UERJ

Dissertação de Mestrado em Engenharia de Computação
Área de Concentração em Geomática

APLICAÇÃO DA GEOMÁTICA NO AUXÍLIO DAS MATRÍCULAS EM ESCOLAS DA REDE PÚBLICA

Autor: Janaína Eliza Fadel

Orientador: DSc. Orlando Bernardo Filho

Programa de Pós Graduação em Engenharia de Computação
Área de Concentração em Geomática

Março - 2003

Faculdade de Engenharia

APLICAÇÃO DA GEOMÁTICA NO AUXÍLIO DAS MATRÍCULAS EM ESCOLAS DA REDE PÚBLICA

Janaína Eliza Fadel

Dissertação submetida ao corpo docente da Faculdade de Engenharia da Universidade do Estado do Rio de Janeiro – UERJ, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Computação – Área de Concentração Geomática.

Orientador: Prof. Orlando Bernardo Filho, D.Sc.

Programa de Pós Graduação em Engenharia de Computação
Área de Concentração em Geomática

RIO DE JANEIRO, RJ - BRASIL

MARÇO DE 2003

FADEL, JANAÍNA ELIZA

Aplicação da Geomática no auxílio das ma trículas em escolas da rede pública [Rio de Ja neiro] 2003

XI, 173 p. 29,7 cm (FEN/UERJ, M.Sc., Engenharia da Computação – Área de Concen tração Geomática, 2003)

Tese - Universidade do Estado do Rio de Janeiro - UERJ

1. Planejamento Educacional

I. FEN/UERJ II. Título (série)

FOLHA DE JULGAMENTO

Título: Aplicação da geomática no auxílio das matrículas em escolas da rede pública

Candidato: Janaína Eliza Fadel

Programa de Pós Graduação em Engenharia de Computação

Área de Concentração em Geomática

Data da defesa: 10 de março de 2003

Aprovada por:

_____

Orlando Bernardo Filho, Dsc., UERJ

_____

João Araújo Ribeiro, Dr., UERJ

_____

Reiner Olíbano Rosas, Dsc., UFF

Dedicatória

Aos meus pais Selma e Nage
aos meus irmãos, Alexandre e Augusto
e a minha avó Almerinda
pelo carinho e apoio no decorrer
deste trabalho.

AGRADECIMENTOS

À CAPES, pela bolsa concedida durante o curso.

À Secretaria Municipal de Educação da Cidade de Nova Iguaçu, representada pela prof. Eliana Papaléo, pelo fornecimento de dados referentes às escolas.

À Fundação CIDE – Centro de Informações e Dados do Rio de Janeiro, pelo fornecimento de dados.

Ao Augusto Cesar Mazza, por ter entendido minha dedicação total à dissertação e compreendido meus momentos de crise nesta fase final do mestrado.

Ao amigo Hugo da Costa Fioravante, por algumas dicas fundamentais.

À amiga Luciana Ribeiro Cordeiro, pela revisão da tradução do texto.

Ao professor orientador, Orlando Bernardo Filho, por ter sido um ORIENTADOR presente durante toda a elaboração deste trabalho e pela revisão do mesmo.

À Maria Elisabeth Nascimento da Silva, ou simplesmente tão carinhosamente chamada Beth, por “quebrar todos os galhos” e ser mais que uma secretária do curso, uma amiga.

Ao Carlos Roberto Costa, pelo auxílio em várias horas no Laboratório da Geomática e pela amizade.

Aos demais professores e amigos do curso de Geomática, cada um ao seu jeito contribuiu para com minha formação.

Resumo da Dissertação apresentada à FEN/UERJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.).

APLICAÇÃO DA GEOMÁTICA NO AUXÍLIO DAS MATRÍCULAS EM ESCOLAS DA REDE PÚBLICA

Janaína Eliza Fadel

Março/2003

Orientador: Orlando Bernardo Filho, Dsc., UERJ.

Programa de Pós-Graduação em Engenharia de Computação - Área de Concentração Geomática – Mestrado

O objetivo desta dissertação de mestrado foi elaborar um aplicativo calcado em uma base de dados geo-referenciados, principalmente para as variáveis aluno e escola. Através destas variáveis localizadas pontualmente no sistema através de suas coordenadas *X* e *Y* (*Universal Projection Transversa of Mercator - UTM*), foi proposto um exemplo de aplicação de técnicas de Geomática. O Sistema de Matrículas da Rede Pública de Ensino sofreria uma modificação em sua estrutura. O aplicativo batizado de **MatriGeo** foi implementado para tornar o processo de matrículas mais ágil. A matrícula de um aluno (*A1*) em alguma escola (*E1*) será efetuada pela menor distância (calculada pelas coordenadas *X* e *Y*) entre estes dois pontos. Mas a efetivação da matrícula levará em conta outros fatores, não somente a localização. São eles: se a escola oferece a mesma série em que o aluno estará se inscrevendo, primeiro ou segundo segmento (no caso do primeiro segmento, ainda tem que saber se é creche ou não); se há turmas e se há professores para ministrar aulas em uma quantidade determinada de turmas.

Abstract of Dissertation presented to FEN/UERJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.).

APPLICATION OF GEOMATIC SO AS TO HELP THE REGISTRATIONS IN PUBLIC SCHOOLS

Janaína Eliza Fadel

March/2003

Advisors: Orlando Bernardo Filho

Program Computing Engineering - Geomatic

The objective of this dissertation was development of a system, using a geo-referencing database to the variables student and school. Those variables was located in points X and Y (Universal Projection Transverse of Mercator - UTM). It was proposed an example of Geomatics techniques application based on such variables. The procedure of registration in the public schools would be changed. The system developed, named by MatriGeo, was programmed to do a registration procedure faster. The registration of a student (A1), in some school, will be done according smallest distance (calculated by X and Y coordinates) between the point of the student and the point of the school. However the conclusion of the registration procedure will consider others factors, not only the location. The others factors are: 1) the school shall offer the year of the student; 2) there are classes to the student; 3) there are teachers to the class of the student.

ÍNDICE

CAPÍTULO I

Introdução

I.1 Apresentação

A motivação inicial para a elaboração da presente dissertação de mestrado, partiu da análise e discussão sobre um sistema já existente, o Rio Atlas [1] que foi desenvolvido pela Prefeitura da Cidade do Rio de Janeiro através do Instituto Municipal de Urbanismo Pereira Passos.

Nesse sistema, temos um atlas digital do município do Rio de Janeiro capaz de gerar mapas, permitindo consultas simples às informações contidas em seu banco de dados, possibilitando visualizações, manipulação, impressão e personalização de mapas e tabelas. Entretanto foi diagnosticada a escassez de algumas informações no tocante aos dados de escolas.

A partir de tal diagnóstico e da manifestação de pessoas durante um evento realizado em julho de 2000, na cidade de Florianópolis (SC) [2], aflorou-se a idéia de fazer um trabalho similar ao já mencionado sistema, levando-se em conta somente dados sobre a variável escola e, tendo como área de estudo, não o Município do Rio de Janeiro, e sim um município da Baixada Fluminense.

Para fins de estudo, escolheu-se o município de Nova Iguaçu, pelo fato de as suas administrações mais recentes terem se mostrado dispostas a empregar as tecnologias da geomática [3]. Embora tenha sido sondada a possibilidade, através de consultas, para se aplicar o projeto em outros municípios da baixada fluminense, apenas a Secretaria Municipal de Educação da Cidade de Nova Iguaçu respondeu e colocou-se à disposição para qualquer ajuda viável no desenvolvimento do projeto.

Todo início de ano, a mesma cena se repete: mães, pais e responsáveis passam a noite em frente a uma escola com a finalidade de conseguir uma vaga na rede municipal para seus filhos. As filas são quilométricas e, no final, muitos alunos ficam sem vagas.

Tendo em vista esse quadro, a elaboração de um banco de dados com informações sobre as escolas, como, por exemplo: 1) a quantidade de escolas por bairro; 2) número total de salas em cada escola; 3) infra-estrutura existente nessas escolas, possibilitará agilizar e otimizar a matrícula dos alunos.

Assim, integrando esses dados a uma base digital de um determinado município em um Sistema de Informação Geográfica (SIG) já existente, como o Arcview®, será possível fazer análises espaciais para se identificar às áreas de influência de cada escola.

A partir dos dados gerados e editados no ArcView® [4], será elaborado um aplicativo que irá auxiliar nas matrículas de alunos na rede municipal de ensino.

I.2 Objetivos

Esta dissertação de mestrado teve como finalidade criar uma base de dados geo-referenciados, assim como desenvolver um sistema para ser utilizado junto ao planejamento urbano das grandes cidades. Para atender a este fim, dentre as inúmeras aplicações possíveis, tomou-se como foco o estudo da Rede Municipal de Educação do Município de Nova Iguaçu, para assim, racionalizar o processo de matrícula dos alunos nesse município.

O sistema proposto foi batizado de **MatriGeo** para proceder com a localização de escolas. Este sistema apresenta limitações devido ao tempo disponível para seu desenvolvimento, como também, pela falta de uma equipe multidisciplinar composta de vários profissionais (engenheiros cartógrafos, profissionais da educação, assistentes sociais etc.).

Os fatores mencionados acima são fundamentais para a elaboração de um sistema mais completo (como o Rio Atlas, entre outros), onde além de serem cadastradas mais informações, seria possível adicionar maiores funcionalidades ao **MatriGeo**.

Assim sendo, esta dissertação trabalhará apenas com o objetivo de cadastro das escolas municipais (localizadas espacialmente e *linkadas* a um banco de dados), assim como a pré-matrícula de cada aluno, que será efetuada no sistema proposto levando-se

em consideração apenas a variável *"localização mais próxima"*. Tal variável refere-se à distância da residência do aluno (dado de entrada no sistema) à escola mais próxima.

Entretanto, vale ressaltar que o sistema apenas priorizará este critério levando-se em conta a inclusão do aluno no mesmo. Assim sendo, caso um aluno more próximo a uma dada escola, mas deixe para se cadastrar nos últimos dias da pré-matrícula, ou se inscreva após todas as vagas de tal escola se esgotar, sua alocação não obedecerá apenas ao critério mais próxima. Este aluno cairá num *loop* de escolas mais próximas, até ser alocado numa que realmente possua vaga.

Futuramente este sistema poderá ser remodelado com a finalidade de atender a um número maior de critérios para o cadastramento de alunos e alocação de suas vagas.

I.3 Contribuições

O sistema **MatriGeo** é fruto do estudo de outros trabalhos na mesma linha. Porém, estes são em sua maioria, relacionados apenas à localização das escolas. Leva-se em conta a avaliação da sua distribuição espacial através da *p mediana* [5,6,7], como também a capacidade de oferta de cada uma identificando áreas onde existe escassez ou excesso de vagas [5,8]. Existem trabalhos que tratam da estrutura da própria escola, com informações de planta baixa, saneamento, entre outras [9]; e outros onde o ponto fundamental é o raio de abrangência de cada escola [10,11]. O setor censitário é abordado como unidade de mapeamento em [5,6,7,8,10], enquanto que a unidade de mapeamento adotada para o sistema **MatriGeo,** publicado em [12], é o que se chama de setores de planejamento da cidade de Nova Iguaçu. Outra fonte de consulta [13] se refere a um sistema já implantado na Secretaria de Educação do Estado de Pernambuco, onde os alunos indicam três escolas de preferência para estudar (através de telefone) e os atendentes incluem o aluno naquela que possuir vaga. Também há um caso em que são levantadas as comparações da distribuição espacial de alunos da 8ª. Série do Ensino Fundamental, das escolas municipais e particulares de um determinado bairro [14].

I.4 Sumário

A presente dissertação de mestrado foi dividida em 5 (cinco) capítulos. O primeiro refere-se a esta pequena **Introdução**, onde se tem um apanhado geral sobre o trabalho.

No segundo capítulo, que diz respeito às **Metodologias Utilizadas no Projeto**, serão encontradas informações sobre Sistemas de Informação Geográfica e Projeto de Banco de Dados.

Em seguida, no terceiro capítulo, denominado **Organização do Projeto**, serão visualizadas informações gerais sobre a Cidade de Nova Iguaçu, um pequeno apanhado sobre a educação no presente município, a base de mapas digitais, bem como os recursos disponíveis do **MatriGeo**. Este é o primeiro capítulo que começa a falar diretamente sobre como foi desenvolvido o projeto.

No quarto capítulo, **Implementação do Projeto**, será apresentada toda a estrutura empregada para o desenvolvimento do projeto. Enquanto o capítulo anterior refere-se a como deve ser utilizado o sistema, este diz respeito a como foi desenvolvido.

O quinto capítulo refere-se aos **Testes e Resultados**.

Finalmente, o sexto capítulo diz respeito às **Conclusões**, onde serão descritos todos os objetivos alcançados, como também, possíveis avanços ou seqüências para melhorias futuras.

CAPÍTULO II

Metodologias Utilizadas no Projeto

II.1 Introdução

Este é um capítulo de fundamentos teóricos, no qual serão abordados sucintamente: uma descrição sobre Sistemas de Informação Geográfica e uma descrição sobre a construção de um projeto de banco dados.

II.2 Sistemas de Informação Geográfica

A tecnologia dos Sistemas de Informações Geográficas (SIGs) [15,16,17] pode ser usada para investigações científicas, administração de recurso e planejamento de desenvolvimento. Por exemplo, um SIG poderia permitir a um gerente calcular a resposta para uma emergência facilmente no caso de um desastre natural, ou um SIG poderia ser usado para achar fontes de água potável que precisariam de proteção contra poluição.

Na sentido mais rígido, um SIG é um sistema de computador capaz de fazer montagem, armazenamento e manipulação de informações referenciadas geograficamente, isto é, dados que são identificados de acordo com as suas localizações, podendo esse tipo de informação ser desde um interruptor em uma sala de um edifício até uma montanha em um continente.

O projeto de um Sistema de Informação Geográfica, em geral, lida com diversos quesitos dentre os quais podem se destacar:

- Relacionamento entre informações de fontes diferentes;
- Aquisição de dados;
- Integração de dados;
- Projeções e registros;
- Estruturas de dados;
- Modelagem de dados.

O que torna os Sistemas de Informações Geográficas especiais não é o puro e simples fato dos mesmos tratarem-se de um tipo de banco de dados com referências espaciais e sim a possibilidade que tais dados georeferenciados reunidos em um banco permitirem a execução de diversas análises complexas. Como exemplos de tais análises, podem ser destacadas:

- Recuperação de informação;
- Modelagem topológica;
- Fluxo de dados em malhas de redes de natureza variada;
- Sobreposição;
- P rodução de dados.

II.2.1 Relacionamento entre Informações de Fontes Diferentes

Se fosse possível relacionar informação sobre a chuva de um estado com fotografias aéreas de um município desse estado, seria possível contar quais áreas irrigadas secam em certas épocas do ano. Um SIG que pode usar informação de muitas fontes diferentes de muitas formas diferentes pode ajudar com tais análises. A exigência primária para os dados da fonte é que as localizações para as variáveis são conhecidas. Localização pode ser descrita através de coordenadas (x, y e z) de longitude, latitude e elevação, ou por sistemas como códigos de endereçamento postal ou ainda por marcadores de quilômetros das estradas.

Qualquer variável que pode ser localizada espacialmente, pode ser inserida em um SIG. Várias bases de dados computadorizadas que podem ser adicionadas diretamente em um SIG estão sendo produzidas por agências federais e empresas privadas. Diferentes tipos de dados que se encontram em forma de um mapa podem ser adicionados a um SIG.

Um SIG pode também converter informação existente na forma digital, mas que ainda não se encontra em forma de mapa. Tais informações podem ser reconhecidas e utilizadas (inseridas) em um mapa. Por exemplo, podem ser analisadas imagens de satélite digitais para produzir um mapa como uma camada de informação digital sobre coberturas de vegetação.

II.2.2 Aquisição de Dados

Os Sistemas de Informações Geográficas podem capturar a informação na forma de mapa de diversas maneiras. O mapa capturado deve ser colocado em uma forma que o computador possa reconhecer, pois só assim o mesmo poderá armazenar e processar as informações de um mapa. Mapas podem ser digitalizados ou feitos manualmente com um ¨mouse¨ de computador, ou ainda podem ser armazenadas as coordenadas cartesianas que descrevem o mapa.

Um SIG pode ser usado para enfatizar as relações de espaço entre os objetos que são traçados. Enquanto um traçado feito por uma ferramenta de desenho (CAD) pode representar uma estrada simplesmente como uma linha, um SIG também pode reconhecer aquela estrada como a fronteira entre uma área rural para plantio e uma outra área para o desenvolvimento urbano.

A aquisição de dados é a etapa mais demorada do trabalho de desenvolvimento de um SIG. Devem ser especificadas as identidades dos objetos no mapa, como também as relações de espaço entre eles. A edição de informação que é capturada automaticamente também pode ser difícil. ¨Scanners¨ eletrônicos registram marcas em um mapa da mesma maneira fiel como eles registram as características do mesmo mapa. Por exemplo, uma mancha de sujeira poderia conectar duas linhas que não deveriam ser conectadas. Os dados estranhos devem ser editados ou removidos do arquivo de dados digital.

II.2.3 Integração de Dados

Um SIG torna a integração de dados possível, ou seja, ele une ou integra informação que seria difícil de associar por qualquer outro meio. Assim, um SIG pode usar combinações de variáveis traçadas para construir e analisar variáveis novas.

A tecnologia de SIG sendo usada por uma companhia de água que possui a informação sobre o seu faturamento, poderia simular a descarga de materiais rio acima

nos sistemas sépticos em um bairro de um município. As contas mostram quanta água está sendo usada em cada endereço. A quantidade de água que um dado cliente usa irá predizer a quantidade de material que será descarregado nos sistemas sépticos asperamente, de forma que áreas de descarga séptica pesada podem ser localizadas usando um SIG.

II.2.4 Projeções e Registros

O mapa de uma propriedade pode estar em uma escala diferente de um mapa de terras. As informações de mapas em um SIG devem ser manipuladas para que elas registrem ou se ajustem às informações oriundas de outros mapas. Antes dos dados digitais serem analisados, eles podem ter que sofrer outras manipulações como, por exemplo, conversões de projeção. Isso os integra em um Sistema de Informações Geográficas. Projeção é um componente fundamental da elaboração de um mapa. Uma projeção é um meio matemático de transferir informação da superfície curva tridimensional da Terra para uma área bidimensional como papel ou uma tela de computador. Projeções diferentes são usadas para tipos diferentes de mapas porque cada projeção é particularmente apropriada a certas aplicações. Por exemplo, uma projeção que representa com precisão as formas dos continentes irá distorcer os seus tamanhos relativos.

Considerando que muito da informação em um SIG vem de mapas existentes, um SIG usa o poder de processamento do computador para transformar, em informação digital, uma projeção comum oriunda de várias fontes com projeções diferentes.

II.2.5 Estruturas de Dados

O mapa de uma propriedade relacionado a uma imagem de satélite pode ser um indicador oportuno de usos da terra? Sim, mas desde que os dados digitais são adquiridos e armazenados de vários modos, as duas fontes de dados podem não ser completamente compatíveis. Assim um SIG deve poder converter dados de uma estrutura para outra.

Dados de um SIG podem ser lidos da imagem de um satélite que foi interpretado por um computador para produzir um mapa de uso da terra em um formato conhecido como ¨raster¨. Os arquivos de dados ¨raster¨ consistem de linhas de células uniformes

codificadas de acordo com os valores dos dados. Um exemplo seria a classificação da cobertura da terra.

