= (SELECT gc FROM gc IN Solo
             WHERE gc.nome = "solos94");
campo_aptidao= NEW (nome = "Aptidao_94")
  REPRESENTED_BY RASTER(RES_X= 100 m, RES_Y= 100 m);
campo_aptidao= SWITCH
  { "Boa" :      mapa_solo.THEME = "Le"
    AND mapa_chuva >= 1000
    AND mapa_topo <= 1500;
    "Media" :   mapa_solo.THEME = "Aq"
    AND mapa_chuva >= 600
    AND mapa_topo <= 1000;
    "Baixa"   : OTHERWISE;
  }
```

6.6.3 OPERAÇÕES MATEMÁTICAS

As operações matemáticas sobre geo-campos que são especializações de DTM e IMAGE, incluem:

- operações aritméticas: soma (+), subtração (-), multiplicação (*) e divisão (/);
- funções matemáticas: seno (sin), cosseno (cos), tangente (tan), arco tangente (atan), logaritmo (log), exponencial (exp), raiz quadrada (sqrt);
- relações: menor que (<), maior que (>), menor ou igual (<=), maior ou igual (>=), igual (==), diferente (!=).

Como exemplo de operador matemático, considere a *equação universal de perda de solo*, que é utilizada em Ciências do Solo.

VARIABLE	precip_anual	Precipitacao;
VARIABLE	erodibilidade	Erosao;
VARIABLE	declividade	Altimetria;
VARIABLE	exposicao	Altimetria;
VARIABLE	cobertura_solo	Solo;
VARIABLE	indice_protecao	Indice;
	perda_solo: = precip_anual x erodibilidade x	
	declividade x exposicao x cobertura_solo x	
	indice_protecao;	

6.6.4 OPERADORES LOCAIS

Os operadores locais básicos incluem: média, desvio padrão, soma, máximo, mínimo e diversidade, calculados a partir de uma vizinhança em torno de cada ponto.

Para definir os operadores locais é necessário inicialmente definir o conceito de vizinhança. Uma *vizinhança* em LEGAL pode ser definida como um *retângulo* envolvendo cada ponto, onde devem ser dadas as dimensões horizontal e vertical ou como um *círculo* ao redor de cada ponto, aonde o raio (nas dimensões do geo-campo) deve ser fornecido. Assim, a sintaxe geral dos operadores locais é

```
operador_local (geo-campo, LOCALREGION)
```

onde a definição LOCALREGION pode ser implementada como um retângulo ou um círculo. No primeiro caso, teremos:

```
operador_local (geo-campo, RETANGLE, <dim_h>, <dim_v>)
```

e no segundo,

```
operador_local (geo-campo, CIRCLE, <raio>)
```

Os operadores locais incluem:

- LOCAL_SUM(*geo-campo*, LOCALREGION): calcula a soma local dos pontos do *geo-campo* para uma vizinhança.
- LOCAL_MEAN (*geo-campo*, LOCALREGION): calcula a média local dos pontos para uma vizinhança especificada.
- LOCAL_MAX(*geo-campo*, LOCALREGION): determina o máximo local dos pontos do *geo-campo*, dada uma vizinhança.
- LOCAL_MIN (*geo-campo*, LOCALREGION): determina o mínimo local dos pontos do *geo-campo*, dada uma vizinhança.
- LOCAL_STDEV (*geo-campo*, LOCALREGION): calcula o desvio padrão dos pontos do *geo-campo*, dada uma vizinhança.

- LOCAL_VARIETY (geo-campo, LOCALREGION): determina o número de valores distintos do geo-campo, para uma vizinhança em torno de cada ponto.

Outra importante classe de operações inclui os *filtros*. Aplicáveis a geo-campos do tipo IMAGEM e MNT, os filtros computam uma equação local, calculada a partir de uma máscara em torno de cada ponto. Para tornar mais flexível o uso destas operações, definimos a existência de uma máscara, através da seguinte sintaxe:

```
VARIABLE <nome da mascara> MASK;
<nome_da_mascara>= NEW
( DIM_X = <dimensao_horizontal>,
  DIM_Y = <dimensao_vertical>,
  COEF = <lista de coeficientes>);
```

A lista dos pesos deve ser fornecida a partir do ponto superior esquerdo, da esquerda para a direita. No exemplo a seguir, definimos uma máscara para um filtro de realce de imagem: a imagem é subtraída de uma média local, aonde o ponto central tem um peso maior. Trata-se de um processo conhecido como “unsharp masking” com “addback”.

A partir da definição de uma máscara, podemos aplicar um filtro à uma IMAGEM (ou a um MNT) utilizando a sintaxe:

```
<geo-campo>= FILTER (geo_campo, MASK);
```

como mostra o exemplo a seguir.

```

VARIABLE          add_back          MASK;
VARIABLE  sjc_spotpan, sjc_realce  Spot_PAN;
add_back= NEW (DIM_X = 3, DIM_Y = 3,
              COEF = (-1, -2, -1, -2, 20, -2,
                    -1, -2, -1));
sjc_spotpan= (SELECT imagem FROM imagem IN Spot_PAN
             WHERE imagem.nome = "sjc_spotpan");
sjc_realce= NEW (nome= "sjc_realce")
             REPRESENTED_BY RASTER
             (RES_X = 10, RES_Y = 10);
sjc_realce= FILTER(sjc_spotpan, add_back);

```

Outros operadores importantes incluem:

- **REFINE**: obter um DTM de maior densidade do que o existente, com diferentes técnicas de interpolação (linear, superfícies quadráticas ou de quinto grau). Sua sintaxe é:

```

<mnt saida>= REFINE (<mnt_entrada>, <processo>);
<processo>= LINEAR | BICUBIC | QUINTIC

```

- **SLOPE, ASPECT** : calculam as derivadas locais de uma superfície e obtêm, como resultado, o módulo (declividade) e a orientação (exposição de vertentes), com a sintaxe:

```

<mnt saida>= SLOPE (<mnt entrada>);
<mnt saida>= ASPECT (<mnt entrada>);

```

6.6.5 OPERAÇÕES ZONAIS

No caso de operadores zonais, precisamos indicar o campo aonde será calculado o operador e a restrição de cálculo (que pode ser um mapa temático ou um geo-objeto).

Os operadores zonais tem a seguinte sintaxe geral:

```
<campo_zonal>= op_zonal(<geo_campo>, <restricao>)
```

aonde

```
<restricao>= <geo_campo> | <geo_objeto>
```

As operações zonais incluem:

- ZONAL_MAX: determina o valor máximo do campo de entrada para cada região de restrição.
- ZONAL_MIN: determina o valor mínimo do campo de entrada para cada região de restrição.
- ZONAL_MEAN: determina o valor médio do campo de entrada para cada região de restrição.

Outro operador importante é:

- ZONAL_STATS: produz uma estatística dos valores para cada região. A saída é um relatório , indicando para cada região zonal, os valores máximo, médio e mínimo.

6.6.6 OPERAÇÕES DE CLASSIFICAÇÃO CONTÍNUA

O uso de técnicas de classificação contínua busca utilizar as noções de conjuntos nebulosos (“fuzzy”), para substituir os processos tradicionais de geração de mapas.

Este aspecto é particularmente evidente no manuseio de mapas temáticos para análises de meio-ambiente. Uma fronteira arbitrária, definida precisamente por uma linha, entre dois tipos de solo, representa erradamente o que é, na realidade, uma variação contínua (Burrough, 1986). Quando realizamos operações de superposição entre mapas temáticos, o erro inerente à divisão arbitrária dos mapas em áreas estanques é propagado.

Estudos realizados pelo Centro Nacional de Pesquisa em Solos da EMBRAPA evidenciaram que os processos tradicionais de análise geográfica (discretizar variáveis para posteriormente combiná-las) engendram uma grande perda de sensibilidade no resultado final.

A alternativa é trabalhar sempre com dados em representação contínua, e utilizar análises quantitativas sobre mapas geográficos. Isto equivale, na prática, a trabalhar sempre com modelos numéricos de terreno para representar variáveis espaciais como solo, geomorfologia, vegetação.

Para uma discussão em maior detalhe sobre o problema, veja-se Druck e Braga (1995).

Em LEGAL, está disponível o operador de transformação de um geo-campo numérico num campo nebuloso (“fuzzy”), cujos valores variam entre $[0, \dots, 1]$. Uma aproximação de uma função de pertinência nebulosa é dada pelas equações quadráticas $\mu_L(x)$ e $\mu_U(x)$, como segue:

$$\begin{aligned} \mu_L(x) &= 1 && \text{se } x \geq \beta, \\ \mu_U(x) &= 1/[1 + \alpha(x - \beta)^2] && \text{se } x < \beta. \end{aligned}$$

$$\mu_U(x) = 1 \quad \text{se } x < \beta,$$

$$\mu_U(x) = 1/[1 + \alpha(x - \beta)^2] \quad \text{se } x \geq \beta.$$

Na equação $\mu_L(x)$, o parâmetro β indica o valor máximo, acima do qual a pertinência “fuzzy” é considerada total (isto é, igual a 1). Abaixo deste valor, a função tem uma forma quadrática, dependendo da variação do parâmetro α .

Conversamente, na equação $\mu_U(x)$, o parâmetro β indica o valor mínimo, abaixo do qual a pertinência “fuzzy” é considerada total (isto é, igual a 1). Acima deste valor, a função tem uma forma quadrática, dependendo da variação do parâmetro α . A figura 6.4 ilustra a função $\mu_L(x)$ para o caso $\alpha = 1$ e $\beta = 3$.

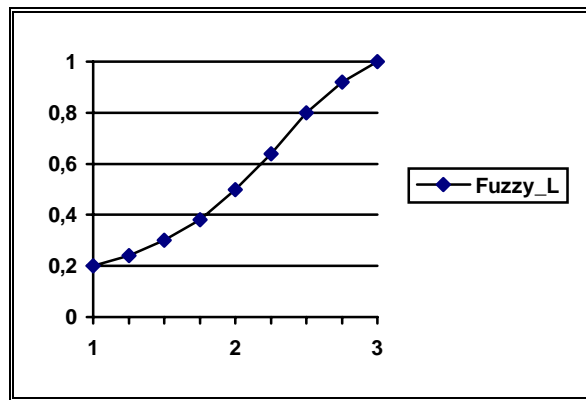


Figura 6.4 - Exemplo de função de pertinência “fuzzy”.

Em LEGAL, estão disponíveis duas operações que implementam as funções $\mu_L(x)$ e $\mu_U(x)$, com a seguinte sintaxe:

```
<mnt>= FUZZYL(<mnt_entrada>, <alfa>, <beta>);
```

```
<mnt>= FUZZYU(<mnt_entrada>, <alfa>, <beta>);
```

Em seu trabalho, Druck e Braga (1995) mostram um caso aonde se procura determinar classes de fertilidade de solos. As terras foram classificadas conforme sua exigência em termos de utilização dos insumos. A tabela 6.1 mostra os níveis de propriedades químicas para alguns parâmetros: cálcio e magnésio, fósforo e alumínio.

TABELA 6.1

<i>Propriedade</i>	<i>Classes Fertilidade</i>			
	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>
Ca+++Mg++	Ca+Mg>3	2<Ca+Mg<=3	Ca+Mg<=2	Ca+Mg<2
P	P >= 30	10 < P < 30	P < 10	P < 10
Al++	Al > 0,3	0,3 < Al < 1,5	1,5 < Al < 4	Al > 4

Nesta tabela, a classe 1 indica o solo mais fértil e 4, o menos fértil. Para determinar as funções “fuzzy” correspondentes é escolhido o parâmetro β tal que a primeira classe de níveis de fertilidade de solo tenha o valor nebuloso 1; o parâmetro α é tal que o valor da função nebulosa $\mu_A(x)$ seja igual a 0,5 quando x tiver o valor inferior da segunda classe de fertilidade. A tabela 6.2 mostra os parâmetros “fuzzy” obtidos por este critério:

TABELA 6.2 - PARAMETROS FUZZY

<i>Propriedades</i>	α	β	Função
Ca+++Mg++	1	3	$\mu_L(x)$
P	0,0025	30	$\mu_L(x)$
Al++	0,3	0,694	$\mu_U(x)$

6.6.7 CÁLCULO DE PROPRIEDADES

As operações de cálculo de propriedades objetivam obter estatísticas descritivas sobre os geo-campos e incluem:

- HISTOGRAMA: distribuição de frequência para as várias classes (ou valores) de um geo-campo e parâmetros estatísticos associados.
- TAB-CRUZ: tabulação cruzada é a distribuição de frequência mostrando as ocorrências comuns entre as classes de dois geo-campos.
- SPATIAL_AUTOC: calcula a autocorrelação espacial entre dois geo-campos.

6.7 TRANSFORMAÇÕES ENTRE GEO-CAMPOS E OBJETOS CADASTRAIS

6.7.1 TRANSFORMAÇÕES DE CAMPOS PARA OBJETOS

Como discutido no capítulo anterior, uma das principais operações de transformação entre geo-campos e geo-objetos é a operação de indentificação., que transforma um geo-campo temático num objeto cadastral e cria geo-objetos representados neste mapa, onde um dos atributos de cada geo-objeto é o valor do geo-campo.

Esta operação será realizada em dois passos. Inicialmente, cria-se um objeto cadastral que tem a mesma geometria do geo-campo temático, através da operação IDENTIFY_MAP, com a sintaxe:

```
<objeto_cadastral>= IDENTIFY_MAP (<temático>);
```

A seguir, será criado um conjunto de geo-objetos, no qual um dos atributos corresponde aos valores possíveis para o geo-campo temático, e que estará mapeado no objeto cadastral cuja geometria foi computada através da operação anterior. A operação que cria o conjunto de geo-objetos é IDENTIFY_OBJECTS, com a sintaxe:

```
<conjunto_geo-objetos>=
IDENTIFY_OBJECTS(<temático>, <atributo>)
ON MAP <cadastral>;
```

Nesta operação, o primeiro parâmetro indica o geo-campo temático a ser transformado. O segundo parâmetro indica o atributo dos geo-objetos que receberá o valor do tema do geo-campo temático (este atributo deverá ser do tipo “*string*”). A cláusula ON MAP denota o objeto cadastral criado anteriormente, cuja geometria (vetorial) é a mesma do geo-campo temático. A operação IDENTIFY_OBJECTS tem ainda a propriedade de tornar persistente uma coleção de geo-objetos.

Para ilustrar esta operação, consideremos a situação onde deseja-se transformar um geo-campo temático que contém informação de solos, num conjunto

de geo-objetos (chamadas aqui de *unidades de mapeamento*)⁵, mapeadas num objeto cadastral (*mapa de unidades*). O programa em LEGAL é mostrado, de forma simplificada, a seguir.

Inicialmente, temos as seguintes declarações de classes.

```
CREATE Campo_Solo (ATTRIBUTE nome STRING)
      IS_A THEMATIC (THEMES = "Lr", "Aq", "Cb",....);
CREATE unid_mapeamento
      (ATTRIBUTE nome          STRING,
       ATTRIBUTE tipo_solo     STRING,
       ATTRIBUTE ph            FLOAT)
      IS_A GEO-OBJECT;
CREATE mapa_unidades (ATTRIBUTE nome  STRING)
      IS_A CADASTRAL;
```

O programa em LEGAL que realiza a operação tem a forma abaixo.