Os arquivos de dados “raster” podem ser manipulados rapidamente pelo computador, mas eles são freqüentemente menos detalhados e visualmente menos agradáveis do que os dados de arquivos vetorizados que podem aproximar a aparência de mapas tradicionais traçados manualmente. Dados digitais vetorizados têm sido adquiridos como pontos, linhas (uma série de coordenadas de ponto), ou áreas (formas limitadas por linhas). Um exemplo de dados tipicamente mantidos como um arquivo de vetores seria as fronteiras de uma propriedade para subdivisão de moradia.

Reestruturação de dados pode ser executada por um SIG para converter dados em formatos diferentes. Por exemplo, um SIG pode ser usado para converter um mapa de uma imagem de satélite para uma estrutura de vetores, gerando linhas ao redor de todas as células com a mesma classificação, enquanto determina as relações espaciais das células, tais como adjacência ou inclusão.

II.2.6 Modelagem de Dados

É difícil relacionar mapas de terras com quantidade de chuva registrada em pontos diferentes como aeroportos, estações de televisão e escolas. Porém, um SIG pode ser usado para descrever características de duas e três dimensões da superfície, do subsolo e da atmosfera da Terra, oriundas de diversos pontos de informação. Por exemplo, um SIG pode gerar um mapa rapidamente com linhas que indicam quantidades de chuva.

Tal mapa pode ser pensado como um mapa de contorno de chuva. Muitos métodos sofisticados podem calcular as características de superfícies de um número limitado de medidas de ponto. Um mapa de contorno bidimensional criado da superfície que modela as medidas de pontos de chuva pode ser sobreposto com qualquer outro mapa em um SIG que cobre a mesma área.

II.3 Projeto de Banco de Dados

Um banco de dados pode ser projetado tendo por base várias tecnologias, tanto em nível conceitual, quanto em nível do modelo lógico. Veremos a seguir os modelos ER e Relacional [18].

II.3.1 Os Modelos ER e Relacional

O modelo conceitual adotado nesta dissertação de mestrado é representado na figura 1. Cabe ressaltar que o mesmo é fundamentado no modelo Entidade Relacionamento (ER). Posteriormente, este modelo conceitual passa por um algoritmo para então ter como resultado um modelo lógico relacional.

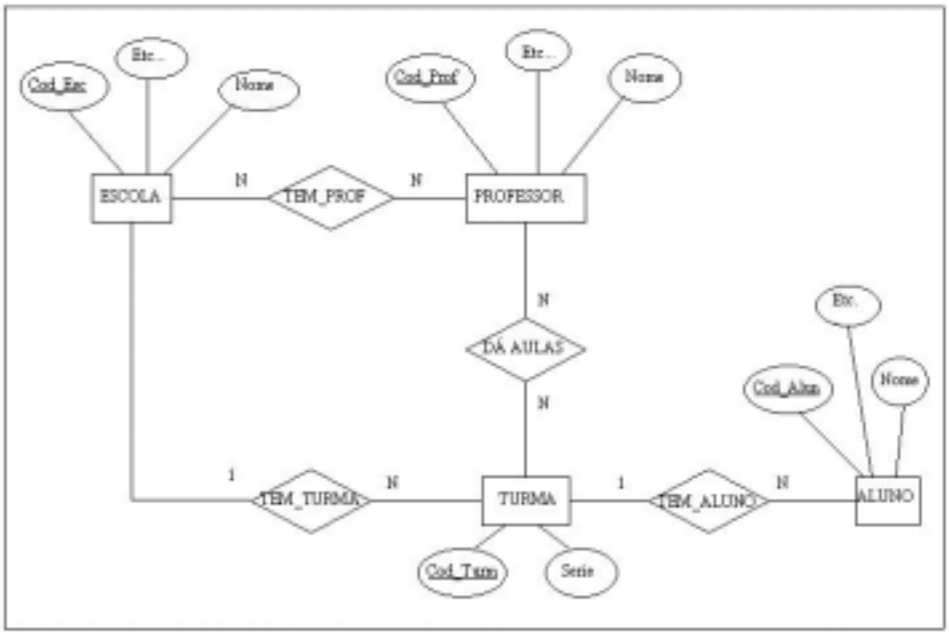


Figura 1 – Modelo ER adotado

II.3.1.1 O Modelo ER

Os tipos-entidades (classes de objetos) modelados devem ter os seus significados esclarecidos em um Dicionário de Dados (DD). Isto porque modelo algum é suficientemente claro se não for acompanhado de uma definição formal de seus elementos (dicionário de dados ou similar). Devem-se conceituar as entidades mostrando: as regras que as definem (e as exceções), os exemplos ilustrativos, as correlações entre conceitos e outras informações que facilitem o entendimento.

Uma entidade (tipo-entidade/classe) define um conjunto de objetos que possuem os mesmos atributos. Cada entidade no banco de dados é descrita por um nome e por uma lista de atributos.

No diagrama do modelo ER, cada atributo chave (pode ser apenas um atributo, ou um conjunto de atributos que irá identificar unicamente um objeto), possui o seu nome sublinhado.

A razão de cardinalidade especifica o nº· de instâncias em um relacionamento de que uma dada entidade pode participar.

Na figura 1, o relacionamento binário TEM_ALUNO (TURMA:ALUNO) possui a razão de cardinalidade 1:N, significando que cada turma pode estar relacionada com vários alunos, mas que um aluno pode estar relacionado com apenas uma turma.

As razões de cardinalidade mais comuns são: 1:1; 1:N e M:N.

II.3.1.2 Conceitos do Modelo Relacional

O banco de dados é representado como uma coleção de relações (tabelas).

Cada linha da tabela representa um conjunto de valores de dados relacionados. Estes valores podem ser interpretados como fatos, descrevendo uma entidade no mundo real ou um relacionamento.

Os nomes escolhidos para a tabela e para as colunas devem ajudar na interpretação do significado para os valores que aparecem nas linhas. Todos os valores em uma coluna são do mesmo tipo de dados.

No Modelo Relacional são usadas as seguintes terminologias respectivamente, para nomear as linhas e o nome das colunas em uma tabela: tuplas e atributos (ver figura 2). Enquanto que o nome da tabela corresponde à relação. O domínio é um conjunto de valores atômicos que identifica um atributo. Como alguns exemplos de domínios temos entre outros: nomes de pessoas, de ruas, bairros (combinação das letras do alfabeto); números de telefone, cep (um conjunto de números válidos).

O Modelo Relacional apresenta algumas restrições, tais como:

a) Restrições de Domínio - especificam que o valor de cada atributo A deve ser um valor atômico do domínio dom(A).

Especifica-se um domínio através de tipos primitivos de dados, tais como **integer**, **float**, **char**, **date**, **time** etc.

b) Restrições de chave - uma relação é definida como um conjunto de tuplas.

Por definição, todos os elementos de um conjunto são distintos. Portanto todas as tuplas em uma relação devem ser distintas.

Isto significa que não se podem ter duas tuplas com a mesma combinação de valores para todos os seus atributos.

c) Super-chave - conjunto de um ou mais atributos que, tomados coletivamente, nos permitem identificar de maneira unívoca uma entidade em um conjunto de entidades (tipo-entidade).

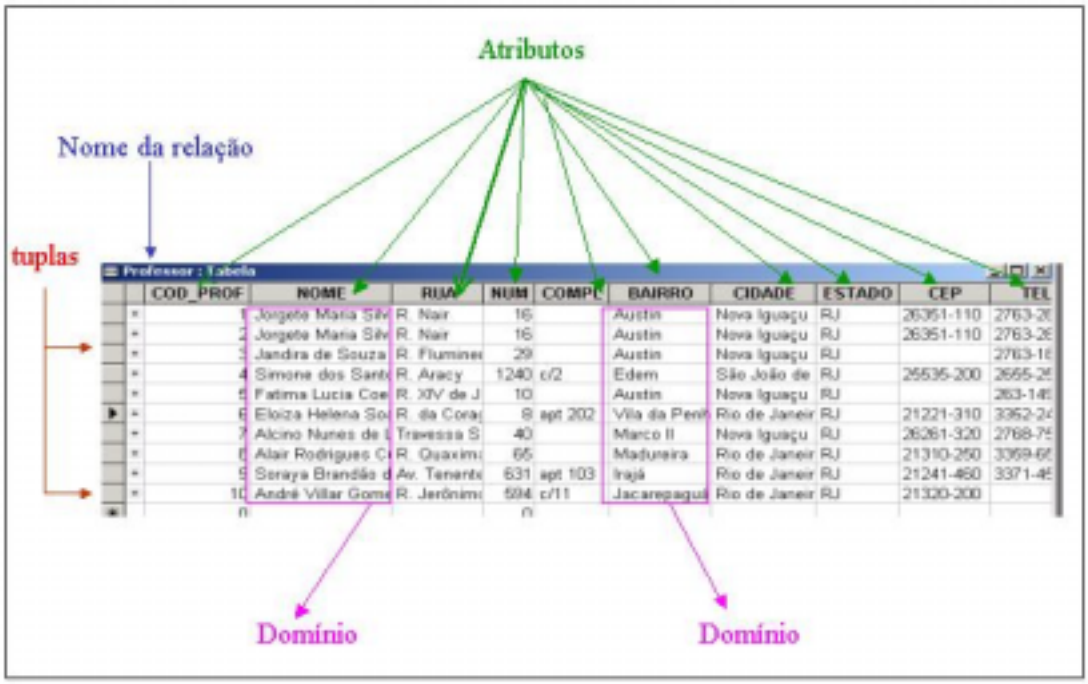


Figura 2 – Terminologias do Modelo Relacional

d) Chave candidata - é uma das chaves de um esquema de relação R.

e) Chave primária - é uma chave candidata escolhida pelo projetista do BD para a identificação de entidades em um conjunto de entidades (por convenção são sublinhadas no esquema da relação).

f) Restrições de Integridade de Entidade - estabelece que nenhuma chave primária pode ter valor nulo.

Esta restrição tem por objetivo garantir a identificação de todas as tuplas, uma vez que a chave primária é usada para identificar as tuplas na relação.

II.3.1.3 Mapeamento ER → Relacional

Um esquema de Banco de Dados Relacional pode ser derivado de um Modelo Conceitual que utilize o Modelo Entidade-Relacionamento. Isto será visto nos passos a seguir.

a) Passo 1: incluir as Relações Entidades

Para cada tipo entidade ***E*** regular do esquema ER, crie uma tabela ***R*** que inclua todos os atributos simples de ***E***. (figura 3)

– Inclua somente os componentes simples de um atributo composto.

–Escolha um dos atributos chaves de ***E*** como uma chave primária para ***R***. Se a chave escolhida for composta, o conjunto de atributos simples que a formam definirão a chave primária de ***R***.

Figura 3 - Após o passo 1

b) Passo 2: incluir as Relações Tipo-Entidades Fracas

Para cada tipo-entidade fraco W, com tipo-entidade proprietária E do esquema ER, crie uma tabela R e inclua todos os atributos simples (ou componentes simples de atributos compostos) de W como atributos de R.

Inclua como chave estrangeira de R a chave primária da tabela que corresponde ao tipo-entidade proprietário E (de W). Isto corresponde ao relacionamento de identificação *(identifying relationship).* A chave primária de R é a combinação da(s) chave(s) primária(s) do(s) proprietário(s) de W e da chave parcial da entidade fraca W.

O modelo de dados do Matrigeo não apresenta tipo entidades fracas.

c) Passo 3: trata dos relacionamentos 1:1 do esquema ER

Para cada relacionamento binário ***R (1:1)*** do esquema ER, identifique as tabelas ***S*** e ***T*** que correspondem aos tipos-entidades que participam de ***R***.

Escolha uma das tabelas, p. ex. ***S***, e inclua, como chave estrangeira de ***S***, a chave primária de ***T*** (para o papel de ***S*** é melhor escolher o tipo-entidade com participação total em ***R***).

Inclua todos os atributos simples (ou componentes simples de atributos compostos) do relacionamento ***R (1:1)*** como atributos de ***S***.

Como pode ser visto na figura 1, o modelo conceitual contém apenas relacionamentos do tipo M:N e 1:N e não possui relacionamentos do tipo 1:1.

d) Passo 4: trata dos relacionamentos 1:N do esquema ER

Para cada relacionamento binário regular (não-fraco) ***R (1:N)*** do esquema ER, identifique a tabela ***S*** que representa o tipo-entidade que participa no lado ***N*** do relacionamento. (figura 4)

Inclua, como chave estrangeira em ***S*** a chave primária da tabela ***T*** que representa o outro tipo-entidade participante de ***R***.

Inclua qualquer atributo simples (ou componentes simples de atributos compostos) do relacionamento ***1:N*** como atributos de ***S***.

e) Passo 5: trata dos relacionamentos M:N do esquema ER

Para cada relacionamento binário ***R (M:N)*** crie uma nova tabela ***S*** para representar ***R***. (figura. 5)

Inclua, como chaves estrangeiras em ***S***, as chaves primárias das tabelas que representam os tipo-entidades participantes do relacionamento. A combinação destas chaves estrangeiras formará a chave primária de ***S***.

Inclua qualquer atributo simples do relacionamento M:N como atributo de ***S***.

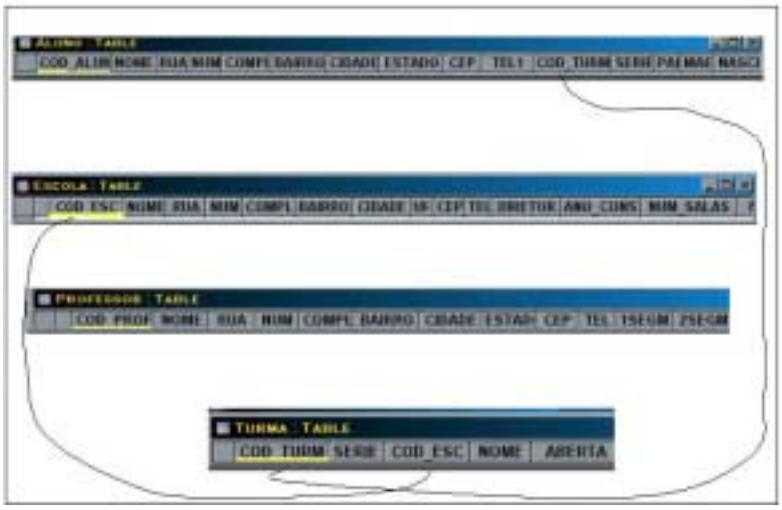


Figura 4 - Após o passo 4

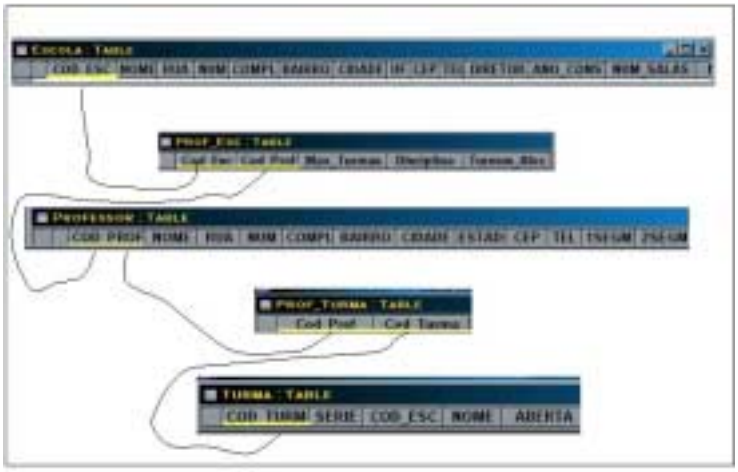


Figura 5 - Após o passo 5

f) Passo 6: trata dos atributos multi-valorados no esquema ER

Para cada atributo multi-valorado ***A*** crie uma nova tabela ***R*** que inclua um atributo correspondente de ***A*** mais a chave primária ***K*** (como chave estrangeira em ***R***) da tabela que representa o tipo-entidade ou relacionamento que possua ***A*** como atributo. A chave primária de ***R*** é a combinação de ***A*** e ***K***. Se o atributo ***A*** é composto deve-se incluir todos os seus componentes simples.

O modelo também não apresenta atributos multi-valorados.

g) Passo 7: trata dos relacionamentos de grau *n* no esquema ER

Para cada relacionamento ***R*** de grau ***n***, ***n > 2***, crie uma nova tabela ***S*** para representar ***R***. Inclua como chaves estrangeiras em ***S*** as chaves primárias das tabelas que representam as entidades-tipos participantes.

Inclua também qualquer atributo simples do relacionamento *n-ário* (ou componentes simples de atributos compostos) como atributos de S.

A chave primária de ***S*** usualmente é uma combinação de todas as chaves estrangeiras que referenciam as tabelas que representam os tipos-entidades participantes.

Se a restrição de participação ***(min, max)*** de um dos tipos-entidades ***E*** participantes em ***R*** possui ***max=1***, então a chave primária de ***S*** pode ser a única chave estrangeira que referencia a tabela ***E´*** correspondente a ***E,*** porque, neste caso, cada entidade ***e*** em ***E*** participará com no máximo uma instância no relacionamento ***R*** e assim pode identificar univocamente aquela instância do relacionamento.

Este caso também não ocorre no modelo.

II.4 Comentários

Este foi um capítulo voltado para as metodologias aplicadas no projeto, tendo um embasamento teórico.

O sistema proposto nesta dissertação utiliza arquivos de banco de dados e arquivos digitais relacionados aos temas de um mapa. Sendo assim, fez-se necessário uma pequena descrição sobre os Sistemas de Informação Geográfica e sobre a construção de um projeto de banco dados.

Nos capítulos III e IV serão analisados respectivamente como foram adquiridos os mapas digitais; e visualizados todos os arquivos de banco de dados e tema dos mapas.

CAPÍTULO III

Organização do Projeto

III.1 Introdução

Estará sendo abordada, a partir deste capítulo, a estruturação de todo o sistema **MatriGeo**, do ponto de vista do usuário. Serão descritas as suas funcionalidades (*interface* homem-máquina) ficando para o próximo capítulo uma breve exposição da sua arquitetura interna.

Este capítulo trata-se em responder à seguinte pergunta: O que faz o **MatriGeo**? Em outras palavras, será apresentado um pequeno tutorial com todas as funcionalidades deste sistema, para tornar o mais fácil possível o seu entendimento por parte do usuário.

III.2 Caracterização da Pesquisa

Este item é dividido em dois sub-ítens que tratam da localização e dados gerais do município de Nova Iguaçu, como também de uma pequena abordagem da educação municipal.

II.2.1 Localização e Dados Gerais do Município Alvo

O Município de Iguassú foi criado a 15 de janeiro de 1833 com sua sede instalada às margens do rio que lhe deu o nome. O progresso deve ser creditado à abertura da Estrada Real do Comércio, primeira via aberta no Brasil para o escoamento

do café do interior do país. Em 1891, passa a ser chamada de Iguassú Velha. Neste ano, a sede do município foi transferida para o arraial de Maxambomba. Em 1916, Maxambomba passa a ser chamada de Nova Iguaçu.

Nova Iguaçu possui um total de 750.485 habitantes (1996), destes 386.400 são mulheres e 364.085 são homens. Possui 433.515 eleitores, que são alocados em 10 zonas, distribuídas em 1.377 seções eleitorais, de acordo com os dados da eleição de 2000.

Sua sede municipal está localizada pontualmente nas seguintes coordenadas geográficas, a - 22º 45`33" e - 43º 27`04" e a 25 metros de altitude. A extensão Norte-Sul é de 36 Km, enquanto que a extensão Leste-Oeste é de 19Km. Possui uma área de 524,04 Km^2, e sua densidade demográfica é de 1.413,8 hab/ Km^2.

Os municípios limítrofes são: Rio de Janeiro, Nilópolis, São João de Meriti, Duque de Caxias, Belford Roxo, Seropédica, Queimados, Japeri, Miguel Pereira e Mesquita.

Os principais acessos rodoviários são as estradas federais BR-116 (Rodovia Presidente Dutra) e BR-465 (Antiga Rodovia Rio - São Paulo); e as estaduais RJ-105, RJ-113 e RJ-111.

Nova Iguaçu é dividida em 5 setores de Planejamento [19], a saber: Setor de Planejamento Centro, Setor de Planejamento Sudoeste, Setor de Planejamento Noroeste, Setor de Planejamento Nordeste e Setor de Planejamento Norte. Existem ainda, duas áreas, denominadas de áreas não abairráveis, como pode ser observado na figura 6. Estas duas últimas áreas mencionadas, constituem na porção norte, a Reserva Biológica do Tinguá, e na porção sul, o Parque Municipal de Nova Iguaçu.

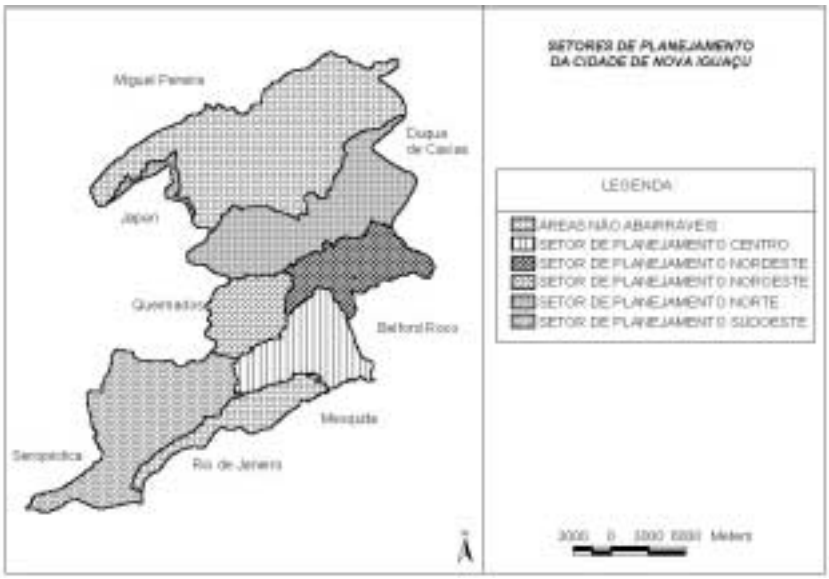


Figura 6 – Setores de Planejamento de Nova Iguaçu

II.2.2 Nova Iguaçu e a Educação Municipal

De acordo com [19], a Secretaria Municipal de Educação possui, sob sua responsabilidade 94 Unidades Escolares e 11 creches, sendo a mais nova, inaugurada em 22 de junho de 2001.

Com a emancipação do município de Mesquita, houve uma perda de 07 Unidades Escolares e 2 creches. Dentre as Unidades Escolares, inclui-se uma construção recente, cujo prédio foi entregue ao novo município em dezembro de 2000, para iniciar suas atividades.

Em contrapartida, a Prefeitura da Cidade alugou 10 prédios para atendimento às comunidades carentes de vagas na rede pública de ensino. Considerando os 10 prédios alugados, as três escolas construídas, uma reativada e as unidades que estão sendo reformadas e ampliadas, foi possível aumentar significativamente o quantitativo de vagas: 53.476 alunos estudando na rede pública municipal, abrangendo o atendimento às creches, ao Ensino Fundamental e ao Ensino de Jovens e Adultos.

O município possui 80% da população alfabetizada. A porcentagem correspondente aos chefes de famílias com mais de 8 anos de estudo é de 33%, enquanto que esta porcentagem atinge um número um pouco maior, 37%, quando se refere a chefes de família que tenham entre 4 a 8 anos de estudo.

Ainda em [19], as creches municipais atendem à população infantil, na faixa etária de 02 a 06 anos, proporcionando-lhes o desenvolvimento integral, em seus aspectos físico-psicológicos, intelectual e social, complementando a ação da família e da comunidade.

Nova Iguaçu conta com 11 creches em funcionamento, 132 profissionais e atende 319 crianças. Nas unidades de ensino as crianças ficam de 7 às 17 horas, fazem três refeições e dispõem de brinquedos pedagógicos, televisão e vídeo.

Os pais interessados em vagas para seus filhos, com idade entre 02 e 06 anos, nas creches do município de Nova Iguaçu, devem procurar a creche mais próxima de sua residência para realizar o cadastramento. O pré-requisito para matrícula da criança é a mãe estar trabalhando ou ser tutelada pelos avós.

A Secretaria de Educação da Cidade de Nova Iguaçu possui denominações próprias para definir a seriação escolar.

A Educação Infantil abrange as creches (alunos com idade entre 2 e 4 anos) e o pré-escolar (alunos com idade de 5 anos, a completar até 31 de março do ano corrente).

Já o Ensino Fundamental abrange a 1ª. Etapa (corresponde à classe de alfabetização), 2ª. Etapa (corresponde à 1ª. Série), 3ª. Etapa (corresponde à 2ª. Série) e as demais séries (3ª. Série à 8ª. Série).

III.3 Entendendo o MatriGeo

O sistema **MatriGeo** foi elaborado a partir de mapas digitais da área de Nova Iguaçu. Nesta seção estarão sendo abordadas as estruturas de como foi feito o mapa que deu origem ao sistema, bem como algumas nomenclaturas próprias usadas no **MatriGeo** e que devem ter uma pequena explicação para que não haja nenhum tipo de dúvida por parte do usuário.

III.3.1 Base de Mapas Digitais

A base de mapas em formato de arquivos digitais .DXF (padrão do AutoCad® [20]), referente ao Município de Nova Iguaçu, foi adquirida junto à Fundação CIDE do Estado do Rio de Janeiro (a mesma não passou por um processo de validação). Foram retiradas algumas camadas dessa base digital como, por exemplo, curvas de nível, pontos cotados, entre outras, e logo em seguida procedeu-se com a conversão para o formato de arquivo .SHP (*shapefile* do Arcview®). A exclusão de algumas camadas teve como principal objetivo diminuir a complexidade dos arquivos dos mapas de tal forma a facilitar a sua manipulação, sem que tenha ocorrido uma perda de dados por isso, visto que tais camadas excluídas não afetariam o desenvolvimento deste projeto.

A base digital dos mapas de Nova Iguaçu, obtida da Fundação CIDE, é constituída de 13 folhas na escala 1:10000, cujas designações são: 236D, 236F, 237A, 237B, 237C, 237D, 237E, 237F, 238C, 259B, 259D, 260A e 260B. Além da retirada de algumas camadas dessas folhas, foi montado um único arquivo com toda a base digital de mapas necessária ao projeto.

A princípio o projeto seria realizado tendo como base todo o município, entretanto, devido ao tempo pequeno e à falta de uma equipe para elaboração de um banco de dados referente a uma área de abrangência muito maior, optou-se por eleger

uma área de estudo mais restrita que irá servir de protótipo para a construção do aplicativo. Tal aplicativo, posteriormente, poderá trabalhar com uma massa de dados mais ampla, quando essa estiver disponível.

Nova Iguaçu é dividida em bairros e esses são agrupados em setores de planejamento. A área de estudo foi definida levando-se em conta essa divisão. Para fins de testes e implementações, foi escolhido o setor de planejamento noroeste, pois neste tem-se o bairro Austin, que possui o maior número de escolas por bairro.

Uma vez tendo a área definida, selecionaram-se as camadas que seriam importantes para proceder com a localização das escolas e residências dos alunos. Tais camadas são: **rios**, **logradouros**, **ferrovia**, **austin** (consiste na divisão de bairros) e **área** (identifica um conjunto de logradouros).

O mapa da área de estudo (setor de planejamento noroeste) foi elaborado a partir da junção de quatro folhas, a saber: 236-D, 236-F, 237-C e 237-E. As camadas mencionadas anteriormente de cada uma dessas folhas após serem convertidas de arquivo *dxf* para arquivos *shapefile*, passaram por um processo de tratamento chamado *intersect*. Este tratamento foi realizado no ArcView, utilizando-se a opção *GeoProcessing*. O *intersect* é realizado entre duas camadas, uma como base (austin) que será usada entre todas as outras (rios, logradouros e ferrovia).

Além das camadas mencionadas anteriormente, obtidas através da base digital da Fundação CIDE, também foi inserida a camada **escola**, cujo endereço encontra-se no banco de dados, juntamente com outras informações. O arquivo *shapefile* desta camada foi criado então, a partir do endereço de cada escola.

III.3.2 Nomenclaturas do MatriGeo

Existem algumas palavras utilizadas no sistema que precisam ter uma breve descrição, pois podem causar alguma interpretação errada sobre o objetivo da utilização da mesma para definir algumas funções. Elas estarão sendo separadas por ordem de localização no sistema: nos menus e nos *forms*.

Nos menus, as palavras que necessitam ter esta breve explicação, são:

a) Pré-Matrícula: convencionou-se em chamar de pré-matrícula esta opção do menu, pois na verdade a matrícula em si só se procede após o processamento das

turmas. Na pré-matrícula, os alunos são cadastrados, mas os mesmos só serão matriculados após o processamento das turmas.

b) Aproximar por seleção: é um tipo de *zoom*, onde um detalhe de um mapa poderá ser visualizado através de uma seleção (feita através de uma área com o *mouse*).

c) Aproximar: corresponde ao *zoom in* encontrado em vários *softwares*, como o ArcView.

d) Afastar: corresponde ao *zoom out* encontrado em vários *softwares*, como o ArcView.

e) Extensão Total: corresponde ao z*oom to full extent* encontrado em vários *softwares*, como o ArcView.

f) Deslocar: corresponde ao *pan* encontrado em vários *softwares*, como o ArcView.

Enquanto que nos *Forms* Consulta às Escolas, Base de Dados – Escola, Consulta aos Professores, Base de Dados – Professor e Consulta às Turmas Processadas são as palavras abaixo que necessitam do mesmo tratamento:

a) Avaliação: corresponde ao tipo de avaliação aplicada na escola: bimestral, trimestral, semestral etc.

b) Corrente Pedagógica: indica a qual corrente pedagógica a escola segue em sua didática: tradicional, construtivista etc.

c) Energia: indica se a escola está equipada com energia elétrica.

d) Água: indica se a escola está equipada com abastecimento de água.

e) Esgoto: indica se a escola está equipada com rede de tratamento de esgoto.

f) Primeiro Segmento: indica se a escola trabalha com turmas de primeiro segmento. Vale ressaltar que para este sistema, convencionou-se chamar de primeiro segmento as séries nas quais possuam apenas um professor por turma. Entende-se como primeiro segmento as turmas das seguintes séries: creche, pré-escola, primeira etapa, segunda etapa, terceira etapa, terceira série e quarta série.

g) Segundo Segmento: indica se a escola trabalha com turmas de segundo segmento. Vale ressaltar que para este sistema, convencionou-se chamar de segundo segmento as séries nas quais mais de um professor leciona por turma. Entende-se como segundo segmento as turmas das seguintes séries: quinta série à oitava série. As turmas de jovens e adultos também se enquadram nesta denominação, apesar do sistema estar trabalhando a priori com turmas da creche até a oitava série.

h) Código da Turma: indica o código interno do sistema para uma certa turma.

i) Série da Turma: indica a série relacionada a uma certa turma. Ex: turma 301 corresponde a uma turma de terceira série.

j) Nome da Turma: utilizando o exemplo anterior, o nome da turma será 301. Deve-se deixar bem clara a diferenciação entre o significado do código da turma e do nome da turma. Os dois são um numeral, mas adquiridos de formas diferentes.

III.4 Recursos Disponíveis

Nesta seção serão apresentados os menus e botões que fazem parte do **MatriGeo**, como podem ser observados nas figuras 7 e 8.

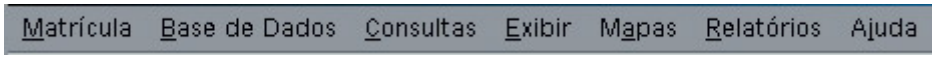


Figura 7 – *Menus* do **MatriGeo**

Figura 8 – Botões do **MatriGeo**

O último botão não aparece visível na barra de botões, pois quando o mesmo encontra-se desabilitado a figura que o representa não irá aparecer. O botão é representado pelo seguinte ícone  .

III.4.1 Menus do MatriGeo e Botões Associados

O sistema **MatriGeo** contém sete menus principais, nos quais pode-se observar a existência de sub menus (e em alguns casos, itens relacionados a estes sub menus); e oito botões, dos quais os cinco primeiros possuem as mesmas funcionalidades apresentadas nos menus (funcionando como atalho do sistema), enquanto que as funções dos outros três, somente são acionadas a partir dos mesmos e não tem correspondência na barra de menus.

1) Matrícula - Esse menu contém quatro sub menus: pré-matrícula (que por sua vez também possui submenu associado a ele), processar turmas, definir calendário e sair.

1.1) Pré – Matrícula - Dentre os submenus desta funcionalidade Matrícula, este é o único que apresenta um grupo de submenus ligados a ele. O usuário poderá escolher entre uma das três opções: cadastrar, atualizar e excluir cadastro.

1.1.1) Cadastrar – Abre uma janela "Cadastrar Pré-Matrícula" (figura 9), onde o usuário deverá preencher todos os campos e clicar em "Cadastrar".

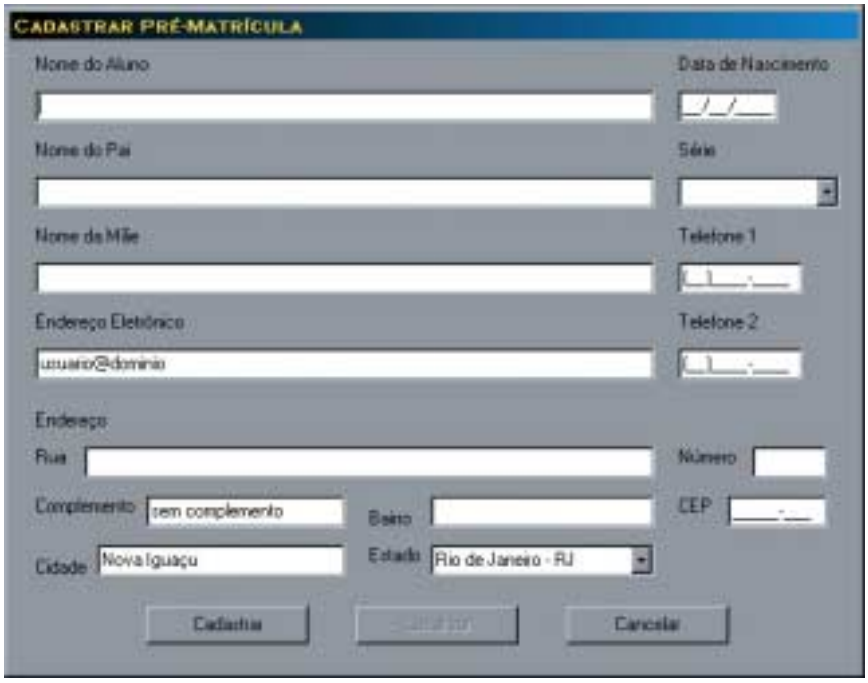


Figura 9 – Cadastrar Pré Matrícula

Caso algum campo não seja preenchido, abrirá uma janela de erro informando os campos que deverão ser preenchidos. Note que o botão "Localizar" encontra-se desabilitado. Somente após entrar com os dados do aluno e clicar em "Cadastrar" que o botão "Localizar" ficará ativo, enquanto que os botões "Cadastrar" e "Cancelar" ficarão desabilitados. A pré-matrícula só será efetivada após a realização destas duas etapas: cadastrar e localizar no mapa a residência do aluno. Ao clicar em "Localizar", será aberta uma mensagem indicando que a localização do aluno deverá ser feita com o botão direito do *mouse*. Caso o usuário não tenha localizado a área do aluno através de seu logradouro, todos os menus estarão desabilitados, com exceção de Exibir e Ajuda. Neste caso, o usuário deverá clicar em Exibir – Zoom – Logradouro, e digitar o logradouro do aluno e finalmente localizá-lo no mapa. Quando o aluno for localizado, surgirá uma mensagem em uma janela informando que o aluno foi localizado.

1.1.2) Atualizar – Primeiramente abrirá uma janela denominada “Selecionar Cadastro” (figura 10).

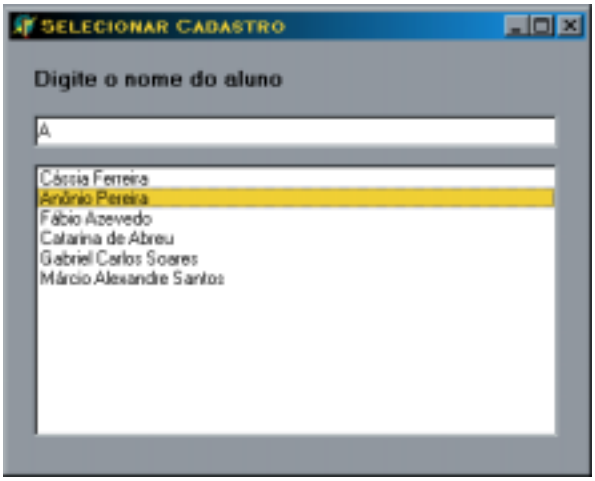


Figura 10 – Selecionar Cadastro do Aluno

O usuário deverá digitar o nome do aluno e selecionar na *listbox* o nome de tal aluno. Abrirá uma nova janela denominada “Atualizar Pré-Matrícula” (figura 11).

Esta janela é bem parecida com a “Cadastrar Pré-Matrícula”, entretanto os botões localizados na parte inferior da janela apresentam novas denominações. No lugar do botão “Cadastrar”, agora se tem o botão “Aplicar” e no lugar de “Localizar”, agora se tem “Atualizar Localização”.

Como no caso de cadastrar, quando os botões “Aplicar” e “Fechar” estiverem habilitados, o botão “Atualizar Localização” permanece desabilitado. Caso a atualização do aluno não seja referente ao seu endereço, o processo de atualização termina neste passo e o usuário deverá fechar a janela “Atualizar Pré-Matrícula”. Mas, se houver mudanças referentes ao endereço, uma mensagem irá sinalizar que as informações foram atualizadas e assim sendo, a atualização da localização será obrigatória.

Da mesma forma como o botão “Localizar” da opção “Cadastrar”, quando o usuário clicar em “Atualizar Localização”, surgirá uma mensagem informando que a localização deverá ser feita com o botão direito do *mouse*. Conseqüentemente aparecerá a mensagem informando que a localização do aluno foi atualizada.

1.1.3) Excluir Cadastro – Abrirá uma janela denominada “Selecionar Cadastro” (figura 10). O usuário deverá digitar um nome de aluno e selecionar o registro que será excluído na *listbox*. Após selecionar um aluno, uma mensagem de confirmação de exclusão será acionada. O usuário verifica o nome do aluno, do pai, da mãe e a data de

nascimento do aluno. Caso realmente seja este o registro a ser excluído, bastará que o usuário clique em “OK”. Caso contrário deverá clicar em “Cancel” (figura 12).