```
VARIABLE  solo_MG                Campo_Solo;
VARIABLE  mapa_unid_MG           Mapa_Unidades;
COLLECTION unid_solo             Unidades_Mapeamento;

solo_MG = (SELECT gc FROM gc IN Campo_Solo
           WHERE gc.nome = "Solos_MG");

mapa_unid_MG = NEW (nome = "Mapa Unidades MG")
               REPRESENTED_BY VECTOR (SCALE = 50000);

mapa_unid_MG = IDENTIFY_MAP (solo_MG);

unid_solo = IDENTIFY_OBJECTS (solo_MG, tipo_solo)
           ON MAP mapa_unid_MG;
```

⁵Esta terminologia é empregada pelo Centro Nacional de Pesquisa em Solos da EMBRAPA.

Um segundo tipo de operações de transformação de geo-campos em objetos cadastrais é da dado pela operação INTERSECT, que realiza a geração de mapas de objetos como regiões homogêneas obtidas por superposição de um mapa de geo-campos.

Esta operação deve ser vista como uma generalização do caso anterior. Também será realizada em dois passos. No primeiro passo, cria-se um objeto cadastral, cuja geometria corresponderá à representação vetorial resultante da intersecção espacial de um conjunto de geo-campos temáticos, através da operação INTERSECT_MAP, cuja sintaxe é:

```
<cadastral> = INTERSECT_MAP (<conjunto de geo_campos>);
```

A seguir, são criados os novos geo-objetos, mapeados no objeto cadastral gerado a partir da operação anterior. A sintaxe desta operação é:

```
<conjunto_geo-objetos>=
INTERSECT_OBJECTS (<geo-campos>, <atributo>, ... )
ON MAP <cadastral>;
```

A operação INTERSECT_OBJECTS tem como parâmetros pares de geo-campos e atributos, o primeiro parâmetro de cada par indica o geo-campo temático a ser transformado. O segundo parâmetro indica o atributo dos geo-objetos que receberá o valor do tema do geo-campo temático (este atributo deverá ser do tipo “string”). A cláusula ON MAP denota o objeto cadastral criado anteriormente, cuja geometria (vetorial) é a mesma do geo-campo temático. A operação INTERSECT_OBJECTS tem ainda a propriedade de tornar persistente uma coleção de geo-objetos.

Como exemplo, tome-se um caso de zoneamento ecológico-econômico. Como parte deste tipo de estudo, uma fase inicial é obter, a partir dos condicionantes físicos, um mapa de regiões homogêneas (também chamadas *geofacies*) através da intersecção espacial dos mapas de geologia, solos e vegetação.

Inicialmente, consideremos as seguintes definições de classes em LEGAL.

```
CREATE Campo_Vegetacao (ATTRIBUTE nome STRING)
    IS_A THEMATIC (THEMES = {"Floresta Ombrofila",
        "Mangue", "Campina",....} );
CREATE Campo_Geologia (ATTRIBUTE nome STRING)
    IS_A THEMATIC (THEMES = {"Formação Cambriana",
        "Metamorficas",....} );
CREATE Campo_Solo (ATTRIBUTE nome STRING)
    IS_A THEMATIC (THEMES = "Lr", "Aq", "Cb",....);
CREATE Regioes_Homogeneas
    (ATTRIBUTE nome          STRING;
    ATTRIBUTE tipo_solo      STRING;
    ATTRIBUTE tipo_veget     STRING;
    ATTRIBUTE tipo_geol      STRING)
    IS_A GEO-OBJECT;
CREATE Mapa_Geofacies (ATTRIBUTE nome STRING)
    IS_A CADASTRAL;
```

A seguir, realizaremos as operações de intersecção.

```

VARIABLE geologia_AP      Campo_Geologia;
VARIABLE solos_AP        Campo_Solos;
VARIABLE vegetacao_AP    Campo_Vegetacao;
VARIABLE geofacies_AP    Mapa_Geofacies;
COLLECTION regioes_AP    Regioes_Homegeneas;
geofacies_AP= NEW (nome = "Mapa Geofacies Amapa",
    REPRESENTED_BY VECTOR
    (LATLONG BOX = [(02 00 00 S, 61 30 00 W),
                    (04 00 00 N, 60 00 00 W)]);
    SCALE = 1.000.000);
geo_facies_AP = INTERSECT_MAP (geologia_AP, solos_AP,
    vegetacao_AP);
regioes_AP =
    INTERSECT_OBJECTS (geologia_AP, tipo_geol,
    solos_AP, tipo_solo,
    vegetacao_AP, tipo_veget)      ON
MAP geo_facies_AP;

```

6.7.2 TRANSFORMAÇÕES DE OBJETOS EM CAMPOS

As operações que nos permitem gerar geo-campos a partir de geo-objetos incluem:

- *Reclassificação por Atributos*: geração de geo-campo a partir de um atributo de um conjunto de geo-objetos. Sua sintaxe é:

```
<geo-campo>= RECLASS_ATR (<conjunto_geo-objetos>,
    <atributo>) ON MAP <mapa_de_objetos>;
```

- *Distância*: geração de um mapa de distância a partir de um geo-objeto ou conjunto de geo-objetos, com sintaxe:

```
<geo-campo>=DISTANCE_MAP (<conjunto_geo-objetos>);
```

Para ilustrar estas operações, consideremos o caso onde um mapa de distribuição da pobreza do Brasil será obtido a partir dos dados do IBGE, agregados por estado. Suporemos que um dos atributos é a *renda per capita*. Para isto é necessário uma composição de operações: primeiramente é gerado um modelo de terreno com a distribuição da variável *renda per capita*, que será em seguida fatiado.

```
COLLECTION estados                Estado_Brasil;
VARIABLE  mapa_estados            Mapa_IBGE;
VARIABLE  campo_renda            Campo_Renda;
VARIABLE  tab_fatia              SLICE_TABLE;
ATTRIBUTE renda_per_capita       FLOAT;
mapa_renda:= NEW (nome = "Mapa Riqueza Brasil")
    REPRESENTED_BY RASTER
    (RES_X = 1000m, RES_Y = 1000m);
tab_fatia:= NEW ( CLASS_OUT = RENDA,
    FUNCTION =
    ([0-500]           : "Miseravel",
    (500-1000]        : "Pobre",
    (1000-2000]       : "Remediado",
    >2000             : "Rico"));
mapa_renda:= SLICE
    (RECLASS_ATR (estados, renda_per_capita)
    ON MAP mapa_estados), tab_fatia);
```

6.8 MECANISMOS DE FLUXO DE CONTROLE

6.8.1 ITERAÇÃO SOBRE RESPOSTA DE CONSULTA

No modelo ODMG para coleções, adotado em LEGAL, é possível percorrer iterativamente os elementos de uma coleção. Para tal fim, é necessário definir um elemento do tipo `ITERATOR`, que mantém a posição corrente na coleção a ser percorrida, cuja estrutura é dada a seguir (Catell, 1994):

```

tipo ITERATOR <T>
propriedades:
    stable?: Boolean
    iteration_order: Enum (forward, backward)
operadores:
    next()-> element:T
    first()-> element:T
    last() -> element:T
    more?()-> element:T
    reset()
    delete()

```

Nesta estrutura, o operador `next` posiciona o iterador sobre o próximo elemento da coleção. Depois que `next` é aplicado ao último elemento da coleção, o valor do iterador será `nil`. A chamada inicial a `next` retorna o primeiro elemento da coleção, se o parâmetro `iteration_order` vale `forward`, e o último elemento, se o parâmetro `iteration_order` vale `backward`.

Em LEGAL, o tipo `ITERATOR` faz parte dos tipos básicos da linguagem. A associação entre um elemento do tipo `ITERATOR` e uma coleção é feita a partir da definição de uma instância desta classe:

```
ITERATOR <nome_iterador> OVER <coleção>;
```

O percorrimento sobre uma coleção será controlado por mecanismos do tipo `DO...UNTIL` e `WHILE`.

Para ilustrar este mecanismo, considere a operação “selecione todas as cidades do Ceará a menos de 50 km de açudes com capacidade de mais de 50.000 m³ de água”.

```
// Definicoes dos elementos do Banco
VARIABLE mapa_mun_CE          Mapa_Municipal;
INSTANCE mapa_acudes_CE      Mapa_Acudes;
// definições da coleção e do iterador
COLLECTION cid_acu_CE        (Cidade, Acude);
ITERATOR  elem              OVER   cid_acu_CE;
// recuperacao dos mapas
mapa_mun_CE= (SELECT map FROM map IN Mapa_Municipal
              WHERE map.nome = "mapa_Ceara");
mapa_acude_CE= (SELECT map FROM map IN Mapa_Acudes
                WHERE map.nome = "mapa_Acudes");
// consulta sobre geo-objetos
cid_acu_CE = (SELECT cidade, acude
              FROM cidade IN Cidade ON MAP mapa_mun_CE
               acude IN  Acude ON MAP mapa_acudes_CE
              WHERE acude.capacidade > 50.000
                 AND DISTANCE (cidade, acude) < 50 km);
// percorre a coleção
WHILE MORE (elem)
BEGIN
    //processamento da colecao (nao mostrado)
    elem= NEXT (elem);
END
```

6.8.2 CONTROLE DE LAÇOS E TESTE DE CONDIÇÕES

As operações de manipulação requerem que se disponha de meios para estabelecer um fluxo de controle (interação e testes). Em LEGAL, utilizamos os comandos `FOR..TO..STEP` e `WHILE` para estabelecer os laços e o comando `IF...THEN...ELSE` para testar condições.

Como exemplo, tome-se o caso de cálculo do índice de vegetação normalizado (IVDN) calculado a partir das bandas 1 e 2 do sensor AVHRR do satélite NOAA. Como o índice pretende medir o “stress” hídrico da vegetação, ele precisa ser computado a partir de localizações sem a presença de nuvens. Assim, para o cálculo deste índice, tomamos uma sequência de imagens (pelo menos quinze dias). Se o ponto não estiver coberto por nuvens, calculamos o índice; senão, tenta-se a próxima imagem. O que se espera é que cada ponto esteja livre de nuvens pelo menos uma vez a cada quinze dias.

O programa a seguir supõe ainda a existência de uma representação matricial para os geo-campos das classes `Imagem_NOAA` e `Indice_Vegetacao`.