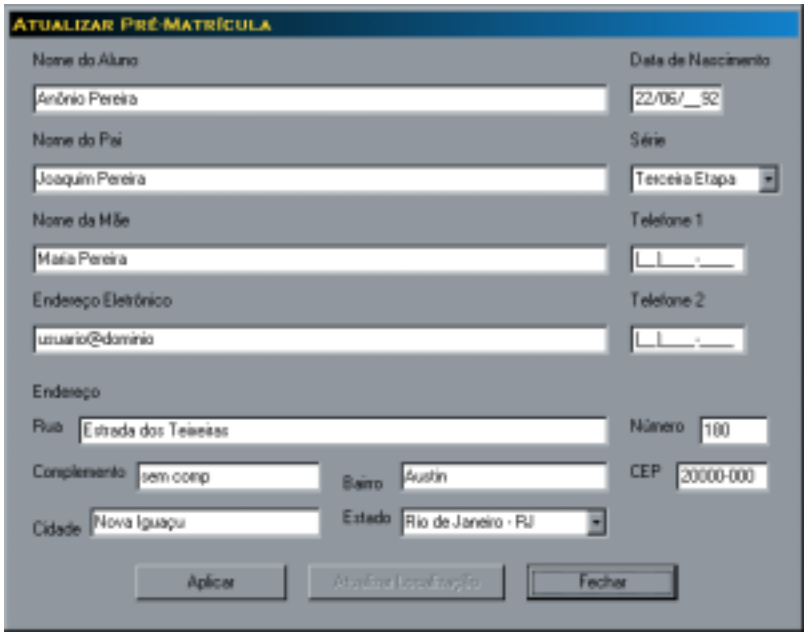


Figura 11– Atualizar Pré-Matrícula

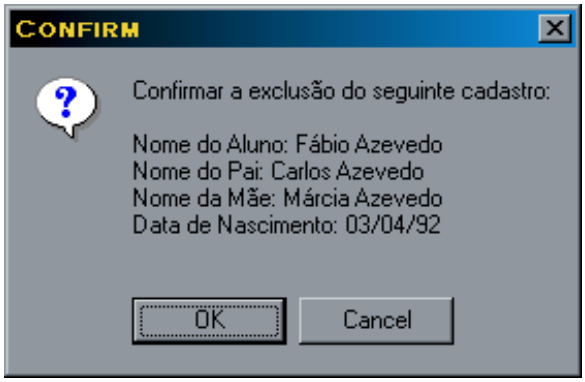


Figura 12 – Confirmação de Exclusão de Cadastro

1.2) Processar Turmas – As turmas só serão processadas após o término do período da pré-matrícula. Caso o período de cadastramento ainda esteja em vigor, uma mensagem irá informar que ainda não está no prazo para processar as turmas. Em caso contrário, se o prazo para a pré-matrícula já tenha sido encerrado, haverá o processamento das turmas.

Uma mensagem informará que, se houver o processamento, os dados anteriores sobre as turmas serão apagados. Na barra de *status* “Processando Turmas”, o usuário verá o andamento do processamento. Ao ser concluído, surgirá uma mensagem informando que as turmas foram processadas. Para ver o resultado das turmas

processadas, o usuário deverá ir a Relatórios – Turmas Processadas – Alunos Matriculados nas Escolas.

1.3) Definir Calendário – Nesta opção, o calendário de matrícula, de férias, etc, poderá ter suas datas inicial e final modificadas. Para validar o intervalo de tempo para cada uma das opções (figura 13), será necessário clicar em "Aplicar".

Figura 13 – Definir Calendário

No caso do **MatriGeo**, só foi implementada a opção de matrícula.

1.4) Sair – O usuário sairá do sistema.

2) Base de Dados – Este menu contém três sub menus: Escola, Logradouro e Professor. É através deste menu que o usuário entrará com as alterações no banco de dados. As demais tabelas do banco de dados já sofrem alterações em outros menus. É o caso de Aluno, que sofre alterações no menu Matrícula – Pré-Matrícula; Calendário, que sofre alterações em Matrícula – Definir Calendário e; Turma, que não pode ser alterada individualmente para que não haja inconsistências. As alterações são feitas a partir do menu Matrícula – Processar Turmas.

2.1) Escola – Ao clicar nesta opção uma janela será aberta com a seguinte denominação: "Base de Dados – Escola" (figura 14). Nesta janela são apresentadas todas as informações contidas na tabela "Escola".

O usuário poderá escolher uma das opções abaixo:

a) Pesquisar – para pesquisar uma escola o usuário deverá digitar uma palavra chave, clicar no botão "Pesquisar" e escolher na lista de escolas que contenham tal palavra, aquela à qual ele irá querer ver e/ou alterar as informações.

b) Incluir – Ao clicar neste botão, o usuário deverá incluir um novo registro de uma nova escola. Todos os campos deverão ser preenchidos, caso contrário, uma mensagem de erro será aberta, contendo todos os campos que se encontram vazios e que deverão ser preenchidos. Os demais botões estarão desabilitados, com exceção de "Aplicar". Ou seja, uma vez que o usuário clicar em "Incluir", deverá preencher todos

os campos e clicar em “Aplicar”. Somente após estes passos é que os demais botões retornarão a ficar habilitados.

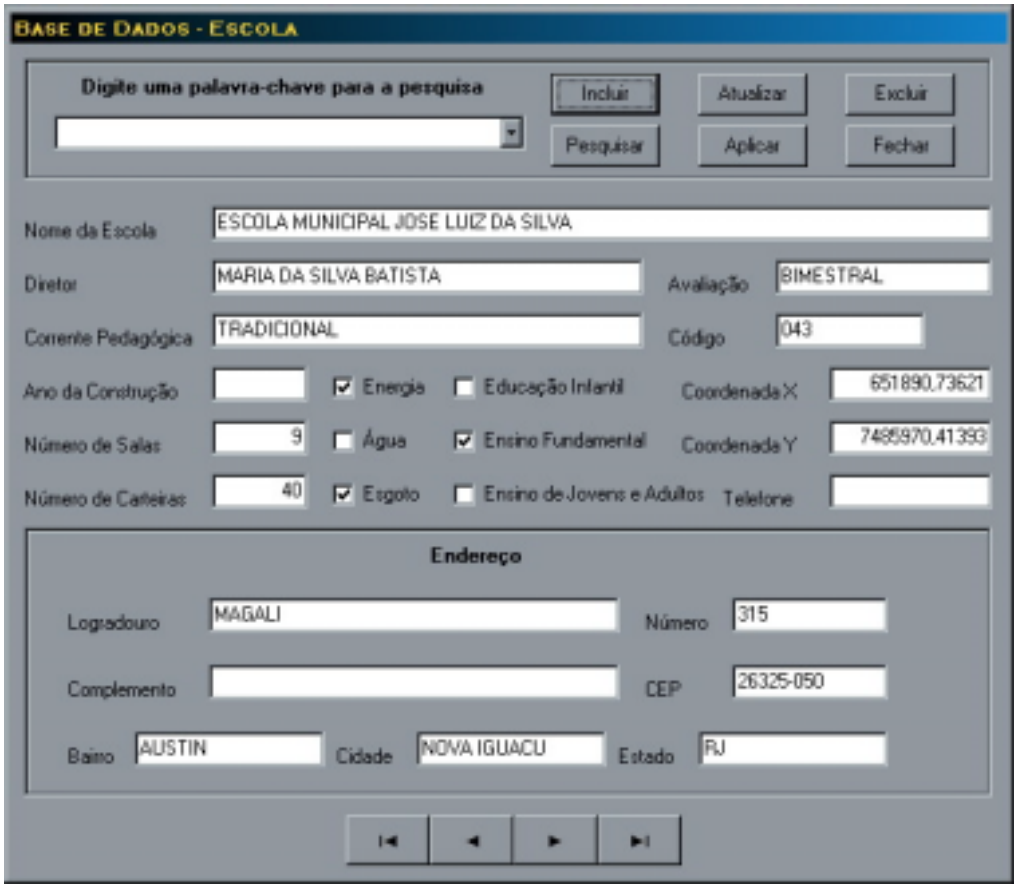


Figura 14 – Base de Dados Escola

c) Atualizar - O usuário poderá atualizar as informações de uma determinada escola. Ele pode utilizar o botão “Pesquisar” para localizar a escola que sofrerá as alterações. Neste caso, como visto anteriormente, todos os outros botões também estarão desabilitados, com exceção de “Aplicar”. Ao clicar nesse botão, os demais voltarão a ficar habilitados.

d) Excluir – O usuário deverá pesquisar e localizar a escola que queira excluir. Observe que ao clicar neste botão, irá aparecer uma mensagem de confirmação de exclusão daquela escola. Se o usuário clicar em “OK”, o registro de tal escola será apagado. Mas se ele desistir da exclusão, poderá clicar em “Cancel” e a operação será cancelada.

e) Aplicar – Este botão está associado aos botões “Incluir” e “Atualizar”, como já foi visto nos itens *b* e *c*.

f) Fechar – Irá fechar a janela.

2.2) Logradouro – Em primeiro lugar essa opção só estará habilitada quando o mapa selecionado for “setor de planejamento região noroeste”. Em segundo lugar, para que possam ser alterados os dados desta opção, é necessário que as camadas “bairros” e “quadras de localização” estejam selecionadas. Então, ao clicar nesta opção uma janela será aberta com a seguinte denominação: “Base de Dados – Logradouro” (figura 15). Nesta janela são apresentados o nome do logradouro e a quadra de localização a que este logradouro está inserido.

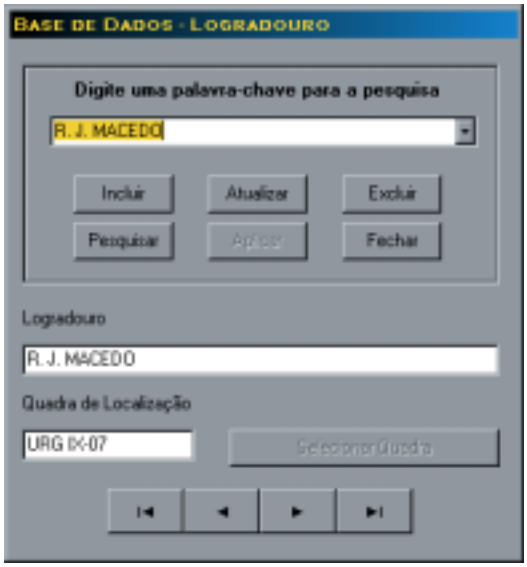


Figura 15 – Base de Dados Logradouro

O usuário poderá escolher uma das opções abaixo:

a) Pesquisar – para pesquisar um logradouro o usuário deverá digitar uma palavra chave, clicar no botão “Pesquisar” e escolher na lista de logradouros que contenham tal palavra, aquele ao qual ele irá querer ver as informações e posteriormente excluir ou alterar o nome do logradouro.

b) Incluir – Ao clicar neste botão, o usuário deverá incluir um registro de um novo logradouro. Os demais botões estarão desabilitados, com exceção de “Aplicar” e “Excluir”. No caso de incluir, o campo referente ao nome de logradouro estará aberto para edição, mas o campo referente à quadra de localização só será preenchido após clicar em “Selecionar Quadra” e seguir os passos que serão indicados (aperte a tecla ctrl e o botão esquerdo do *mouse* para selecionar uma quadra no mapa). Em seguida, após clicar na quadra (a mesma se destaca, assumindo a coloração cinza escura), para confirmar tal seleção o usuário deverá clicar no botão “confirmar seleção da quadra” , localizado na barra de ferramentas.

Para finalizar a operação deve-se clicar em “Aplicar”. Somente após estes passos é que os demais botões retornarão a ficar habilitados.

c) Atualizar - O usuário poderá atualizar as informações de um determinado logradouro. Ele pode utilizar o botão “Pesquisar” para localizar o logradouro que sofrerá as alterações. Neste caso, como visto anteriormente, todos os outros botões também estarão desabilitados, com exceção de “Aplicar”. Ao clicar nesse botão, os demais voltarão a ficar habilitados. Nesta opção não é possível alterar a quadra de localização, mas o nome do logradouro poderá ser alterado.

d) Excluir – O usuário deverá pesquisar e localizar o logradouro que queira excluir. Observe que ao clicar neste botão, irá aparecer uma mensagem de confirmação de exclusão daquele logradouro. Se o usuário clicar em “OK”, o registro de tal logradouro será apagado. Mas se ele desistir da exclusão, poderá clicar em “Cancel” e a operação será cancelada.

e) Aplicar – Este botão está associado aos botões “Incluir” e “Atualizar”, como já foi visto nos itens *b* e *c*.

f) Fechar – Irá fechar a janela.

2.3) Professor – Ao clicar nesta opção uma janela será aberta com a seguinte denominação: “Base de Dados – Professor” (figura 16). Nesta janela são apresentadas as informações de um certo professor e são referentes ao endereço, telefone, e-mail, nome, código e se tal professor leciona no primeiro ou segundo segmento.

O usuário poderá escolher uma das opções abaixo:

a) Pesquisar – Para pesquisar as informações de um professor, o usuário deverá digitar uma palavra chave, clicar no botão “Pesquisar” e escolher na lista de professores que contenham tal palavra, aquela à qual ele irá querer ver as informações.

b) Incluir – Ao clicar neste botão, o usuário deverá incluir um novo registro de um novo professor. Todos os campos deverão ser preenchidos, caso contrário, uma mensagem de erro será aberta, contendo todos os campos que se encontram vazios e que deverão ser preenchidos. Os demais botões estarão desabilitados, com exceção de “Aplicar”, “Excluir” e “Alocação nas Escolas”. Ou seja, uma vez que o usuário clicar em “Incluir”, deverá preencher todos os campos e clicar em “Aplicar”. Somente após estes passos é que os demais botões retornarão a ficar habilitados.

c) Atualizar - O usuário poderá atualizar as informações de um determinado professor. Ele pode utilizar o botão “Pesquisar” para localizar o professor que sofrerá as alterações. Neste caso, como visto anteriormente, todos os outros botões também

estarão desabilitados, com exceção de “Aplicar” e “Alocação nas Escolas”. Ao clicar nesse botão, os demais voltarão a ficar habilitados.

d) Excluir – O usuário deverá pesquisar e localizar o professor que queira excluir. Observe que ao clicar neste botão, irá aparecer uma mensagem de confirmação de exclusão daquela escola. Se o usuário clicar em “OK”, o registro de tal professor será apagado. Mas se ele desistir da exclusão, poderá clicar em “Cancel” e a operação será cancelada.

e) Aplicar – Este botão está associado aos botões “Incluir” e “Atualizar”, como já foi visto nos itens *b* e *c*.

f) Fechar – Irá fechar a janela.

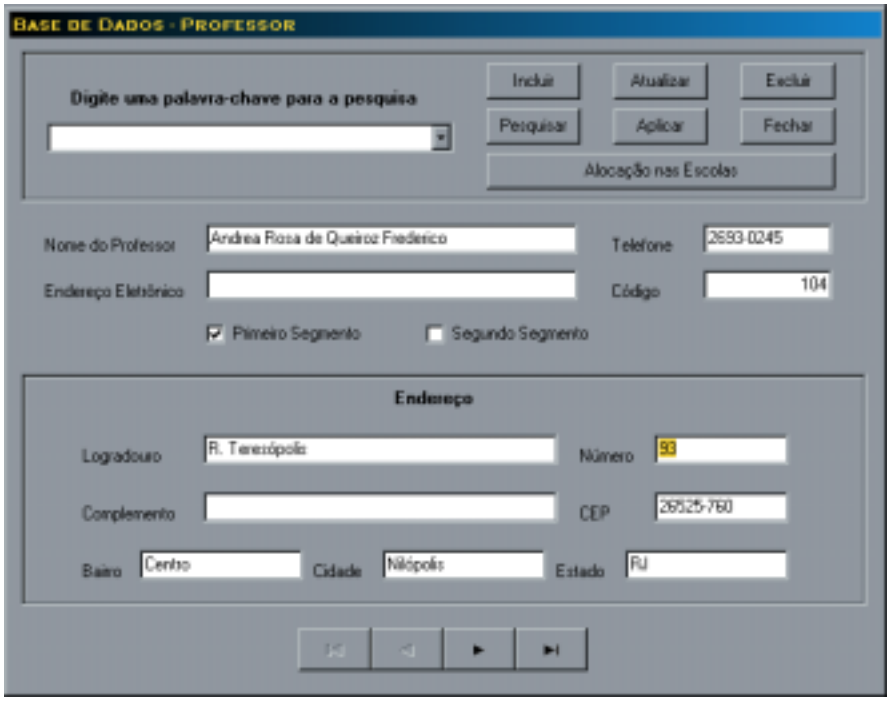


Figura 16 – Base de Dados Professor

g) Alocação nas Escolas – A janela que se abre denominada “Base de Dados – Alocação de Professor na Escola” (figura 17), está associada ao professor cuja informação aparece na janela anterior.

O usuário deverá localizar a escola na qual o professor leciona e após selecioná-la com o botão “Pesquisar”, proceder com as alterações através dos botões listados abaixo:

g.1) Incluir – Só poderão ser alteradas as informações de disciplina e número máximo de turmas. Se estes campos não forem preenchidos uma mensagem de erro será aberta informando que os mesmos se encontram em branco.

g.2) Atualizar – Só poderão ser atualizadas as informações dos mesmos campos listados no item anterior.

g.3) Excluir – Irá excluir o registro de algum professor caso este deixe de lecionar em uma certa escola. Uma janela de confirmação será aberta e o usuário deverá escolher entre excluir o registro (OK) ou cancelar a operação (Cancel).

g.4) Aplicar – Estará associado aos botões “Incluir” e “Atualizar”.

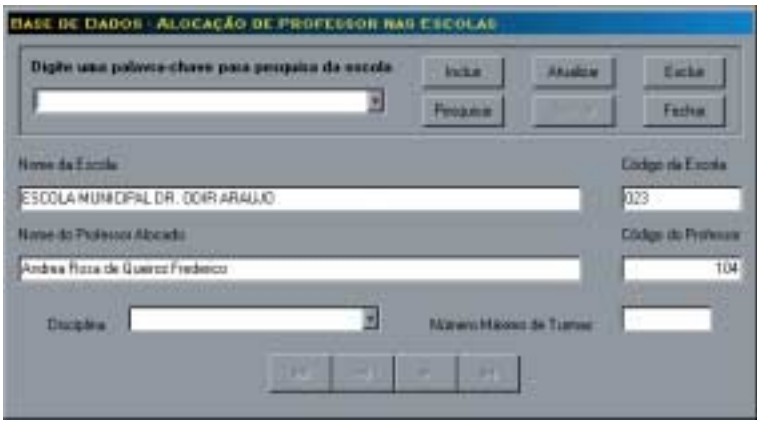


Figura 17 – Base de Dados Alocação de Professor na Escola

3) Consultas - Esse menu contém cinco sub menus: aluno, calendário, escola, professor e turmas processadas.

3.1) Aluno – Ao clicar nesta opção, será aberta a janela “Consultas aos Alunos” (figura 18).

O usuário deverá escrever uma palavra chave e clicar em “Pesquisar”. Serão listados na *combobox* todos os nomes de alunos que contenham tal palavra chave. Então o usuário deve escolher qualquer um dos nomes, para que sejam listadas todas as informações deste aluno. O usuário poderá cancelar a consulta, ou mesmo “navegar” pelo banco de dados, através da barra localizada na parte inferior da janela.

3.2) Calendário – O usuário poderá consultar o período de início e fim de uma determinada opção do calendário. Como por exemplo, o período de matrícula dos alunos.

3.3) Escola – Assim como foi visto na opção “Consulta – Aluno”, o usuário entrará com uma palavra chave na janela que irá se abrir, denominada “Consulta às Escolas” (figura 19). Ao clicar em “Pesquisar”, serão listadas todas as escolas que contenham a palavra chave. Ao selecionar com o *mouse* a escola desejada, todos os campos serão preenchidos com as informações de tal escola. O usuário poderá ainda,

cancelar a janela ou mesmo “navegar” pelas escolas através da barra localizada na parte inferior da janela.

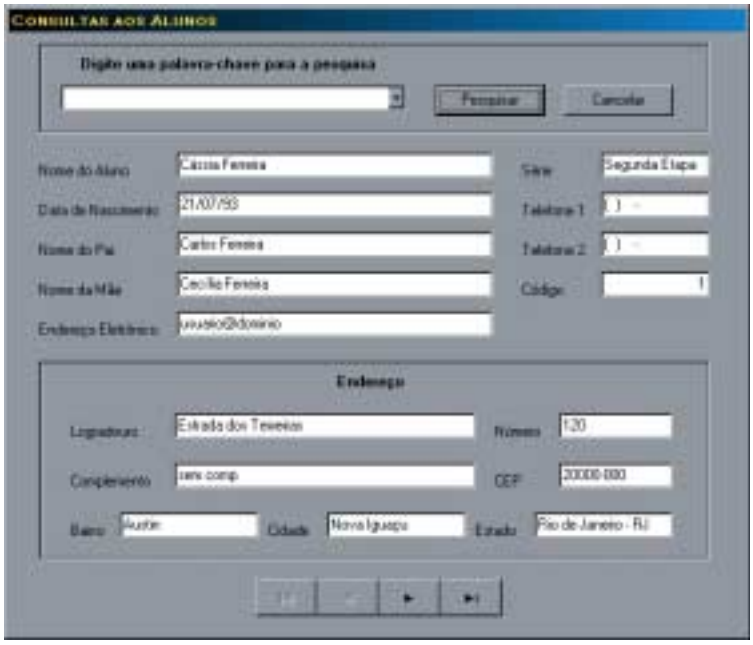


Figura 18 – Consulta aos Alunos

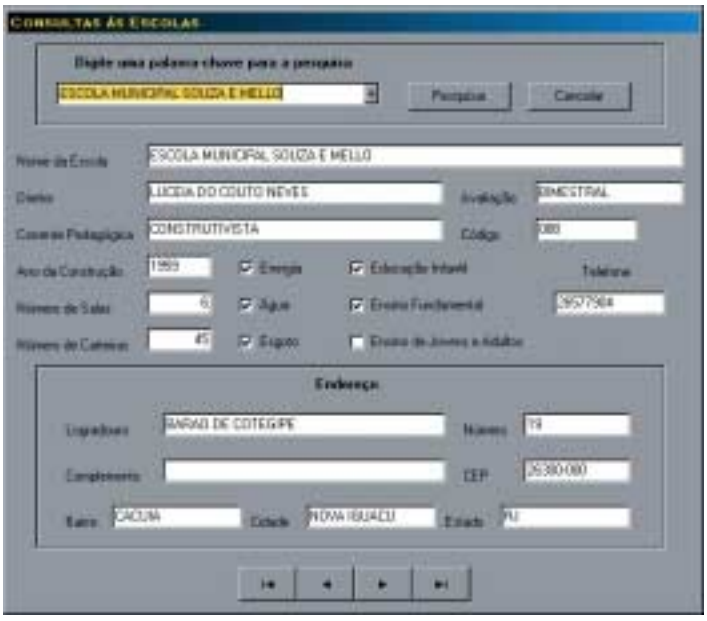


Figura 19 – Consultas as Escolas

3.4) Professor – Como no caso das consultas de alunos e escolas, ao clicar na opção “Professor”, será aberta uma janela denominada “Consulta aos Professores” (figura 20). Nesta janela, o usuário fará os mesmos procedimentos listados anteriormente: digitar uma palavra chave, clicar em “Pesquisar” e selecionar entre todos os nomes, aquele que corresponda ao professor desejado. Em seguida serão mostradas

as informações pessoais de tal professor. O usuário poderá cancelar ou “navegar” pelos professores através da barra localizada na parte inferior da janela.

Figura 20 – Consultas aos Professores

3.5) Turmas Processadas – Quando o usuário clicar nesta opção, será aberta uma janela denominada “Consulta às Turmas Processadas” (figura 21). O usuário deverá digitar uma palavra-chave para localizar uma escola desejada e clicar em “Pesquisar”. Quando a escola é selecionada, na *combobox* “Nome da Turma”, serão listadas todas as turmas que foram geradas para aquela escola. Ao selecionar a turma desejada, os nomes dos alunos e professores daquela turma serão recuperados.

4) Exibir – Esse menu contém três sub menus: *zoom* (que por sua vez também possui submenu associado a ele), deslocar e camadas.

4.1) Zoom – Dentre os submenus desta funcionalidade Exibir, este é o único que apresenta um grupo de submenus ligados a ele. O usuário tem como opção escolher entre um dos cinco tipos de *zoom* (aproximar por seleção, aproximar, afastar, extensão total e logradouro), aquele que melhor atenda as suas necessidades.

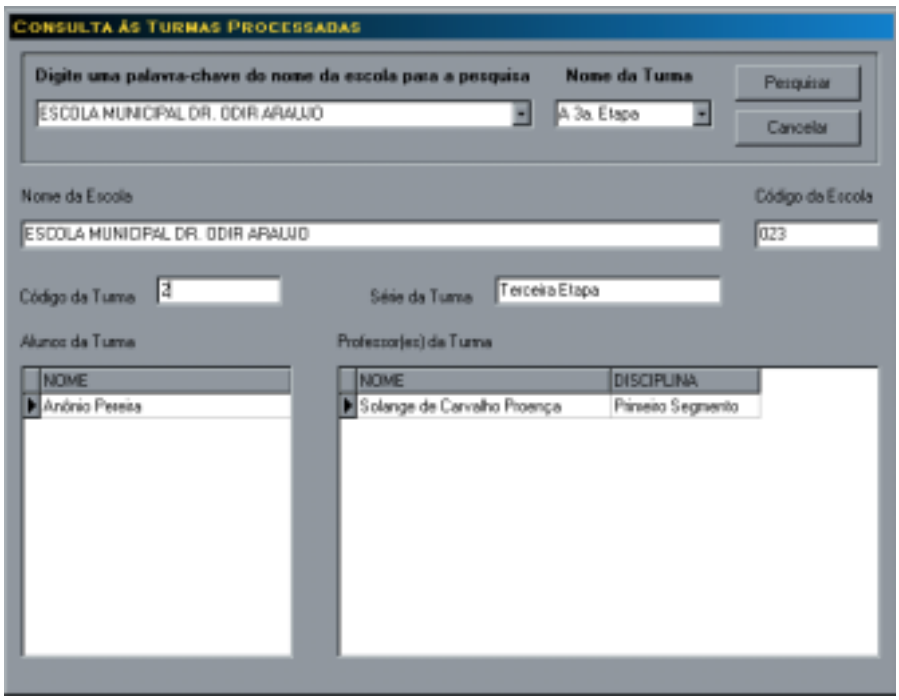


Figura 21 – Consultas às Turmas Processadas

4.1.1) Aproximar por Seleção – Esta opção é utilizada quando o usuário quer ampliar uma área determinada por ele, através de um retângulo que será feito clicando com o botão direito do *mouse* e arrastando o mesmo até formar a área desejada para a ampliação.

Ao escolher esta opção, serão desabilitadas as funções de zoom aproximar, *zoom* afastar e deslocar. Isto ocorre porque todas elas utilizam o mesmo evento *MouseDown* do objeto Mapa_Austin. Estas funções só serão habilitadas novamente depois que for deselecionada a opção marcada como ativa.

4.1.2) Aproximar – Basta que o usuário dê um simples clique no mapa para que haja uma ampliação da área clicada. Quando ela se encontra selecionada; deslocar, *zoom* aproximar por seleção e afastar ficam desabilitadas. Como mencionado no item anterior, para que as demais funcionalidades voltem a ficar habilitadas, deve-se remover a seleção desta opção clicando com o *mouse* novamente sobre ela.

4.1.3) Afastar – Basta um clique no mapa para que haja um recuo na área (redução). Como já foi mencionado nas opções anteriores, quando esta se encontra habilitada, as funções de deslocar, aproximar por seleção e aproximar, permanecerão desabilitadas até que o usuário remova a seleção da opção afastar.

4.1.4) Extensão Total – Ao ser clicada, o mapa volta a sua extensão total.

4.1.5) Logradouro – O usuário também pode selecionar o *zoom* para uma área específica, ou seja, optar por visualizar uma área associada a um logradouro. Após

clicar em *Logradouro*, irá se abrir a janela Selecionar Logradouro (esta janela é igual a Selecionar Cadastro do Aluno). Nela o usuário deverá digitar o logradouro que vai querer ter como referência para este *zoom*. Ao digitar uma palavra chave e clicar em *Enter*, na *ListBox* serão listados todos os logradouros que contenham tal palavra chave. O usuário deverá selecionar o logradouro desejado e ao clicar novamente em *Enter*, o *zoom* focalizará a área que contenha tal logradouro.

4.2) Deslocar – Quando o mapa está em qualquer um modo de *zoom*, com exceção do *zoom* extensão total, o usuário poderá deslocar o mapa em qualquer direção. Quando esta opção encontra-se selecionada, aproximar por seleção, aproximar e afastar; estarão desabilitadas.

4.3) Camadas – Quando o usuário clicar neste submenu, irá aparecer uma nova janela Selecionar Camadas (figura 22).

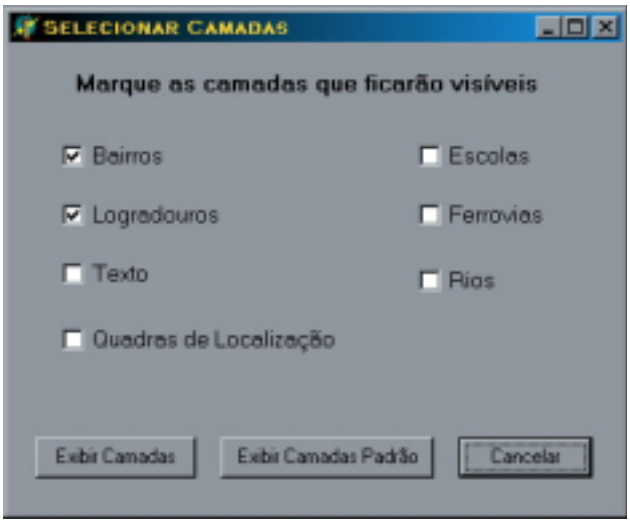


Figura 22 – Selecionar Camadas

O usuário terá duas opções:

a) Exibir Camadas – após clicar nesta, só irão aparecer as camadas marcadas no *checkbox*, ou seja, apenas as selecionadas.

b) Exibir Camadas Padrão - após clicar nesta, só irão aparecer as camadas *bairros* e *logradouros*, as mesmas que aparecem na página inicial do sistema.

5) Mapas – Este menu contém quatro submenus: Nova Iguaçu, Setor de Planejamento Região Noroeste, Escolas e Alunos Matriculados nas Escolas. Todos os mapas apresentam orientação do norte, legenda, escala e informações da origem dos dados. Todos os mapas são elaborados por programação levando-se em conta a base de

dados do **MatriGeo**. O usuário não terá a opção de edição de legendas, inclusão e/ou exclusão de camadas entre outras funcionalidades encontradas normalmente em SIG's.

5.1) Nova Iguaçu – é o mapa de entrada do **MatriGeo** (figura 23). Todos os botões e o menu "Exibir" estarão desabilitados, uma vez que essas funcionalidades não funcionam neste mapa, que serve apenas para que o usuário tenha uma noção da cidade de Nova Iguaçu como um todo e não apenas a área de estudo.

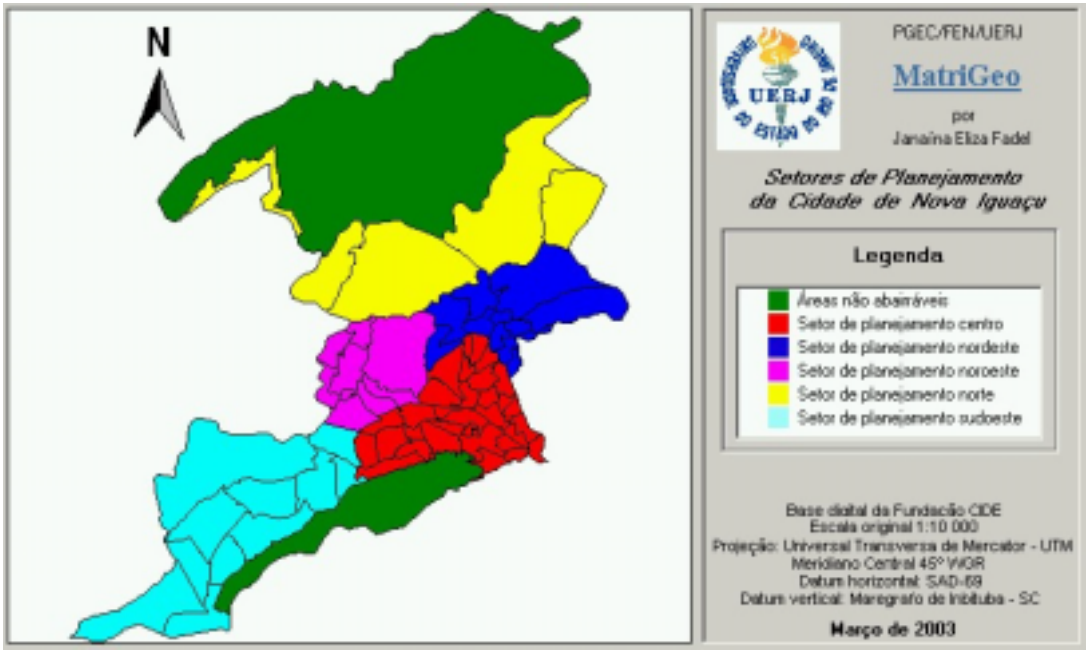


Figura 23 – Mapa Nova Iguaçu

5.2) Setor de Planejamento Região Noroeste – é o mapa principal do **MatriGeo** (figura 24), é nele que serão realizadas quase todas as funções referentes aos botões e aos menus. Os botões confirmar seleção da quadra, identificar escola e mostrar alunos da escola e série selecionadas permanecerão desabilitados quando selecionada esta opção de mapa. Como nos demais mapas, excetuando o de Nova Iguaçu, cada bairro apresenta uma coloração diferenciada. Entretanto, a coloração de um bairro é a mesma em todos os mapas. Como por exemplo, Austin que assume a coloração cinza.

5.3) Escolas – neste mapa, além de poder visualizar a divisão de bairros, como descrita acima, também serão visualizadas as onze (11) escolas cadastradas e localizadas na área de estudo. Ao deixar pressionado o botão (identificar escola) e clicar em uma das escolas, além da mesma assumir a coloração cinza (destacando-se das outras), uma janela denominada "Dados da Escola" (figura 25) se abre com as informações cadastrais de determinada escola.

Desta forma, apenas uma escola irá se encontrar em destaque (cinza) e com suas informações na janela mencionada acima. Entretanto, quando o usuário por algum descuido clicar na área do mapa, mas sem selecionar uma escola, a janela “Dados da Escola” irá ficar em branco, sem nenhuma informação.

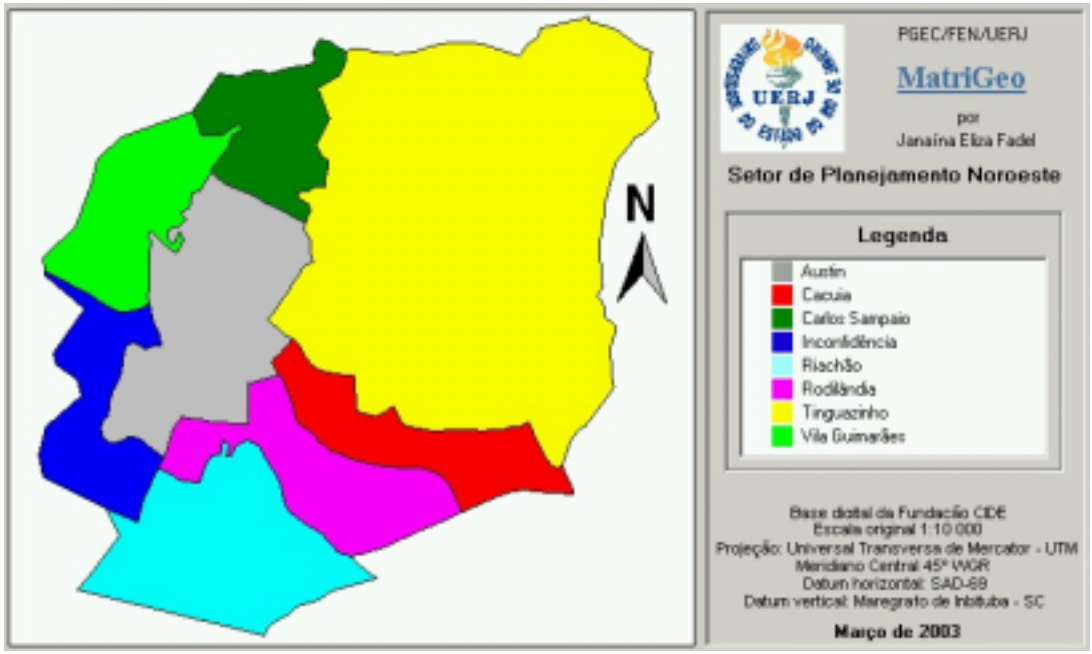


Figura 24 – Mapa Setor de Planejamento Noroeste

Figura 25 – Dados da Escola

O menu Exibir encontra-se desabilitado, assim como todos os botões, excetuando o botão Identificar Escola.

5.4) Alunos Matriculados nas Escolas – neste mapa, diferentemente dos demais que apresentam a mesma área de estudo, a divisão de bairros não é visualizada por sua coloração diferenciada. Neste mapa todos os bairros assumem a coloração amarela.

Por *default*, o mapa inicial apresenta todas as escolas. Entretanto, ao se clicar no botão (mostrar alunos da escola e série selecionadas), uma janela denominada "Selecionar Escola e Série" (figura 26) irá se abrir. Nela o usuário pesquisa a escola que quer visualizar a distribuição dos alunos por apenas uma ou todas as séries. Ao encontrar na lista a escola desejada deverá clicar em "Pesquisar". Após todas as opções selecionadas, o usuário deverá clicar em "Aceitar Seleção". Assim, aparece apenas a escola selecionada, sua área de influência e os pontos em vermelho, simbolizando os alunos.

Figura 26 – Selecionar Escola e Série

Na área logo abaixo da legenda, após os passos acima, irão aparecer o nome da escola e séries selecionadas. E na barra de *status*, o total de alunos dentro e fora da área de influência da escola.

O menu Exibir encontra-se desabilitado, assim como todos os botões, excetuando o botão "Mostrar Alunos da Escola e Série Selecionada".

6) Relatórios – Este menu contém quatro submenus: pré-matrícula, base de dados, turmas processadas e alunos não matriculados. Nenhum destes possui um item associado na barra de menu, mas sim nas janelas que serão abertas.