```
VARIABLE   ivdn                Indice_Vegetacao;
COLLECTION avhrr1, avhrr2      Imagem_NOAA ;
ITERATOR   b1      OVER       avhrr1;
ITERATOR   b2      OVER       avhrr2;
avhrr1= (SELECT imagem FROM imagem IN Imagem_NOAA
        WHERE imagem.banda = 1
        AND imagem.mes = 9
        AND imagem.quinzena = 1);
avhrr2= (SELECT imagem FROM imagem IN Imagem_NOAA
        WHERE imagem.banda = 2
        AND imagem.mes = 9
        AND imagem.quinzena = 1);
ivdn = NEW
      (nome= "IVDN Set95 Quinz1",
       mes = "Setembro", quinzena=1)
      REPRESENTED_BY RASTER
      (RES_X= 1000, RES_Y =1000);

WHILE (MORE (b1))
BEGIN
  IF (VALUE(b1) > 200)
  THEN
    IF (ivdn = 0)
    THEN
      ivdn = (VALUE(b2) - VALUE(b1))
              /((VALUE(b2) + VALUE(b1)));
    b1 = NEXT (b1);
    b2 = NEXT (b2);
END
```

O programa acima merece alguns comentários:

1. A recuperação das coleções `avhrr1` e `avhrr2` dá origem a dois conjuntos de geo-campos no formato matricial. Implicitamente, o programa supõe que se dispõe de uma imagem por dia, para todos os dias da quinzena.
2. A variável `ivdn` irá conter o resultado final da operação, na forma de uma matriz de resolução 1000 x 1000 metros (a mesma das imagens AVHRR). Ao ser criada, ela será inicializada com todos os valores em zero.
3. A interação sobre as coleções `avhrr1` e `avhrr2` é feita através dos ponteiros `b1` e `b2`. O operador `VALUE (b1)` devolve o valor do geo-campo apontado pelo ponteiro.
4. A operação é feita de forma implícita sobre as representações matriciais dos geo-campos. Assim, a operação

$$\text{ivdn} = (\text{VALUE}(\text{b2}) - \text{VALUE}(\text{b1})) \\ / (\text{VALUE}(\text{b2}) + \text{VALUE}(\text{b1}));$$

é computada sobre todos os elementos da matriz `ivdn`.

6.8.3 MECANISMOS DE INTEGRAÇÃO TEMPORAL

Os métodos de análise discutidos nas seções anteriores consistem em operações que computam relacionamentos entre objetos e geo-campos, tomados como visões estáticas da realidade geográfica. Deste modo, tais métodos são mais adequados a descrever *padrões estáticos* do que a representar *processos de evolução dinâmica*.

No entanto, a realidade geográfica está em constante evolução, sendo a ação humana um dos principais fatores a engendrar transformações duradouras do espaço. Como afirma Gregory (1992): “é imprescindível que a investigação dos processos seja feita de forma a auxiliar a compreensão da dinâmica atual do ambiente, da maneira de como funcionou no passado e de como poderá operar no futuro.”

Para que o SIG possa funcionar como apoio à modelagem de processos, será preciso dispor de modelos de simulação numérica que descrevam adequadamente os processos a ser estudados (com o possível uso de técnicas de estatística espacial) e integrar estas técnicas com as ferramentas de manuseio, recuperação e apresentação de dados espaciais.

A integração de modelos de simulação e de ferramentas de estatística espacial é vista como uma necessidade fundamental para suplantar as limitações dos métodos de análise da atual geração de SIGs. Para uma discussão em maior detalhe sobre estas questões, veja-se Goodchild et al. (1992), Goodchild et al. (1993), Bregt (1993) e Kemp (1993).

Um problema importante em modelagem ambiental é poder realizar uma integração temporal de geo-campos que expressem fenômenos naturais. Esta necessidade é particularmente premente em problemas de modelagem ambiental. Assim, precisamos incluir na sintaxe de LEGAL provisão para processos de integração.

Para representar o processo de integração, lançamos mão de mecanismos do tipo FOR . . TO . . STEP.

Para analisar o caso de uma função de integração, considere uma situação onde se quer estimar a aptidão agrícola futura do solo, considerando a manutenção

de uma cultura anual. Este é um problema clássico em análise de áreas agrícolas (Burrough, 1986). Possuímos os dados para a situação atual, e desejamos construir *cenários preditivos* para tentar orientar o uso e a ocupação do solo.

Baseado em Burrough (1986), considera-se que a perda de qualidade de solo provocada pela erosão é o principal fator de degradação dos solos e consequente perda de aptidão agrícola. Para estimar esta perda, pode-se utilizar modelos empíricos, como a *equação universal de perda de solos*, que fornece a perda em profundidade do solo como função do tipo de solo, declividade e profundidade atual.

Em nosso exemplo, tomamos um caso simplificado, onde a perda de solo é função apenas da declividade, da profundidade do solo e de um coeficiente de desgaste e a adequação agrícola é computada baseada apenas na profundidade do solo e declividade.

```

// Definicao de variaveis
VARIABLE   prof_solo       Prof_Solo;
VARIABLE   declividade     Declividade;
VARIABLE   perda_solo     Perda_Solo;
VARIABLE   aptidao_hoje   Aptidao;
VARIABLE   aptidao_fut    Aptidao;
// Recuperacao de dados
prof_solo:= (SELECT gc FROM Prof_Solo
             WHERE gc.nome = "prof_solo_Quixada");
declividade:= (SELECT gc FROM Declividade
              WHERE gc.nome = "declive_Quixada");
// Instanciacao dos novos geo-campos
perda_solo:= NEW (nome="perda_solo")
              REPRESENTED_BY RASTER (RES_X=100, RES_Y=100);
aptidao_hoje:= NEW (NOME="aptidao_Quixada_1995"
                  REPRESENTED_BY RASTER (RES_X=100, RES_Y=100);
aptidao_fut:= NEW (NOME="aptidao_Quixada_2000",
                  REPRESENTED_BY RASTER (RES_X=100, RES_Y=100);
// Calcule a aptidao hoje
aptidao_hoje:= SWITCH (
    "Boa": prof_solo > 0.30 m E
           declividade < 10% ;
    "Ruim": OTHERWISE);
// Calcule a profundidade do solo
// daqui a 5 anos
FOR I = 1 TO 5 STEP 1
BEGIN
    perda_solo:= 0.05*declividade*prof_solo;
    prof_solo:= prof_solo - perda_solo;
END
// Calcule a aptidao estimada futura
aptidao_fut:= SWITCH (
    "Boa": prof_solo > 0.30 m E
           declividade < 10% ;
    "Ruim": OTHERWISE);

```

6.9 INCLUSÃO DE FUNÇÕES EXTERNAS

É rigorosamente impossível prever todo o conjunto de funções a ser utilizado em análise espacial. Deste modo, LEGAL deve embutir previsão para a chamada de funções externas, desenvolvidas pelo usuário e que devem ser ligadas ao sistema.

Por coerência com o paradigma de implementação a ser utilizado (descrito nos próximos capítulos), sugerimos que LEGAL disponha de um *arquivo de descrição*, aonde o desenvolvedor da nova função possa “registrar” sua existência e um *arquivo de configuração*, que relacione o comando em LEGAL com o correspondente programa executável. O arquivo de descrição teria (numa primeira aproximação) uma sintaxe do tipo:

```
COMANDO ,
PARAMETRO = VALOR
```

Já o arquivo de configuração deveria incluir o relacionamento entre o comando em LEGAL e o programa externo a ser chamado, como:

```
COMANDO ,
programa = <nome do executavel>;
```

Como exemplo, considere-se a existência de uma função de segmentação de imagens, não incluída em LEGAL mas a ser chamada externamente. Sua chamada poderia ser como a mostrada a seguir.