6.1) Pré-Matrícula – Ao clicar nesta opção, será aberta a janela "Relatórios da Pré-Matrícula" (figura 27). O usuário poderá escolher entre os três tipos de relatórios:

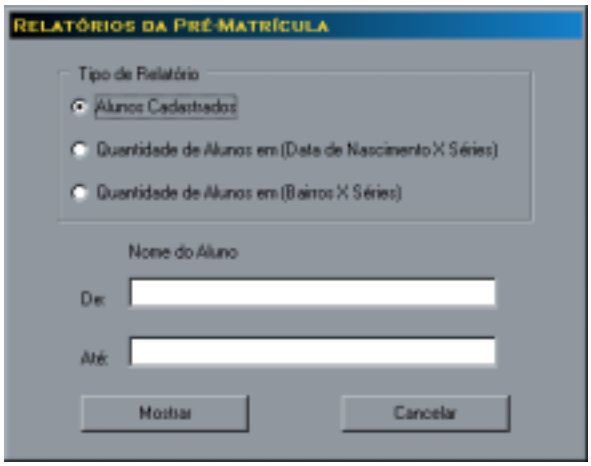


Figura 27 – Relatórios da Pré Matrícula

a) Alunos Cadastrados – somente quando esta opção está selecionada, os campos "Nome do Aluno", "De" e "Até" estarão habilitados. Ao clicar em "Mostrar", serão listados todos os alunos (ou aqueles que se encontrem no intervalo desejado).

b) Quantidade de Alunos em (Idade x Séries) – quando esta opção é selecionada, a parte inferior da janela estará desabilitada. Ao clicar em "Mostrar", serão visualizados a quantidade de alunos distribuídos pelas séries, levando-se em conta a idade dos alunos.

c) Quantidade de Alunos em (Bairros x Séries) – Nesta opção, como no caso anterior, as informações localizadas na parte inferior da janela também se encontram desabilitadas. Ao clicar em "Mostrar", serão listadas as quantidades de alunos por um determinado bairro.

6.2) Base de Dados – Ao clicar nesta opção, será aberta a janela "Relatórios de Base de Dados" (figura 28).

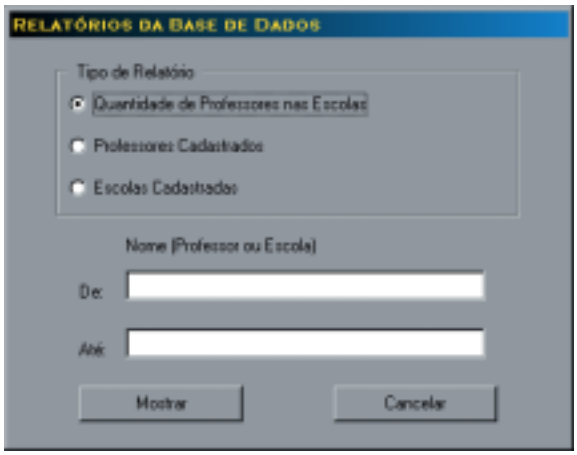


Figura 28 – Relatórios da Base de Dados

O usuário poderá escolher entre os três tipos de relatórios listados abaixo:

a) Quantidade de Professores nas Escolas – aqui o usuário também poderá entrar com um intervalo de nomes, seja do professor ou da escola. Ao clicar em “Mostrar”, serão visualizadas as seguintes informações: nome da escola, quantidade de salas, quantidade de turnos, código da escola, disciplinas, quantidade de professores, o máximo de turmas para cada disciplina, total de turmas possíveis de primeiro segmento e total de turmas de segundo segundo segmento.

b) Professores Cadastrados – ao clicar em “Mostrar”, caso não tenha sido definido nenhum intervalo, serão listados todos os professores. As informações mostradas se referem às informações pessoais de cada professor.

c) Escolas Cadastradas – como visto anteriormente, caso não seja definido nenhum intervalo de nomes, serão listadas todas as escolas cadastradas. Quando o usuário clicar em “Mostrar”, serão visualizadas todas as informações cadastrais de cada escola.

6.3) Turmas Processadas – Ao clicar nesta opção, será aberta a janela “Relatórios das Turmas Processadas” (figura 29).

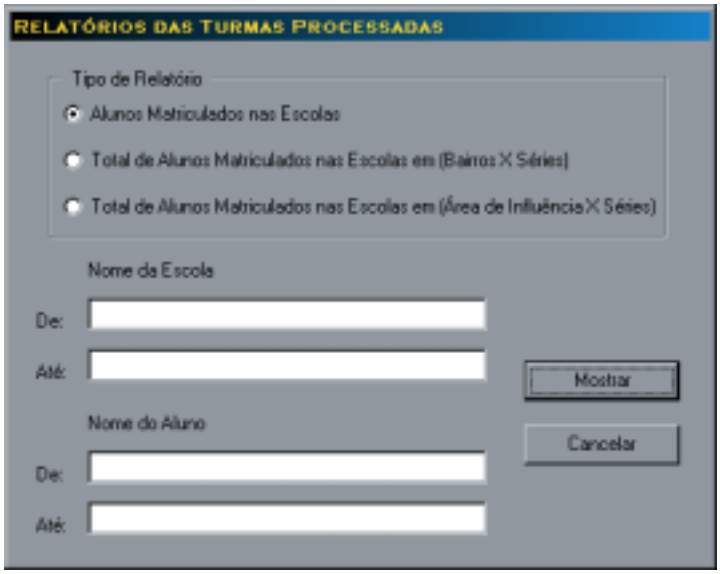


Figura 29 – Relatórios das Turmas Processadas

O usuário poderá escolher entre os três tipos de relatórios:

a) Alunos Matriculados nas Escolas – o usuário poderá fazer a busca num intervalo de nomes das escolas ou do aluno. Caso não seja inserido nenhum intervalo, ao clicar em “Mostrar” serão visualizadas todas as escolas com todos os alunos a elas associadas. As informações contidas neste formulário são: nome da escola, código da escola, nome do aluno, código do aluno, nome da turma, código da turma, série,

distância do aluno para a escola (em metros) e se o aluno se encontra na área de influência da escola (1000m).

b) Total de Alunos Matriculados nas Escolas em (Bairros x Séries) – Quando o usuário selecionar esta opção, o campo referente a "Nome de Aluno" irá ficar desabilitado. Assim sendo, a busca por intervalos de nomes só poderá ser feita pela escola. Caso não seja inserido nenhum intervalo de nomes, ao clicar em "Mostrar", serão visualizadas todas as escolas que tenham alunos alocados. Este relatório apresenta o nome escola, seu código, o nome do bairro, a série e quantidade de alunos (bairro/série) e o total de alunos na escola.

c) Total de Alunos Matriculados nas Escolas em (Área de Influência x Séries) – Como no caso anterior, ao selecionar esta opção, os campos referentes ao "Nome de Aluno" estarão desabilitados. Sendo assim, a busca só poderá ser feita por intervalo de "Nome das Escolas". Ao clicar em "Mostrar", sem definir um intervalo, serão mostradas todas as escolas em que tenham alunos cadastrados. Este relatório apresenta as seguintes informações: nome da escola; código da escola; séries que possuem alunos; quantidade de alunos por cada série; a quantidade de alunos separados pelo critério dentro ou fora da área de influência da escola; e os totais de cada um dos itens.

6.4) Alunos não Matriculados – Ao clicar nesta opção, será aberta a janela "Relatório dos Alunos não Matriculados" (figura 30).

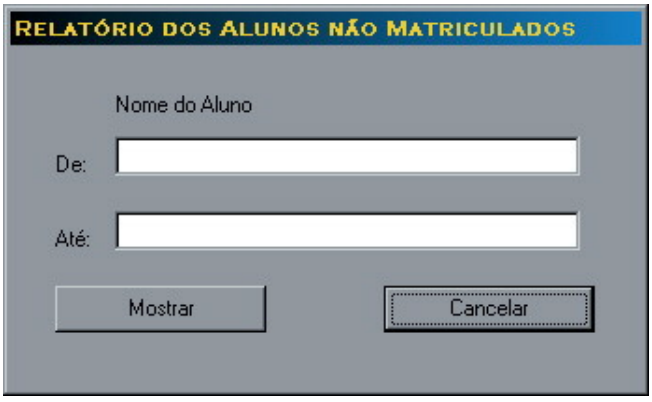


Figura 30 – Relatório dos Alunos não Matriculados

Caso o usuário não insira um intervalo de nomes nesta janela e clicar em "Mostrar", serão listados os cadastros de todos os alunos não matriculados.

7) Ajuda - A única opção associada a esse menu é *Sobre o* **MatriGeo**.

7.1) Sobre o MatriGeo – Neste submenu o usuário terá as informações pertinentes ao ano de criação do sistema, bem como seus desenvolvedores.

III.5 Comentários

É claro que mesmo com todas as funcionalidades descritas, será necessário que os usuários (trabalhadores vinculados à Secretaria Municipal de Educação) passem por um treinamento para que possam se familiarizar com o sistema. Este poderá ser ministrado para um grupo pequeno de pessoas e essas ficariam responsáveis por repassar o que for aprendido nas escolas (caso o treinamento seja realizado com um representante de cada escola), ou grupo de escolas (no caso de um representante para um pólo de ensino, que englobaria mais de uma escola).

É importante ressaltar que para o **MatriGeo** ser executado é necessário que na máquina de trabalho esteja instalado o MapObjects, e que neste primeiro momento, todo o conteúdo (*units*, *forms*, *shapes*, banco de dados, e demais dados utilizados no sistema) esteja no mesmo caminho em que ele foi elaborado (D:\mestrado\tese\Aplicativo).

CAPÍTULO IV

Implementação do Projeto

IV.1 Introdução

Neste capítulo será abordada a parte referente à implementação do Projeto. Serão descritas as *units* associadas a cada um dos *forms* e o algoritmo principal do sistema.

IV.2 Arquitetura do MatriGeo

A estrutura interna do **MatriGeo** foi composta por um total de trinta e uma *units*, três bancos de dados, onze arquivos *shapefile* e um arquivo *.dwg*.

O arquivo *dwg* chama-se **text_log_reg_austin**. Os arquivos *shapefiles* e de banco de dados serão descritos a seguir. Enquanto que as *units* e *forms* serão associadas a sete blocos principais, sendo listados posteriormente.

Na tabela 2 pode-se visualizar todos os temas que integram os mapas do **MatriGeo**. Cada tema possui um conjunto de arquivos, dos quais o *shapefile* refere-se ao tipo de feição (ponto, linha ou polígono) e o *.dbf* à tabela do arquivo. Os campos da mesma são descritos na coluna **Tabela**.

No caso dos temas rios, logradouros e ferrovia, todos foram gerados a partir da opção *intersect* do ArcView (ver capítulo 2). Por esta razão, eles apresentam o nome do arquivo mais o complemento *_inter*. Outro ponto em comum seria o fato dos três arquivos terem sido convertidos de um arquivo *.dwg* para um arquivo *shapefile*. Assim sendo, todos possuem o campo *Layer* que foi gerado automaticamente. O campo bairro

também consta nestes três temas, uma vez que o *intersect* foi feito entre o arquivo de cada um deles com o arquivo *Bairros*, que na ocasião do processamento só constava com tal campo.

Tabela 2 – Temas dos Mapas

Nome do Tema	Arquivo	Tipo	Tabela
Quadras de Localização	area.*	Shape (polígono)	ID, Unidade
Mapa Austin Colorido	austin.*	Shape (polígono)	Bairro
Mapa Austin Monocromático	austin_mc.*	Shape (polígono)	Bairro
Mapa Nova Iguaçu	Bairros.*	Shape (polígono)	Bairro, URG, Setores
Área de Influência da Escola	buff2.*	Shape (polígono)	Bufferdis, Escola
Escolas	escolas.*	Shape (ponto)	ID, X_Coord, Y_Coord, Nome, Rua, Num, Compl, Bairro, Cidade, UF, CEP, Tel, Diretor, Ano_Const, Num_Salas, Num_Cart, Esgoto, Água, Energia, Avaliação, Corrent_pe, Edu_Infant, Ens_funda, Ens_jovens, Salas_aloc
Ferrovia	ferrovia_inter.*	Shape (linha)	Layer, Bairro
Logradouros	lograd_inter.*	Shape (linha)	Layer, Bairro
Norte	norte.*	Shape (polígono)	ID
Norte	norteni.*	Shape (polígono)	ID
Rios	rios_inter.*	Shape (linha)	Layer, Bairro

Como já foi mencionado anteriormente o **MatriGeo** possui três banco de dados, que são: Matrigeo.mdb, Calendario.mdb e Logradouro.mdb. Destes, somente o primeiro tem um número maior de tabelas, seis no total (aluno, escola, professor, prof_esc,

prof_turma e turma). Os outros dois banco de dados possuem apenas uma tabela, respectivamente, calendário e log_unidade. Isto pode ser visualizado nas tabelas 3, 4 e 5.

Tabela 3 – Tabelas do Matrigeo.mdb

Nome da Tabela	Campos	Tipo do Campo
ALUNO	COD_ALUN, COD_TURM, COORD_X, COORD_Y, DISTANCIA	Número (Long Integer)
	NOME, RUA, NUM, COMPL, BAIRRO, CIDADE, ESTADO, CEP, TEL1, SERIE, PAI, MAE, TEL2, EMAIL,	Texto
	NASCIMENTO	Data
	AREA_INFLU	Sim/Não
ESCOLA	COD_ESC, NOME, RUA, NUM, COMPL, BAIRRO, CIDADE, TEL, ESTADO, CEP, DIRETOR, ANO_CONST, AVALIACAO, CORRENT_PEDA	Texto
	ID, NUM_SALAS, NUM_CART, SALAS_ALOC	Número (Long Integer)
	ESGOTO, AGUA, ENERGIA, EDU_INFANTIL, ENS_FUNDA, CRECHE, ENS_JOVENS_ADULT	Sim/Não
	COORD_X, COORD_Y	Número (Double)
	NUM_TURNOS	Número (Byte)
PROFESSOR	COD_PROF	Número (Long Integer)
	1SEGM, 2SEGM	Sim/Não
	NOME, RUA, NUM, COMPL, BAIRRO, CIDADE, TEL, ESTADO, CEP, E-MAIL	Texto
POF_ESC	Cod_Esc, Disciplina	Texto
	Cod_Prof, Max_Turmas, Turmas_Aloc	Número (Long Integer)
PROF_TURMA	Cod_Prof, Cod_Turma	Número (Long Integer)
TURMA	COD_TURM, CART_ALOC	Número (Long Integer)
	SERIE, COD_ESC, NOME	Texto
	ABERTA	Sim/Não

Tabela 4 – Tabela do Calendario.mdb

Nome da Tabela	Campos	Tipo do Campo
CALENDARIO	INICIO, FIM	Data
	DESCRICAO	Texto

Tabela 5 – Tabela do Logradouro. Mdb

Nome da Tabela	Campos	Tipo do Campo
Log_Unidade	UNIDADE, LOGRADOURO	Texto

Na figura 30 observa-se a interação entre o usuário e os diversos blocos (sete, sendo o primeiro o Menu Principal). Cada um dos blocos acessa um ou mais arquivos representados pelos cilindros na figura.

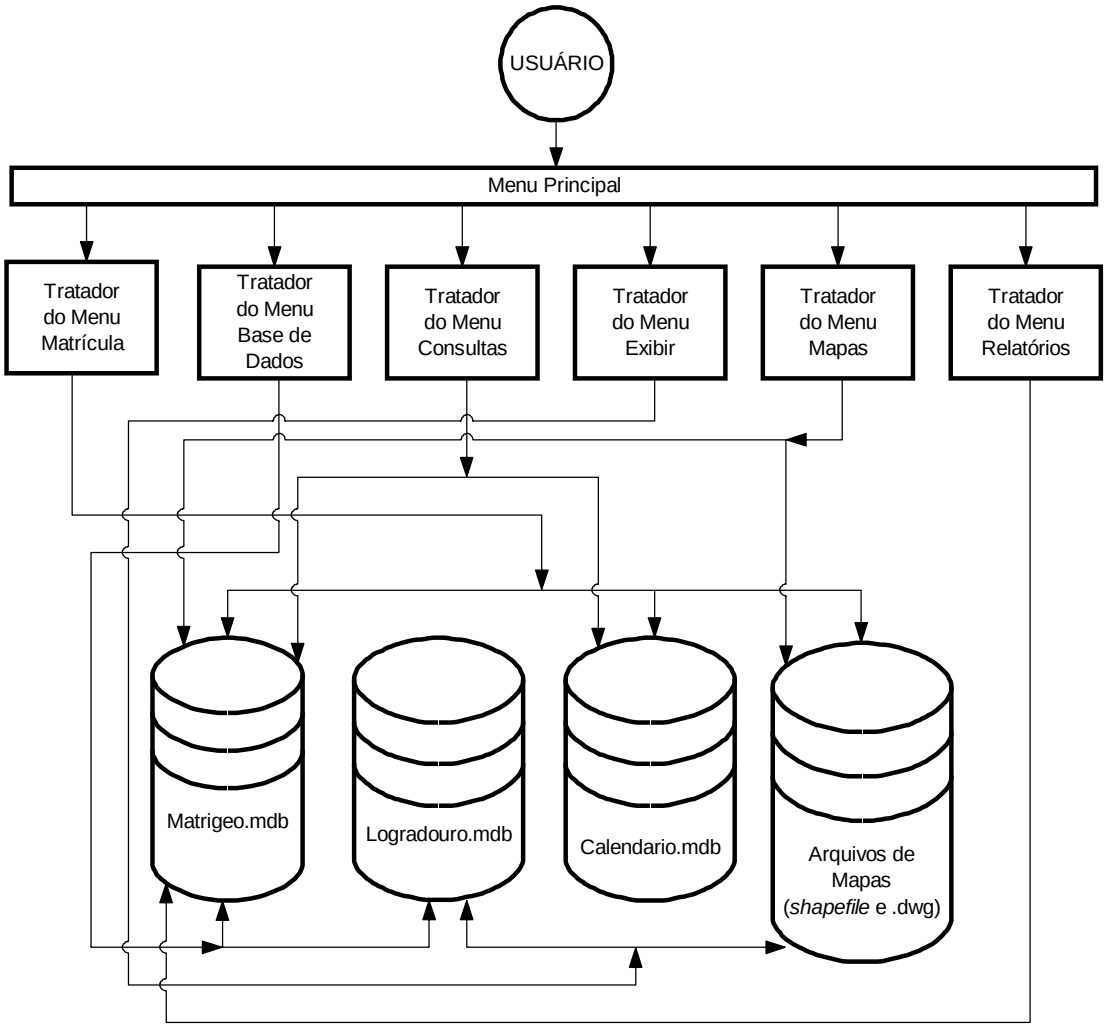


Figura 30 – Arquitetura do **MatriGeo**

Uma vez que todos os arquivos já foram detalhados anteriormente, a leitura da figura 30 se dará na seguinte forma: análise de cada um dos blocos, identificando-se as *units* e *forms* que serão utilizados por cada um deles, bem como os arquivos (dados pela trajetória das setas).

O Primeiro Bloco representa o menu principal do sistema, que será acionado pelo usuário, que por sua vez aciona os seis tratadores de menu.

Tabela 6 – *Unit* e *Forms* (Menu Principal)

UNIT	FORM
Interface_Usuario	Form1

Na tabela 6 temos a Interface_Usuario, que é a *Unit* responsável pela tela inicial do sistema, contendo menus, botões e uma área destinada à visualização de mapas e suas informações.

O Segundo Bloco representa o Tratador do Menu Matrícula, que acessa os arquivos Matrigeo.mdb, Calendario.mdb e os arquivos de mapa. As *Units* utilizadas serão listadas na tabela 7.

Tabela 7 – *Unit* e *Forms* (Tratador do Menu Matrícula)

UNIT	FORM
Interface_Usuario	Form1
DefCalen	DefinirCalendario
CadMatri	CadastrarMatricula
SelCad	SelecCad

Na tabela 7 temos as quatro *units* acessadas por este segundo bloco. A Interface_Usuario já foi descrita anteriormente. As demais são:

a) DefCalen – *Unit* responsável pela definição das datas iniciais e finais de cada evento: matrícula, férias etc. No *form* associado a essa *unit*, apenas o administrador do sistema poderá entrar com os valores referentes às datas, para evitar que um outro usuário sem permissão altere as datas do sistema.

b) CadMatri – *Unit* responsável pelo cadastramento das informações pessoais do aluno.

c) SelCad – *Unit* responsável pela seleção do cadastro de um determinado aluno, através de uma lista numa combobox de nomes que contenham o mesmo grupo de caracteres.