```
EXTERNAL SEGMENTA (BANCO = "Amazonia",
                   PROJETO = "Sao Felix do Xingu",
                   NUMBANDAS = 3, IMAGEM_1="TM3",
                   IMAGEM_2="TM4", IMAGEM_3="TM5",
                   LIMIAR_1=10, LIMIAR_2=8);
```

6.10 LINGUAGEM DE APRESENTAÇÃO

Muitos trabalhos na literatura indicam a importância de permitir o controle de apresentação de forma independente do resultado da operação realizada (veja-se Egenhofer, 1994 e Hemerly, 1993a). A conclusão destes estudos é que se requer uma *linguagem de apresentação* separada da linguagem de consulta.

A linguagem de apresentação apresenta as seguintes necessidades (Egenhofer, 1994; Hemerly, 1993a):

- *Associação de descritores de apresentação gráfica e pictórica aos objetos geográficos:* definição de legendas e modos de apresentação gráfica para os dados geográficos.
- *Controle dos objetos apresentados:* associação de diferentes formas de apresentação (cor, texto, legenda, escala) para diferenciar os resultados da consulta.
- *Combinação dos resultados de consulta:* instrumento bastante poderoso em interfaces de consulta, ao realizar operações de união, intersecção e diferença entre consultas subsequentes.
- *Apresentação do contexto espacial:* a resposta a uma consulta espacial nem sempre pode ser interpretada por si só. Por exemplo, a resposta à consulta "mostre a cidade de São José dos Campos" deve conter ainda a localização da cidade em relação ao mapa do estado de São Paulo.
- *Informação adicional associada:* apresentação de gráficos associados a atributos dos objetos sob forma de cartogramas, barras e discos ("pizza").

Além destes princípios, podemos adicionar o uso de princípios de apresentação que facilitem a percepção cognitiva dos resultados. Neste contexto, o trabalho de Bertin (1973) sobre "variáveis visuais" é de grande importância, bem como os preceitos expressos em Monmonier (1992).

Estes mecanismos serão descritos no que segue.

6.10.1 DESCRITORES GRÁFICOS DOS OBJETOS GEOGRÁFICOS

Para prover mecanismos para controle da visualização dos dados, a linguagem inclui o comando `DEFINE VISUAL`. Os parâmetros controlados pela *definição visual* podem variar de acordo com cada especialização:

- **DTM:** valor máximo e mínimo das isolinhas a ser plotadas, cor e estilo das isolinhas (e das isolinhas-mestras), apresentação (opcional) dos valores da grade, lista (opcional) do valores das isolinhas a ser mostrados.

```
DEFINE VISUAL FOR CLASS <mnt>
    CONTOUR_COLOUR <cor>
    CONTOUR_TEXT <YES | NO >
    PLOT_MODE < NUMBER | INTERVAL >
    CONTOUR_NUMBER <numero>
    CONTOUR_INTERVAL <intervalo>
    GRID_VALUES < YES | NO >
END
```

- **THEMATIC:** aparência das geoclasses (cor, estilo e padrão de preenchimento, espessura das linhas).

```
DEFINE VISUAL
FOR CLASS <tematico> AND THEME <tema>
    COLOUR <cor>
    PATTERN <NONE | SOLID | DOT | HATCH >
    HATCH <angulo de hachura>
    THICKNESS <espessura>
END
```

- IMAGE: tabela de cores associada à visualização

```
DEFINE VISUAL
FOR CLASS <imagem>
    COLOUR_TABLE <tab_cor>
FIM
```

- GEOOBJECT : símbolo, texto, cor, estilo de linha e padrão de preenchimento.

```
DEFINE VISUAL
FOR CLASS <objeto>
    COLOUR <cor>
    PATTERN <NONE | SOLID | DASH | HATCH >
    HATCH <angulo de hachura>
    SYMBOL <nome do simbolo>
FIM
```

Na prática, é mais conveniente construir uma interface interativa para definição dos modos de apresentação padrão associados a cada classe, pois esta escolha não é dinâmica.

6.10.2 CONTROLE DOS OBJETOS APRESENTADOS

Os princípios enunciados por Bertin (1973) contemplam um sistema de “variáveis visuais” - forma, cor, orientação, textura, valor e tamanho. Bertin argumenta que nossa percepção é realçada ao utilizarmos uma gradação contínua em apenas uma destas variáveis, e que ficamos confusos ao interpretar gráficos aonde estas variáveis se apresentam misturadas.

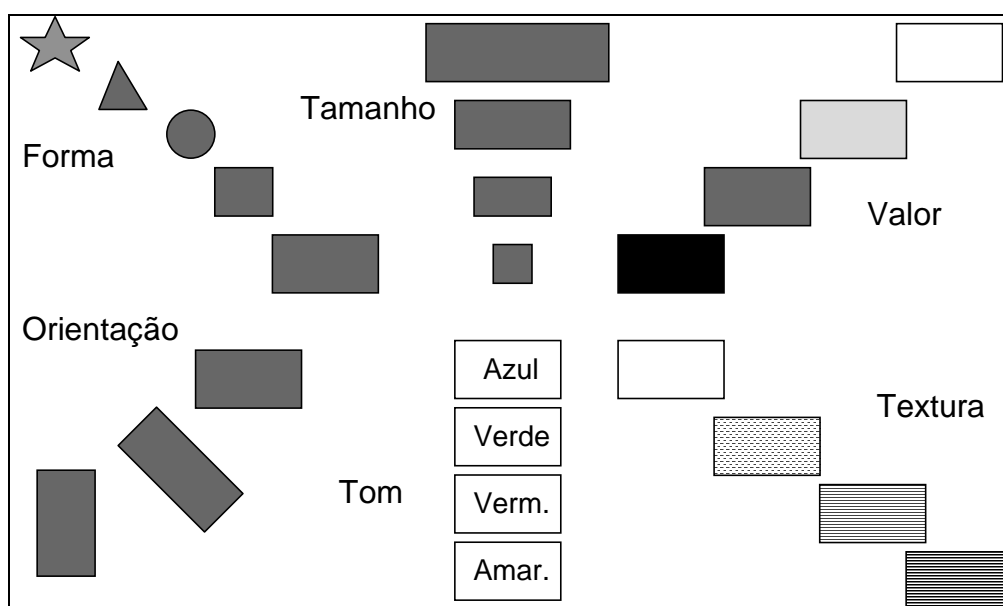


Figura 6.5 - Variáveis visuais de Bertin.

Monmonier (1992), ao discutir o problema de apresentar dados ordinais (valores numéricos referentes a geo-objetos, como população de cidades) alerta para o fato de que o agrupamento destas variáveis pode produzir sérias distorções em sua interpretação.

Com base nestes preceitos, utilizamos o comando `SHOW-AS-WHERE-GROUP BY` para controlar a visualização dos geo-objetos selecionados, com a seguinte sintaxe:

```

SHOW <colecão de geo-objetos | valores >
AS < SHADING | SYMBOL | CARTOGRAM | CHART | TEXT >
GROUP BY < VALUES | INTERVALS >
WHERE <parametros>

```

A cláusula AS controla o tipo de apresentação, que pode ser:

- SHADING: os geo-objetos selecionados são mostrados como polígonos preenchidos.
- SYMBOL: associa símbolos aos objetos selecionados.
- CARTOGRAM: apresenta os valores de atributos dos objetos sob a forma de círculos cujo área é proporcional ao valor. Pode ser modificado pela cláusula GROUP BY.
- CHART: apresenta os valores de atributos de geo-objetos sob forma de gráfico (barras, colunas, áreas, pizza, dispersão), podendo ser modificado pela cláusula GROUP BY.
- TEXT: associa um texto descritivo aos geo-objetos (por omissão, o atributo NOME associado ao objeto).

A cláusula GROUP BY indica o tipo de agrupamento a ser feito nos valores dos conjuntos de objetos, com as opções de valores ou intervalos. O agrupamento poderá ser feito em escala logarítmica ou linear.

A cláusula WHERE indica os parâmetros associados a cada tipo de apresentação, descritos a seguir.

No caso de agrupamento por valores (GROUP BY VALUES), podemos ter um agrupamento em dois, três, quatro ou cinco grupos (respectivamente HALFS, THIRDS, QUARTILES e QUINTILES), definidos pelo parâmetro:

```
VALUE_MODE= HALFS | THIRDS | QUARTILES | QUINTILES;
```

No caso de agrupamento por intervalos (`GROUP BY INTERVALS`), , podemos ter intervalos igualmente espaçados, com valores máximo e mínimo, e número de intervalos desejado, ou podemos ter intervalos definidos pelo usuário, com os seguintes parâmetros:

```
INTERVAL_MODE = EQUAL | UNEQUAL;
INTERVAL_MAX  = <valor maximo>;
INTERVAL_MIN  = <valor minimo>;
INTERVAL_COUNT = <numero de intervalos>;
INTERVAL_1    = [<valor minimo>, <valor maximo>;
INTERVAL_n    = [<valor minimo>, <valor maximo>;
```

O parâmetro `SCALE` permite a escalonamento dos valores do conjunto de objetos a ser agrupado através da opção `GROUP BY`, com a escolha entre as escalas linear e logarítmica. O uso de escala não-linear é importante para evitar distorções na apresentação, como as descritas em Monmonier (1992). A escala de agrupamento (linear ou logarítmica) será controlada pelo parâmetro `SCALE`, como segue:

```
SCALE = LINEAR | LOGARITHM;
```

No caso das opções `CARTOGRAM` e `CHART`, utilizamos por omissão o agrupamento em quartis e a escala linear.

A apresentação em polígonos sombreados (`SHADING`) pode ser controlada por parâmetros :

- O parâmetro `SHADING_MODE` controla a escolha do mecanismo de sombreamento: polígonos sombreados de uma única cor (`COLOUR`), uma palheta contínua de cores (`PALLETE`), um padrão único (`PATTERN`), ou um conjunto sequencial de hachuras (`HATCH`).

```
SHADING_MODE = < NONE | COLOUR | PALLETE |
PATTERN | HATCH > (DEFAULT = PATTERN)
```

- A cor do sombreamento, no caso de uma única cor, é controlada pelo parâmetro `SHADING_COLOUR`.

```
SHADING_COLOUR = <cor> (DEFAULT = RED)
```

- Quando da apresentação de uma variável numérica (como população) organizada por grupos, estes podem ser mostrados através de uma palheta contínua, que varia no espaço de tons (considerado como um círculo onde se percorre do azul ao púrpura, passando pelo cian, verde, amarelo, laranja e vermelho). Os pontos inicial e final da palheta são estabelecidos por `SHADING_PALLETE_MIN` e `SHADING_PALLETE_MAX`.

`SHADING_PALLETE_MIN = <cor> (DEFAULT = BLUE)`

`SHADING_PALLETE_MAX = <cor> (DEFAULT = RED)`

- O uso de um padrão de apresentação é indicado pela variável `SHADING_PATTERN`.

`SHADING_PATTERN = <padrao>`

- O uso de hachuras (`SHADING_MODE = HATCH`) é útil para demarcar uma gradação de texturas de apresentação, em conjunto com a cláusula `GROUP BY` e o parâmetro `SCALE`. Supomos uma gradação semelhante à expressa na figura 6.5.

A apresentação de símbolos pode ser controlada pelos parâmetros:

- `SYMBOL_NAME` indica o nome do símbolo.
- `SYMBOL_COLOUR` indica a cor do símbolo.
- `SYMBOL_SIZE` determina o tamanho do símbolo.
- `SYMBOL_ANGLE` denota a orientação do símbolo.

A apresentação dos dados em forma de cartogramas é controlada pelos parâmetro `CARTOGRAM_COLOUR`, que indica a cor dos círculos.

A apresentação dos dados em gráficos (CHART) tem as opções:

- O tipo de gráfico, controlado pelo parâmetro CHART_SHAPE :

```
CHART_SHAPE = < 2DAREA | 3D AREA | 2D_COLUMNS |  
3D_COLUMNS | 2D_BAR | DISPERSION | PIE_CHART >
```

- As cores associadas a cada uma das variáveis apresentadas:

```
CHART_COLOUR_1 = <cor da primeira variavel>
```

```
CHART_COLOUR_2 = <cor da segunda variavel>
```

```
CHART_COLOUR_3 = <cor da terceira variavel>
```

A apresentação de texto (TEXT) pode ser controlada a partir dos parâmetros:

- TEXT_NAME indica o nome do texto.
- TEXT_COLOUR indica a cor do texto.
- TEXT_SIZE determina o tamanho do texto.
- TEXT_ANGLE denota a orientação do texto.

6.10.3 APRESENTAÇÃO DOS RESULTADOS DE CONSULTA

O comando `SET MODE` controla a combinação dos resultados da consulta com consultas anteriores (Egenhofer, 1990), com a sintaxe:

```
SET MODE < NEW | UNION | INTERSECT | REMOVE >
```

onde:

- `NEW` indica que a tela deve ser limpa antes de mostrar o resultado da próxima consulta;
- `UNION` combina o resultado da próxima consulta com o resultado da consulta anterior;
- `INTERSECT` seleciona todos os objetos que estão presentes na tela e que pertencem ao resultado da consulta atual;
- `REMOVE` elimina o resultado da consulta atual da apresentação existente.

6.10.4 EXEMPLO DE APRESENTAÇÃO EM LEGAL

Para ilustrar a linguagem de apresentação, consideremos o caso de um estudo sobre a diversidade regional no Brasil, a partir dos dados de distribuição de miséria (pessoas cujas famílias ganham até 1/2 salário mínimo por mes), com base nos dados do IBGE, de 1985, mostrados na Tabela 6.3.

TABELA 6.3

PESSOAS COM RENDA FAMILIAR ATÉ 1/2 S.M.

<i>Região</i>	<i>Número (mil)</i>
Norte	1.325
Nordeste	25.830
Sudeste	16.076
Sul	6.551
Centro-Oeste	3.389

Fonte: IBGE (PNAD, 1985).

Inicialmente desejamos apresentar o mapa do Brasil e os dados de miséria sob forma de cartograma. Para tanto iniciamos uma nova consulta e apresentamos o mapa do Brasil (regiões).

```

COLLECTION regioes          Regioes_Brasil;
VARIABLE    mapa_reg_BR    Mapa_Brasil;

SET MODE new;

mapa_reg_BR:= (SELECT mapa FROM mapa IN Mapa_Brasil
                WHERE mapa.nome = "mapa_regioes_BR");

SHOW mapa_reg_BR

    AS SHADING

    WITH SHADING_MODE = NONE;

```

Em cima deste mapa, queremos apresentar um cartograma com a distribuição regional de pobreza. Para isto, escolhemos o modo de união dos resultados da consulta.

```
SET MODE overlay;

regioes:= (SELECT regioes
           FROM regioes IN REGIOES_BRASIL);

SHOW regioes.miseria

AS CARTOGRAM

WHERE CARTOGRAM_COLOUR = "GRAY"

AND SCALE = LINEAR;
```

O resultado da união destas duas consultas é mostrado na figura 6.6.

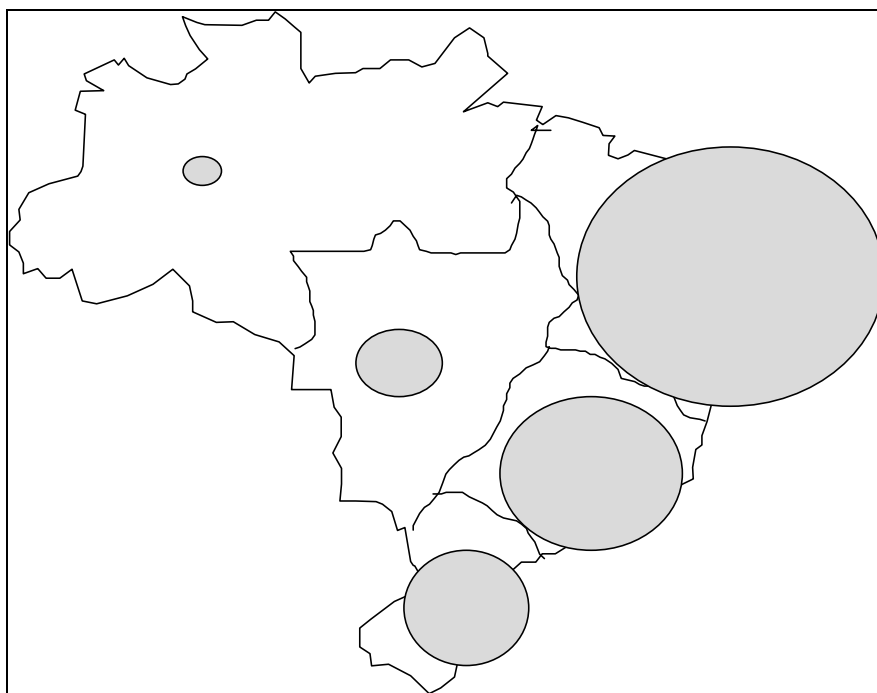


Figura 6.6 - Resultado de combinação de apresentações.

Também podemos querer mostrar estes mesmos dados sob a forma de gráfico:

```
SET MODE NEW;  
  
SHOW regioes.miseria  
  
    AS CHART  
  
    WHERE CHART_MODE = 3D_COLUMN  
  
    AND   SCALE=LINEAR;
```

O gráfico resultante é mostrado na figura 6.7.

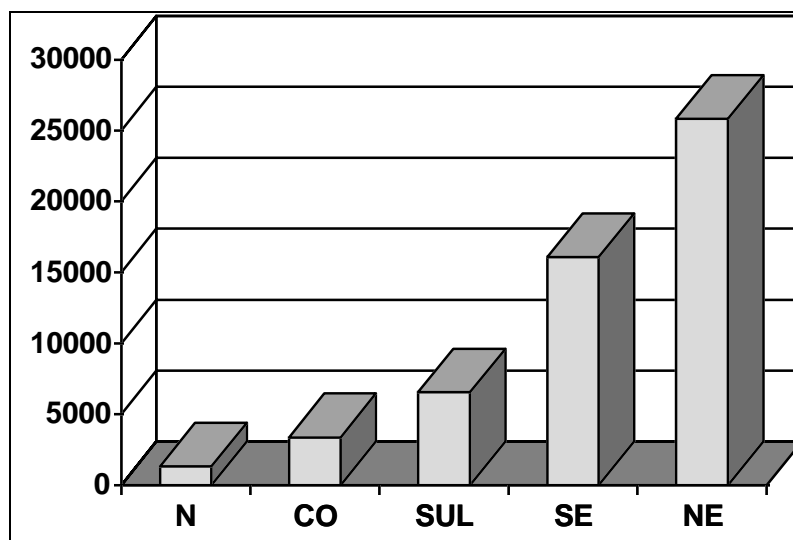


Figura 6.7 - Apresentação em forma de gráfico de barras.

6.11 BREVE ANÁLISE COMPARATIVA

Para concluir esta primeira descrição da linguagem LEGAL, é interessante compará-la com duas propostas apresentadas na literatura: SPATIAL SQL (Egenhofer, 1994) e as extensões da linguagem de consulta do SGBD O₂ (OQL) apresentadas em David e Voisard (1993).

Na definição de LEGAL, adotamos o mesmo princípio preconizado em SPATIAL SQL: a separação entre linguagem de consulta e linguagem. Consideramos, no entanto, que SPATIAL SQL ainda carrega as limitações do modelo relacional, o que torna mais complexo o processo de lidar com objetos individuais. O resultado da seleção continua sendo um tupla, e não um conjunto de valores e objetos.

Com relação às extensões de OQL apresentadas em David e Voisard (1993), temos uma situação aonde o modelo de dados utilizado estabelece uma forte ligação entre um geo-objeto e sua representação. Por contraste, o modelo de dados de LEGAL permite uma separação entre os conjuntos de geo-objetos e os mapas cadastrais que contém uma possível representação deste conjunto.

Em resumo, consideramos que LEGAL representa uma avanço com relação às alternativas de linguagem espacial disponíveis na literatura.

6.12 RESUMO DA LINGUAGEM

Apresentamos a seguir um resumo parcial da sintaxe da linguagem LEGAL, em uma notação informal semelhante à BNF. A sintaxe apresentada está baseada na implementação parcial de LEGAL no SPRING-2.0 feita por Ubirajara Freitas e João Pedro Cordeiro e no trabalho de Chu (1994).

- As palavras reservadas de LEGAL (símbolos terminais da gramática) são dados em COURIER MAÍSCULAS.
- Os símbolos entre <> são não-terminais e aqueles precedidos por @ são reconhecidos pelo analisador léxico.
- O símbolo <@nome> indica a sintaxe de identificadores, p.ex: "Uso do Solo"
- O símbolo <@num> indica o reconhecimento de um número inteiro ou na forma decimal.
- O símbolo [expressão] indica uma expressão opcional.

Definição de Classes

```

definicao_classe ::=
    CREATE <@nome_classe> (<atributos_conv>)
    IS_A <classe_modelo> | <@nome_classe> [<temas>];
<atributos_conv> ::= [<atributos_conv> ,] <atributo>
<atributo> ::= ATTRIBUTE <@nome_atrib> <classe_atrib>
<temas> ::= THEMES '= { ' <atr_tematicos> ' } '
<atr_tematicos> ::= [<atr_tematicos> ,] <@nome_tema>
<classe_modelo> ::= THEMATIC | CADASTRAL |
    GEOOBJECT | NUMERIC | IMAGE | NETWORK
<classe_atrib> ::= INT | FLOAT | STRING |
    MULTIMEDIA

```

Organização de Programa

```

<programa LEGAL> ::= { <lista_de_comandos> }
<lista_de_comandos> ::= <comando> |
    <lista_de_comandos> <comando>
<comando> ::= <declaracao> | <instanciacao> |
    <consulta> | <manipulacao> | <transformacao> |
    <apresentacao>

```

Declaração de Variáveis, Coleções e Atributos

```

<declaracao> ::= <declaracao_variavel> |
    <declaracao_colecao> | <declaracao_atributo> |
    <declaracao_tabela>
<declaracao_variavel> ::=
    VARIABLE <@nome_variavel> <@nome_classe>
<declaracao_colecao> ::=
    COLLECTION <@nome_colecao> <@nome_classe>
<declaracao_atributo> ::=
    ATTRIBUTE <@nome_atributos> <classe_atributo>
<declaracao_tabela> ::=
    TABLE <@nome_tabela> SLICE_TABLE |
    WEIGHT_TABLE | RECLASSIFICATION_TABLE

```

Instanciação

```

<instanciacao> ::= <inst_PI> | <inst_tabela>
<inst_PIs> ::=
    <@variavel_def> = NEW(<param_PI>)
    REPRESENTED_BY <geometria> (param_repres)

```

```

<geometria> ::= RASTER | VECTOR
<param_PI> ::= <param_PI> | <parametro>
<parametro> ::= <@nome_atrib> = <@valor_atrib>
<param_repres> ::=
    RESX = <@num>,
    RESY = <@num>,
    SCALE = <@num>,
    BOX= [<@lat1>, <@long1>, <@lat2>, <@long1>]
<inst_tabela> ::= <@nome_tabela>
    = NEW (<expr_cria_tabela>)
<expr_cria_tabela> ::=
    CLASS_IN = <@nome_classe>,
    TYPE = WEIGHT, <lista_de_pesos> |
    CLASS_OUT = <@nome_classe>,
    TYPE = SLICE, <lista_de_fatias> |
    CLASS_IN = <@nome_classe>,
    CLASS_OUT = <@nome_classe>,
    TYPE = RECLASSIFY, <lista_de_classes>
<lista_de_pesos> ::= <atrib_peso> |
    <lista_de_pesos> , <atrib_de_peso>
<atrib_peso> ::= <@tema> : <@num>
<lista_de_fatias> ::= <atrib_fatia> |
    <lista_de_fatias> , <atrib_fatia>
<atrib_fatia> ::= <@tema> : <@num> , <@num>
<lista_de_classes> ::= <atrib_classe> |
    <lista_de_classe> , <atrib_classe>
<atrib_classe> ::= <@tema_saida> : <@tema_entrada>

```

Consulta Espacial

```

consulta_LEGAL ::= <@nome_colecao> = <expr_consulta>
<expr_consulta> ::=
    SELECT <objetos_e_valores>
    FROM <@nome_objeto> IN <@nome_classe>
        [ON MAP <@nome_cadastral> | <@nome_rede> ]
    [<WHERE_clause>]

<objetos_e_valores> ::= <objetos> | <valores>
<objetos> ::= <objetos> | <@nome_objeto>
<valores> ::= <valores> | <@nome_atributo>