O Terceiro Bloco representa o Tratador do Menu Base de Dados, que acessa os arquivos Matrigeo.mdb e Logradouro.mdb. As *Units* utilizadas serão listadas na tabela 8.

Tabela 8 – *Unit* e *Forms* (Tratador do Menu Base de Dados)

UNIT	FORM
Interface_Usuario	Form1
UnitEscolaBD	FormEscolaBD
UnitLogradouroBD	FormLogradouroBD
UnitProfEscBD	FormProfEscBD
UnitProfessorBD	FormProfessorBD

Na tabela 8, temos as cinco *units* acessadas por este terceiro bloco. A Interface_Usuario já foi descrita anteriormente. As demais são:

a) UnitEscolaBD – *Unit* responsável pela visualização, inclusão, atualização e exclusão dos dados sobre as escolas na Base de Dados.

b) UnitLogradouroBD – *Unit* responsável pela visualização, inclusão, atualização e exclusão dos dados sobre os logradouros na Base de Dados.

c) UnitProfEscBD – *Unit* responsável pela visualização, inclusão, atualização e exclusão dos dados sobre a alocação dos professores nas escolas dentro da Base de Dados.

d) UnitProfessorBD – *Unit* responsável pela visualização, inclusão, atualização e exclusão dos dados sobre os professores na Base de Dados.

O Quarto Bloco representa o Tratador do Menu Consultas, que acessa os arquivos Matrigeo.mdb e Calendario.mdb. As *Units* utilizadas serão listadas na tabela 9.

Tabela 9 – *Unit* e *Forms* (Tratador do Menu Consultas)

UNIT	FORM
Interface_Usuario	Form1
ConsultAlun	ConsultaAluno
ConsultCalen	ConsultaCalendario
ConsultEsc	ConsultaEscola
ConsultProf	ConsultaProfessor
ConsultTurmProc	ConsultaTurmasProcessadas

Na tabela 9 temos as seis *units* acessadas por este quarto bloco. A Interface_Usuario já foi descrita anteriormente. As demais são:

a) ConsultAlun - *Unit* responsável pela consulta das informações pessoais do aluno.

b) ConsultCalen – *Unit* responsável pela consulta das informações referentes a um período do calendário. Atualmente o **MatriGeo** só apresenta a consulta ao período inicial e final do prazo para se proceder com as matrículas.

c) ConsultEsc – *Unit* responsável pela consulta das informações cadastrais referentes às escolas.

d) ConsultProf – *Unit* responsável pela consulta das informações cadastrais referentes aos professores. Possui algoritmos que, através do clique em um botão, faz uma listagem de itens que tenham correspondência a um conjunto de caracteres digitado pelo usuário em uma *combobox*.

e) ConsultTurmProc – *Unit* responsável pela consulta das informações referentes às turmas processadas.

O Quinto Bloco representa o Tratador do Menu Exibir, que acessa os arquivos Logradouro.mdb e arquivos de mapa. As *Units* utilizadas serão listadas na tabela 10.

Tabela 10 – *Unit* e *Forms* (Tratador do Menu Exibir)

UNIT	FORM
Interface_Usuario	Form1
SelCamadas	SelecCamadas
Selog	SelecLog

Na tabela 10, são apresentadas as três *units* acessadas por este quinto bloco. A Interface_Usuario já foi descrita anteriormente. As demais são:

a) SelCamadas – *Unit* responsável pela seleção das camadas que serão visualizadas no mapa de Setor de Planejamento Noroeste.

b) Selog – *Unit* responsável pela seleção de um logradouro, através de uma lista numa *combobox* de nomes que contenham o mesmo grupo de caracteres.

O Sexto Bloco representa o Tratador do Menu Mapas, que acessa os arquivos Matrigeo.mdb e arquivos de mapa. As *Units* utilizadas serão listadas na tabela 11.

Tabela 11 – *Unit* e *Forms* (Tratador do Menu Mapas)

UNIT	FORM
Interface_Usuario	Form1
SelEscSerie	SelecEscolaSerie
UnitDescEsc	DescritorEscola

Na tabela 11, mostram-se as três *units* acessadas por este sexto bloco. A Interface_Usuario já foi descrita anteriormente. As demais são:

a) SelEscSerie – *Unit* responsável pela seleção da escola e série a serem plotadas no mapa “alunos matriculados nas escolas”.

b) UnitDescEsc – *Unit* responsável pela descrição das escolas selecionadas no mapa “escolas”.

O Sétimo Bloco representa o Tratador do Menu Relatórios, que acessa somente os arquivos do Matrigeo.mdb. As *Units* utilizadas serão listadas na tabela 12.

Tabela 12 – *Unit* e *Forms* (Tratador do Menu Relatórios)

UNIT	**FORM**
RelAlunosNaoMatri	RelatorioAlunosNaoMatriculados
RelBD	RelatoriosBD
RelPreMatri	RelatoriosPreMatricula
RelTurProc	RelatoriosTurmasProc
UnitRelAluNaoMatri	QRMDFormAluNaoMatri
UnitRelBD1	QRMDFormBD1
UnitRelBD2	QRMDFormBD2
UnitRelBD3	QRMDFormBD3
UnitRelCadAluno1	QRMDFormCadAluno1
UnitRelCadAluno2	QRMDFormCadAluno2
UnitRelCadAluno3	QRMDFormCadAluno3
UnitRelTurProc1	QRMDFormTurProc1
UnitRelTurProc2	QRMDFormTurProc2
UnitRelTurProc3	QRMDFormTurProc3

Na tabela 12, são colocadas as quatorze *units* acessadas por este sétimo bloco. A Interface_Usuario já foi descrita anteriormente. As demais são:

a) RelAlunosNaoMatri – *Unit* responsável pela escolha de um conjunto de caracteres que indiquem uma listagem de nome de alunos a serem visualizados (campos De – Até) nos relatórios de Alunos não Matriculados.

b) RelBD – *Unit* responsável pela escolha de uma das opções para visualização dos relatórios da Base de Dados.

c) RelPreMatri – *Unit* responsável pela escolha de uma das opções para visualização dos relatórios da Pré Matrícula.

d) RelTurProc – *Unit* responsável pela escolha de uma das opções para visualização dos relatórios das Turmas Processadas.

e) UnitRelAluNaoMatri – *Unit* associada à visualização de alunos não matriculados em RelAlunosNaoMatri.

f) UnitRelBD1 – *Unit* associada à opção de visualização da quantidade de professores nas escolas em RelBD.

g) UnitRelBD2 – *Unit* associada à opção de visualização de professores cadastrados em RelBD.

h) UnitRelBD3 – *Unit* associada à opção de visualização de escolas cadastradas em RelBD.

i) UnitRelCadAluno1 – *Unit* associada à opção de visualização de alunos cadastrados em RelPreMatri.

j) UnitRelCadAluno2 – *Unit* associada à opção de visualização da quantidade de alunos, levando-se em conta sua idade x séries, em RelPreMatri.

l) UnitRelCadAluno3 – *Unit* associada à opção de visualização da quantidade de alunos, levando-se em conta seu bairro x séries, em RelPreMatri.

m) UnitRelTurProc1 – *Unit* associada à opção de visualização de alunos matriculados nas escolas em RelTurProc.

n) UnitRelTurProc2 – *Unit* associada à opção de visualização do total de alunos matriculados nas escolas, levando-se em conta seu bairro x séries, em RelTurProc.

o) UnitRelTurProc3 – *Unit* associada à opção de visualização do total de alunos matriculados nas escolas, levando-se em conta a área de influência da escola x séries, em RelTurProc.

IV.3 Estrutura de uma *Unit*

Todas as *units* e seus códigos são expostos no anexo A. Dentre todas, é a *Interface_Usuario* (anexo A, seção A.1.8) que pode ser considerada a *Unit* principal, pois nela se encontram os códigos que são o pilar do sistema.

Para fins de entendimento dos componentes de uma *unit*, foi escolhida a que possui seu código apresentado no anexo A, seção A.1.7 (**DefCalen**).

```
unit DefCalen;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Mask, Db, ADODB, Interface_Usuario;
```

```
type
  TDefinirCalendario = class(TForm)
    EditDataInicial: TMaskEdit;
    Combo_Descricao: TComboBox;
    EditDataFinal: TMaskEdit;
    Ndescricao: TLabel;
    Ninicial: TLabel;
    Nfinal: TLabel;
    Botao_Aplicar: TButton;
    Botao_Cancelar: TButton;
    procedure Botao_AplicarClick(Sender: TObject);
    procedure Botao_CancelarClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  DefinirCalendario: TDefinirCalendario;

implementation

{$R *.DFM}

procedure TDefinirCalendario.Botao_AplicarClick(Sender: TObject);

var
   inicio, fim : TDateTime;

begin
  if (EditDataInicial.Text = '  /  /    ') or
     (EditDataFinal.Text = '  /  /    ')
  then showmessage ('ERRO - Uma ou mais data(s) está(ão) em branco.')
  else if Combo_Descricao.Text = 'Matrícula'
       then begin
            inicio := StrToDate (EditDataInicial.Text);
            fim := StrToDate (EditDataFinal.Text);
            if fim >= inicio
            then begin
                 Form1.TabCal.first;
                 Form1.TabCal.Edit;
                 Form1.TabCal.FieldValues['INICIO'] := inicio;
                 Form1.TabCal.FieldValues['FIM'] := fim;
                 Form1.TabCal.Post;
                 Form1.Enabled := true;
                 Form1.SetFocus;
                 DefinirCalendario.Visible := false;
                end
            else showmessage ('Datas inconsistentes!');
           end;
end;

procedure TDefinirCalendario.Botao_CancelarClick(Sender: TObject);

begin
  Form1.Enabled := true;
  Form1.SetFocus;
  DefinirCalendario.Visible := false;
end;

end.
```

Os elementos que constituem esta *unit* são:

a) *Unit <nome da unit>*, que vem logo na primeira linha, indicando o nome da *unit*, (unit DefCalen).

b) *Interface*, ela delimita a seção de *interface* onde serão colocadas as definições, tipos e variáveis que poderão ser vistas por outras *units*.

c) Cláusula *Uses*, indica quais as *units* que deverão ser lidas para poder complementar a atual. Por exemplo, toda *unit* ao usar um *Form*, precisa declarar

a localização da classe *TForm*. Isto ocorre indicando nesta cláusula a *unit Forms*. Na *unit DefCalen* foi usada uma tabela que foi implementada na *unit Interface_Usuario*. Por isso, esta última deve ser declarada na cláusula *Uses* da *unit DefCalen*.

d) *Type*, é a definição dos tipos do programa. Estes tipos podem ser utilizados por qualquer *unit* na aplicação que inclua o nome da *unit* atual na sua cláusula *Uses*. Os tipos utilizados em *DelfCalen* são: *TForm*, *TMaskEdit*, *TComboBox*, *TLabel* e *TButton*.

e) *Var*, é a definição das variáveis globais. Estas variáveis são definidas da seguinte forma: *nomevariavel : tipovariavel*, onde *nomevariavel* é o nome da variável global e *tipovariavel* é o tipo desta variável. Por exemplo, *Combo_Descricao: TComboBox*; *Combo_Descrição* é o nome de uma variável, de tipo *TComboBox*.

f) *Implementation*, delimita a segunda seção de uma *unit*. É nesta parte da *unit* em que são colocadas as funções e também variáveis que serão acessadas somente por ela. Esta seção pode conter também uma cláusula *Uses*.

g) *End*, ao final da *unit* é colocado uma linha com essa descrição. Tudo que for colocado após esta linha é desprezado, não sendo considerado como parte do programa.

h) *Initialization* e *finalization*, estas seções são opcionais e contêm comandos que são executados quando a aplicação é carregada.

A programação é feita entre um *begin* inicial e um *end* final. Pode parecer redundante esta colocação, mas na verdade é que entre estes dois termos citados, podem haver outros *begins* e *ends*. Durante o decorrer do código, pode-se incluir algum tipo de comentário, que não interfere na programação. Esses comentários servem para definir alguma passagem do código, a fim de deixar claro o que o desenvolvedor do programa quis dizer, tornando-o de fácil entendimento não só para outra pessoa que venha a trabalhar no programa, como para ele mesmo. Se a opção de comentário for apenas uma linha deve-se usar duas barras (//), e caso seja de interesse deixar como comentário várias linhas do programa, deve-se usar chaves (**{}**), uma no início e outra no final da parte do código que ficará como comentário. Além do uso já descrito, o comentário também pode ser usado para deixar registrado, por exemplo, uma outra forma de execução de uma passagem do código, ou mesmo destacar alguma passagem que não

foi totalmente implementada, a fim de evitar problemas quando rodar a parte do programa que ainda não foi implementada.

IV.4 O Algoritmo Principal

O **MatriGeo** foi estruturado em torno de um algoritmo principal, apesar de contar com vários códigos responsáveis pelo seu funcionamento. Como foi visto anteriormente, cada *unit* foi composta por um ou mais conjuntos de códigos (de implementação interna ou externa, podendo ser acessado pelas demais *units*).

A partir do banco de dados do projeto, torna-se possível implementar o algoritmo principal do aplicativo que irá alocar os alunos nas vagas existentes na rede municipal de escolas, de tal forma que esses alunos fiquem o mais próximo possível de suas localizações. Tal algoritmo pode ser descrito com o seguinte pseudo-código:

1. ***Fazer leitura das coordenadas da residência do aluno (Rxy), através de um clique do mouse na posição do mapa onde se localiza o seu logradouro e criar lista de escolas inválidas vazia (ListEinvalidas ← nil).***
2. ***Achar a escola (E) mais próxima de Rxy, consultando a localização de todas as escolas no banco de dados do projeto (BDproj) com exceção daquelas em ListEinvalidas.***
3. ***Caso a escola E possua vaga para o aluno, então efetuar pré-matrícula e FIM.***
4. ***Caso a escola E não possua vaga para o aluno, fazer ListEinvalidas ← ListEinvalidas + E, vá para o passo 2.***

No algoritmo, pode-se observar que existe uma prioridade ou critério de alocação de alunos segundo a ordem de chegada ou inclusão no sistema. Seria possível alocar os alunos por um outro critério qualquer, como, por exemplo, priorizar sempre a alocação do aluno em uma escola absolutamente mais próxima. Suponhamos que um aluno *A1* não encontre vaga na escola *E1* mais próxima, mas em *E1* foi alocado anteriormente um outro aluno *A2* mais distante, então o sistema poderia retirar *A2* de *E1* para colocar *A1*, pois *A1* teria prioridade sobre as vagas de *E1*. Nesse caso o aluno *A2* alocado inicialmente em *E1* porque não obteve vaga na escola de fato mais próxima de si, ficaria mais longe ainda, pois ele seria realocado para uma outra escola.

O prosseguimento de um processo de realocações poderia levar o suposto aluno *A2* a ficar cada vez mais distante e, no pior caso, ficar sem vaga depois de ter feito a matrícula dentro do prazo e com bastante antecedência. Para se evitar tais injustiças, preferiu-se manter o algoritmo principal do sistema como foi exposto acima e, para se

evitar totalmente o transtorno de filas como comentado na introdução, a idéia é futuramente colocar este projeto sendo acessado pela Internet.

O algoritmo será melhor entendido levando-se em conta o significado de cada passagem do código. Já foi visto que nele prioriza-se a alocação dos alunos pela entrada no sistema. As variáveis do algoritmo são: *Rxy* (coordenadas da residência do aluno); *ListEinvalidas* (listas de escolas inválidas vazias); *E* (escola mais próxima a *Rxy*) e *BDproj* (localização de todas as escolas no banco de dados do projeto com exceção daquelas em *ListEinvalidas*). Entretanto deve-se especificar o que o sistema entende por *vaga para o aluno* e *turma para o aluno*:

a) Definição de vaga para o aluno:

1- Se a escola não funcionar com o segmento do aluno, **não há vaga**.

2- Se a escola não tiver turma para o aluno, **não há vaga**.

b) Definição de turma para o aluno:

1- Se existir uma turma **T** aberta (ainda há vagas na turma) da série do aluno, o aluno é alocado em **T** e FIM.

2- Se a escola possui ainda turma física vazia (total de turmas vazias = (número_de_salas x número_de _turnos) – turmas já alocadas), então criar turma **Tc** candidata para o aluno.

3 - Se a escola possuir professor para a **Tc**, o aluno é alocado em **Tc** e FIM. Senão, **não há vaga** para o aluno.

No capítulo 3, já foi definido o que significa primeiro e segundo segmentos para o sistema **MatriGeo**. Neste caso, para definir se há vaga em uma determinada escola para um aluno, deve-se verificar em qual segmento ele estará se cadastrando. Se o educando estiver se cadastrando em uma turma de 5ª. Série, somente as escolas que possuam o segundo segmento serão escolas válidas, mesmo que estas não sejam as mais próximas da residência do aluno.

Uma turma física é dada pela relação do total de salas existentes em uma escola, multiplicado por dois (equivalendo aos dois turnos de atuação do município que são o da manhã e o da tarde), com exceção das creches, que são de tempo integral (um único turno). Entretanto, é necessário fazer uma nova operação de multiplicação, dada pelo total de carteiras multiplicado pelo valor resultante em turma física (definida acima). E por fim, diminuir o resultado encontrado do total de turmas alocadas.

Para o total entendimento deste algoritmo, resta definir o que é professor para turma. No caso de primeiro segmento, para cada turma, o código deverá verificar apenas

se tem um professor para alocar uma nova turma. Mas no caso de segundo segmento, além de verificar se existem professores de todas as disciplinas obrigatórias (português, inglês, matemática, ciências naturais, história, geografia, educação física e artes) será necessário também, pesquisar no banco, o máximo de turmas a que cada professor poderá ser alocado.

IV.5 Comentários

Este capítulo serviu para orientar, de forma bem resumida, os passos tomados para o desenvolvimento do sistema **MatriGeo**.

Foram indicados: a arquitetura interna do sistema (arquivos utilizados, *units* e *forms*), a estrutura de uma *unit*; bem como a descrição do algoritmo principal. Entretanto, para maiores esclarecimentos no que se refere ao código fonte do sistema, deve-se consultar o anexo A.

A validação da implementação foi feita através de alguns testes do sistema, que serão analisados no próximo capítulo.

CAPÍTULO V

Testes e Resultados

V.1 Introdução

Os testes realizados demonstram que o **MatriGeo** está atendendo ao seu objetivo principal, que é de estar alocando um aluno em uma escola mais próxima, levando-se em conta sua localização como um dos fatores mais importantes, sem deixar de considerar outros fatores, que serão abordados nos testes.

V.2 Testes e Resultados Esperados

Os dois primeiros testes foram elaborados com uma gama bem reduzida de dados na tabela aluno; o terceiro é um teste para verificar a eficácia de um aplicativo criado para entrada de dados no sistema, chamado de **Gera Teste para MatriGeo**; e os quatro últimos foram feitos a partir dos dados gerados por este aplicativo.

V.2.1 Teste Número 1

Desde o início do projeto, quando começaram a ser implementadas as primeiras funções ligadas ao banco de dados, o **MatriGeo** apresenta um leque bem pequeno de dados, no que se refere à tabela Aluno (em matrigeo.mdb).