<WHERE_clause> ::= WHERE <boolean>
<boolean> ::= <bool_expr1>
    | <boolean> OR <bool_expr1>
<bool_expr1> ::= <bool_expr1> AND <bool_expr2> |
    <bool_expr2>
<bool_expr2> ::= [NOT] <bool_prim>
<bool_prim> ::= <predic_topologico> | <predic_metrico>
    | <pred_aritm> | <boolean>
<predic_topologico> ::=
    <@nome_geobjeto> <rel_topologia>
    <@nome_geobjeto> | <regiao_tematica>
<rel_topologia> ::= INSIDE | OVERLAP | DISJOINT |
    CROSS | TOUCH | LINKED_TO
<regiao_tematica> ::=
    REGION(<@nome_geocampo_tematico> == <@tema>)

```

```

<predic_metrice> ::=
    DISTANCE (<@nome_geobjeto>, <@nome_geobjeto>)
    <op_comp> <@numero>
<predic_aritm> ::= <expr_num> <oper_comp> <expr_num>
<expr_num> ::= <termo_aritm> |
    <expr_num> <oper_add> <termo_aritm>
<termo_aritm> ::= <aritml> |
    <termo_aritm> <op_mul> <aritml>
<aritml> ::= [op_add] <primario>
<primario> ::= <@num> | <objetos_e_valores>
<op_comp> ::= > | < | >= | <= | = | !=
<op_add> ::= + | -
<op_mul> ::= * | /

```

Manipulação Espacial

```

<manipulacao> ::=
    <@nome_geocampo_tematico> = <oper_tematica> |
    <@nome_geocampo_numerico> = <oper_numerica>
<oper_tematica> ::=
    <@nome_geocampo_tematico> |
    SLICE (<oper_numerica>, <@nome_tabela_fatia> ) |
    RECLASSIFY(<oper_tematica>, <@nome_tabela_reclass>) |
    SWITCH ({ <lista_de_regras> })
<lista_regras> ::= <regra> | <lista_regras>; <regra>
<regra> ::= <@tema> : <expr_bool>

```

```

<oper_numerica>:= <@numero> |
                    <@nome_geocampo_numerico> |
                    WEIGHT (<oper_tematica>, <@nome_tabela_pond>) |
                    <op_local_num> |
                    <op_zonal> |
                    FUZZYL(<@nome_geocampo_num>, <@num>, <@num>) |
                    FUZZYU(<@nome_geocampo_num>, <@num>, <@num>) |
                    REFINE (<@nome_geocampo_num>, <proc_refin>) |
                    SLOPE (<@nome_geocampo_num>) |
                    ASPECT (<@nome_geocampo_num>) |
                    <funcao_matematica> (<oper_numerica>) |
                    <oper_numerica> |
                    <oper_numerica> <op_add> <oper_numerica> |
                    <oper_numerica> <op_mul> <oper_numerica> |
                    - <oper_numerica>

<funcao_matematica> := SIN | COS | ATAN | LOG |
                    LOG10 | EXP | SQRT | INT | ABS

<expr_bool> ::= <expr_bool> && <expr_bool> |
                <expr_bool> || <expr_bool> |
                [<expr_bool>] |
                <@nome_geocampo_tematico>.THEME == <@tema> |
                <@nome_geocampo_tematico>.THEME != <@tema> |
                <expr_numerica> <op_comp> <expr_numerica>

```

```
<op_local_num> ::=
    LOCAL_SUM (<oper_numerica>, <viz>) |
    LOCAL_MEAN (<oper_numerica>, <viz>) |
    LOCAL_MAX(<oper_numerica>, <viz>) |
    LOCAL_MIN (<oper_numerica>, <viz>) |
    LOCAL_STDEV (<oper_numerica>, <viz>)
<viz> ::= <retangulo> | <circulo>
<retangulo> ::= RECTANGLE, <@num>, <@num>
<circulo> ::= CIRCLE, <@num>
<op_zonal> ::=
    ZONAL_SUM (<oper_numerica>, <@tematico>) |
    ZONAL_MEAN (<oper_numerica>, <@tematico>) |
    ZONAL_MAX(<oper_numerica>, <@tematico>) |
    ZONAL_MIN (<oper_numerica>, <@tematico>) |
    ZONAL_STDEV (<oper_numerica>, <@tematico>)
```

Transformação entre Campos e Objetos

```

<transformacao> ::= <campos_em_objetos> |
                   <objetos_em_campos>

<campos_em_objetos> ::= <identificacao> |
                        <interseccao>

<objetos_em_campos> ::= <mapa_distancia> |
                        <recl_atributos>

<identificacao> ::= <id_cadastral> | <id_objetos>

<id_cadastral> ::= <@nome_cadastral> =
IDENTIFY_MAP (<@nome_tematico>)

<id_objetos> ::= <@nome_colecao> =
IDENTIFY_OBJECTS (<par_tema_atr>)
ON MAP <@nome_cadastral>

<interseccao> ::= <int_cadastral> | <int_objetos>

<int_cadastral> ::= <@nome_cadastral> =
INTERSECT_MAP (<lista_temat>)

<lista_temat> ::= <@nome_tematico> | <lista_temat>

<int_objetos> ::= <@nome_colecao> =
IDENTIFY_OBJECTS (<lista_pares>)
ON MAP <@nome_cadastral>

<lista_pares> ::= <par_tema_atr> | <lista_pares>

<par_tema_atr> ::= (<@nome_tematico>, <@nome_atributo>)

<mapa_distancia> ::= <@nome_numerico> =
DISTANCE_MAP (<geo_objetos>)

<geo_objetos> ::= <@nome_geoobjeto> | <geo_objetos>

<recl_atributo> ::= <@nome_geocampo> =
RECL_ATTR (<geo_objetos>, <@nome_atributo>)
ON MAP <@nome_cadastral>

```

Apresentação

<apresentacao> ::= <op_mostrar> | <controle_tela>

<controle_tela> ::= SET MODE <modo_vis>

<modo_vis> ::= NEW | UNION | INTERSECT | REMOVE

<op_mostrar> ::= SHOW <objetos_e_valores>

AS <modo_apresent>

GROUP_BY <modo_agrupar>

WHERE <lista_param_apresen>

<modo_apresent> ::= SHADING | SYMBOL | CARTOGRAM |
CHART | TEXT

<modo_agrupar> ::= VALUES | INTERVAL

<lista_param_apresen> ::= <param_apres> |
<lista_param_apresen>

<param_apres> ::=

<grup_valores> |

<grup_intervalos> |

<escala> |

<sombreamento> |

<simbolos> |

<cartograma> |

<grafico> |

<texto>

```

<grup_valores>::= VALUE_MODE = <param_valor>
<param_valor>::= HALFS | THIRDS | QUARTILES |
    QUINTILES;

<grup_intervalos>::=
    INTERVAL_MODE = <modo_int> |
    INTERVAL_MAX   = <@num> |
    INTERVAL_MIN   = <@num> |
    INTERVAL_COUNT = <@num> |
    <lista_intervalos>
<lista_intervalos>::= <intervalo> |
    <lista_intervalos>
<intervalo>::= INTERVAL.<@num> = <@num>, <@num>

<escala>::= SCALE = <param_escala>
<param_escala>::= LINEAR | LOG

<sombreamento>::=
    SHADING_MODE = <modo_sombr> |
    SHADING_COLOUR = <@nome_cor> |
    SHADING_PALLETE_MIN = <@nome_cor> |
    SHADING_PALLETE_MAX = <@nome_cor> |
    SHADING_PATTERN = <@nome_padrao>
<modo_sombr>::= NONE | COLOUR | PALLETE |
    PATTERN | HATCH

```

```
<simbolos> ::=
```

```
    SYMBOL_NAME = <@nome_simbolo> |
```

```
    SYMBOL_COLOUR = <@nome_cor> |
```

```
    SYMBOL_SIZE = <@num> |
```

```
    SYMBOL_ANGLE = <@num>
```

```
<cartograma> ::=
```

```
    CARTOGRAM_COLOUR = <@nome_cor>
```

```
<grafico> ::=
```

```
    CHART_SHAPE = <forma_graf>
```

```
<forma_graf> ::= 2D_AREA | 3D_AREA | 2D_COLUMNS |
```

```
    3D_COLUMNS | 2D_BAR | DISPERSION | PIE_CHART
```