Os dados dos alunos foram criados hipoteticamente. Já os dados das escolas (na área de estudo são catorze escolas, entretanto por falta do encaminhamento dos formulários referentes a todas, apenas onze foram localizadas no mapa) foram preenchidos com informações reais, coletadas através do formulário mostrado no anexo B (seção B.1.1). Por fim, os dados referentes aos professores foram preenchidos com base no formulário apresentado no anexo B (seção B.1.2). Entretanto como algumas das escolas não enviaram os formulários de professores, a tabela foi preenchida "alocando" professores de outras escolas. Com o formulário das escolas, foram observadas as séries e a quantidade de turmas de cada uma, assim foi possível estimar a quantidade de professores para cada escola.

Para este primeiro teste foram usados esses poucos dados, um total de seis registros na tabela aluno e os dados totalmente preenchidos das escolas e professores.

Neste teste, observa-se que todos os seis alunos registrados foram alocados nas escolas efetivamente mais próximas de sua residência (ver anexo C, seção C.1.1). Observa-se também que apenas um dos alunos encontra-se localizado fora da área de influência da escola. Em um primeiro momento poderia-se dizer que o aluno Gabriel Carlos Soares não foi corretamente localizado. Sua residência está há mais de 3 Km da escola em que foi alocado. Isto acontece, pois com apenas duas creches na região, esse aluno acaba por ser alocado em uma escola muito distante. Mas o **MatriGeo** atende aos seus objetivos, pois entre esta creche e a outra existente na região, a Creche Municipal de Austin é de fato, a mais próxima do aluno.

Este fato levanta a importância de uma avaliação por parte da prefeitura, para saber onde ocorre falta de um certo tipo de edificação escolar. Isto é trabalho para a área de Planejamento Urbano e Educacional.

O **MatriGeo** não prioriza a determinação de um certo número de turmas por série, como é feito atualmente para determinar se hà vaga ou não para um aluno em uma determinada escola. Ele prioriza a menor distância entre residência e escola, podendo uma ter, por exemplo, somente turmas de 5ª. Série. Tudo depende da demanda de certa região. Entretanto o sistema divide as instalações escolares já existentes em escolas e creches, ou seja, uma creche atualmente continuará sendo somente creche para o **MatriGeo**. A única diferença estará por parte das escolas, que ao invés de terem um certo número de turma por série, terão quantas turmas de uma série forem necessárias para cumprir a demanda.

Assim sendo, tanto no sistema de matrículas atual, quanto no proposto nesta dissertação, um aluno de creche, que não more nas proximidades das duas únicas

creches, fatalmente terá que se deslocar a uma distância muito grande. A construção de uma nova unidade escolar do tipo creche se faz necessária nesta região de estudo.

V.2.2 Teste Número 2

Neste segundo teste, a quantidade de alunos continua bem pequena, quase inalterada em relação ao primeiro. Apenas um registro foi adicionado ao banco de dados. Este registro diz respeito à Gabriela Carlos Soares, que pode ser visualizado no anexo C, seção C.1.2. O que difere nestes dois registros, além do nome, são a sua data de nascimento e a série que foram cadastradas.

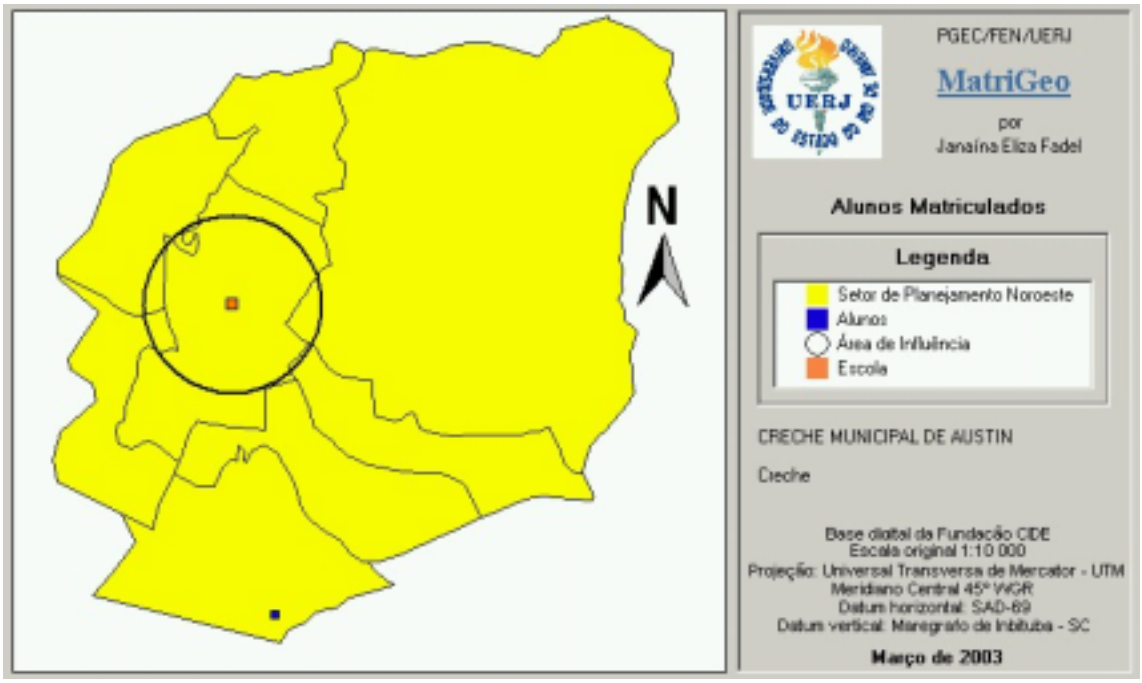


Figura 31 – Gabriel localizado em uma creche

Este registro foi inserido para provar que um mesmo ponto (representando um aluno) pode ser alocado em escolas diferentes por conta de sua série (figuras 31 e 32). Um aluno é alocado sim em uma escola mais próxima de sua residência, mas outros fatores também são determinantes para esta escolha por parte do sistema.

Os fatores são: se a escola oferece a mesma série em que o aluno estará se inscrevendo, primeiro ou segundo segmento (no caso do primeiro segmento, ainda tem

que saber se é creche ou não); se há turmas e se há professores para ministrar aulas em uma quantidade determinada de turmas.

Este último fator é determinado pelo máximo de turmas possíveis para um segmento, levando-se em conta em quantas turmas um professor poderá lecionar, o que poderá ser visto no anexo C, seção C.1.3.

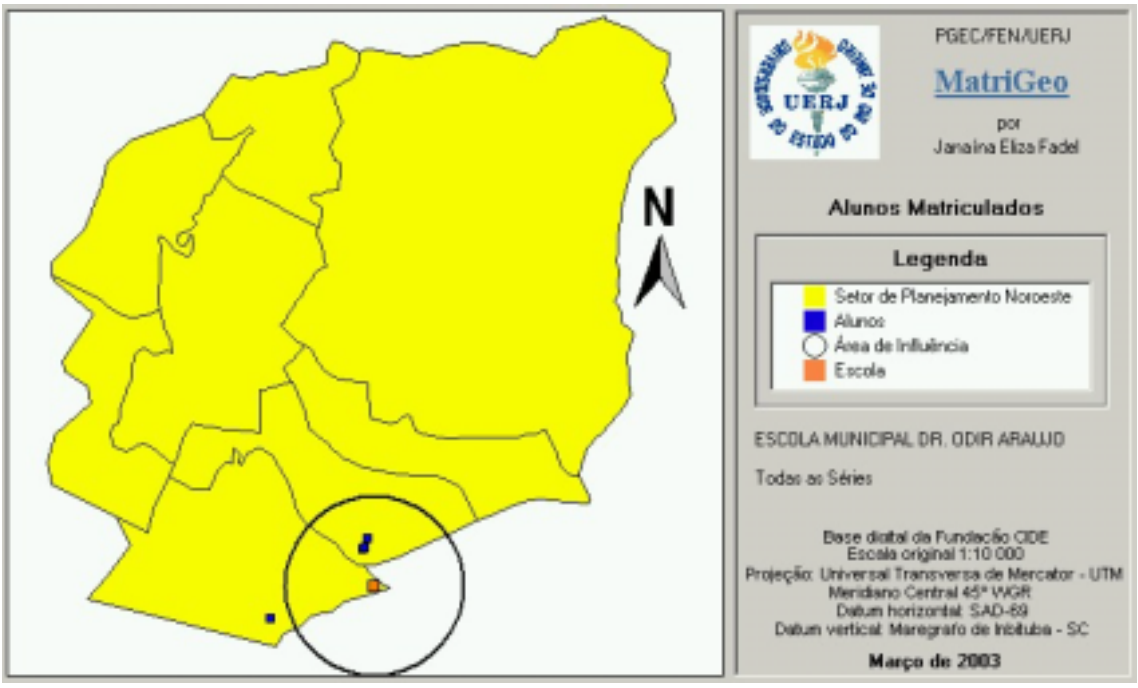


Figura 32 – Gabriela localizada em uma escola

V.2.3 Teste Número 3

O teste número 3 é a verificação do aplicativo elaborado para entrada de dados na tabela aluno, para poder fazer os demais testes, com uma gama bem maior de informações.

O nome do aplicativo é **Gera Teste para MatriGeo** (figura 33). Ele gera uma quantidade determinada de registros no banco de dados, que é inserida no campo "Quantidade de Alunos". Também pode-se entrar com os dados da coordenada do aluno, sua data de nascimento e a série. Podem ser tomados dois procedimentos: clicar em "Gerar Cadastro de Alunos" ou clicar em "Apagar Cadastro de Alunos".

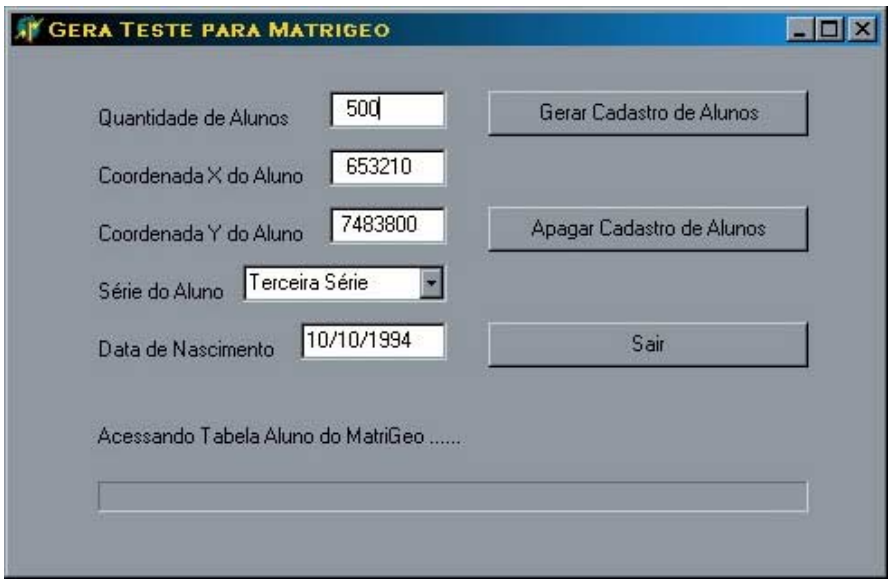


Figura 33 – Aplicativo **Gera Teste para MatriGeo**

Se optar pelo primeiro procedimento, toda vez que entrar com uma certa quantidade de alunos estes serão cadastrados, e quando quiser inserir mais um grupo, o aplicativo irá gerar novos registros começando com o código de aluno subseqüente ao último maior registrado. O nome do aluno será *aluno + o código do aluno*. Por exemplo, se forem inseridos 300 registros, o próximo registro a ser inserido será de número 301 e o nome do aluno será: aluno 301.

No caso de se optar pelo segundo procedimento, todo o cadastro de aluno será apagado.

No anexo C, seção C.1.4, pode-se verificar parte dos 500 novos registros. Note que, por ter sido clicado anteriormente no botão “Apagar Cadastro de Alunos”, todos os alunos cadastrados anteriormente (anexo C, seções C.1.1 e C1.2) foram apagados.

V.2.4 Teste Número 4

Uma vez verificada a eficácia do sistema de geração de cadastro de alunos para ser utilizado no **MatriGeo**, o próximo teste dirá respeito à análise da distribuição nas escolas dos 500 alunos gerados.

Visualmente este não é um resultado relevante a se colocar em forma de mapa, uma vez que todos os 500 alunos estarão localizados em um mesmo ponto. Mas a análise dos relatórios será muito importante.

Após o processamento das turmas, verifica-se que a escola mais próxima não comporta todos os 500 alunos. Apenas 245 alunos são alocados nela, enquanto que os outros 255 alunos foram alocados na segunda mais próxima. Esta distribuição pode ser vista no anexo C, seção C.1.5.

Como já foi comentado anteriormente, o sistema irá gerar turmas, sem levar em conta um número determinado das mesmas por série. Isto também pode ser visualizado no anexo mencionado acima. Nota-se que todos os 500 alunos estudam na mesma série. Assim, uma escola inteira irá possuir apenas turmas desta série.

Este é apenas um ponto importante que os relatórios podem nos demonstrar. Outro diz respeito à relação da quantidade física de uma escola e a quantidade efetivamente preenchida. A quantidade física é dada pelo número total de salas multiplicado pelo número máximo de carteiras, e no caso de escolas este valor ainda deverá ser multiplicado por 2 (equivalendo aos dois turnos).

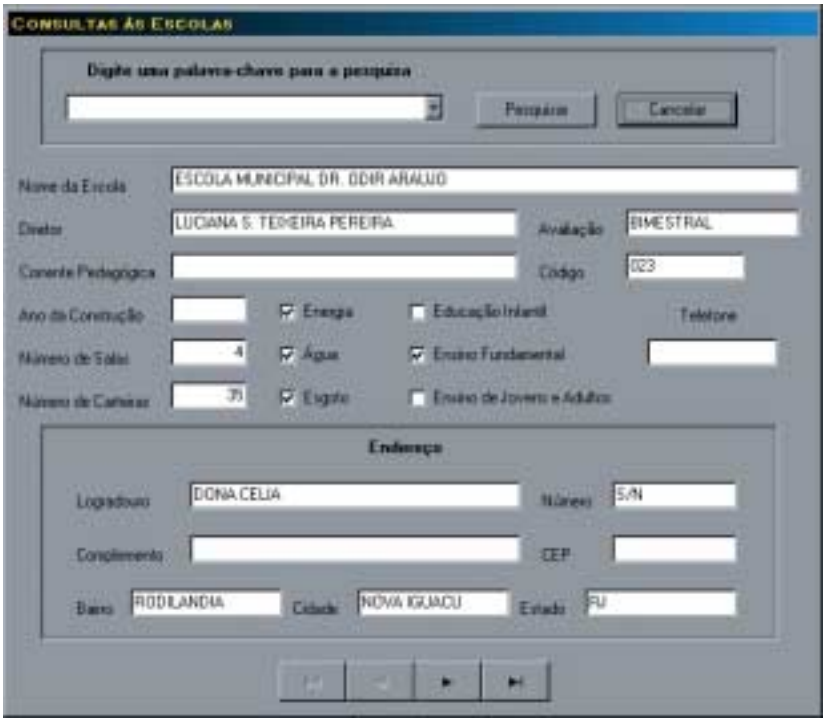


Figura 34 – Dados Escola Municipal Dr. Odir Araújo

Como pode ser visto na figura 34, a Escola Municipal Dr. Odir Araújo possui 4 salas de aula, com capacidade de 35 carteiras. Isto daria um total de 140 vagas por turno, tendo uma capacidade total da escola nos dois turnos de 280 vagas. A escola trabalha com ensino fundamental, mas possui apenas professores de primeiro segmento (ver anexo C, seção C.1.3). Este fator já determina que para esta escola somente serão cadastrados alunos de primeiro segmento. Caso os alunos gerados no teste anterior

fossem de uma oitava série, por exemplo, nenhum deles seria alocado na escola em questão.

O máximo de alunos nesta escola, como foi visto, será de 280, levando-se em conta apenas o quantitativo de salas versus carteiras. Entretanto, na escola foram alocados apenas 245 alunos.

Por que isto ocorreu? A resposta só poderá ser considerada simples de ser dada, se forem analisados os relatórios. Verificando novamente o anexo C, seção 1.3, observa-se que a quantidade total na escola é de 7 de professores. Levando-se em conta que todos são professores de primeiro segmento, cada um só poderá assumir uma turma nesta escola.

Para a Escola Municipal Dr. Odir Araújo trabalhar com um total de vagas compatível com seu potencial quantitativo (salas x carteiras), será necessário que a prefeitura contrate mais um professor de primeiro segmento.

V.2.5 Teste Número 5

Para a realização deste teste, foram criados setenta e cinco alunos, sendo que os sessenta primeiros encontram-se na área de influência da Creche Municipal Vila São Miguel e os quinze últimos na área de influência da Creche Municipal de Austin.

O que se espera com este teste?

Espera-se que os sessenta primeiros alunos sejam alocados nas creches antes dos quinze últimos, mesmo que estes estejam localizados na área de influência de uma delas. Esta hipótese é levantada para comprovar que o sistema aloca um aluno não somente pelo fator localização mais próxima, mas levando-se em conta também a entrada do mesmo no **MatriGeo**.

Qual foi o resultado?

A hipótese levantada só poderá ser comprovada analisando o anexo C, seção C.1.6, onde será possível visualizar o nome do aluno e a creche na qual ele foi alocado. Lembre-se que os registros foram criados a partir do **Gera Teste para MatriGeo**, e que o nome dos alunos é igual ao código de entrada no sistema.

Logo, para comprovar se o resultado esperado foi alcançado, basta verificar onde foram alocados os primeiros sessenta alunos (do aluno 01 ao aluno 60).

No anexo mencionado logo acima, observa-se que os sessenta primeiros alunos ficaram assim distribuídos:

a) Aluno 01 ao Aluno 30 – Creche Municipal Vila São Miguel.

b) Aluno 31 ao Aluno 60 – Creche Municipal de Austin.

Como a Creche Municipal de Austin possui quarenta e cinco vagas, as trinta primeiras foram preenchidas pelos alunos 31 ao 60 e as quinze últimas vagas pelos alunos 61 ao 75.

Assim sendo, comprova-se a hipótese. Realmente o **MatriGeo** estará sempre alocando um aluno em uma escola mais próxima, mas levando-se sempre em conta a entrada do aluno no sistema.

Neste teste, caso fossem incluídos mais registros antes dos quinze últimos (localizados na área de influência da Creche Municipal de Austin), fatalmente alguns (se fossem incluídos menos de 15 registros) ou todos os alunos (se fossem 15 registros ou mais), ficariam sem creche, mesmo morando na área de influência.

V.2.6 Teste Número 6

Para a realização deste teste, foram criados duzentos e quarenta e cinco alunos, todos localizados em um mesmo ponto, dentro da área de influência da Escola Municipal Althair Pimenta de Moraes. Todos os alunos foram cadastrados para uma turma de 5ª. Série.

O que se espera com este teste?

A referida escola possui um total de seiscentas e trinta vagas, distribuídas entre primeiro e segundo segmentos. O máximo para segundo segmento será de seis turmas (ver anexo C, seção C.1.3). Existem professores que assumem as seis turmas (no caso de haver apenas um professor para certa disciplina) ou três turmas (no caso das disciplinas em que existam dois professores).

Espera-se que mesmo a escola tendo um total de vagas (630) maior que as vagas pleiteadas para 5ª. Série (245), falte vaga para um total de 35 alunos, sendo necessário abrir uma turma desta série em uma outra escola que também ofereça turmas de segundo segmento. A escola só poderá oferecer um máximo de duzentas e dez vagas para o segundo segmento, independente da série pleiteada. Este total é dado pelo

máximo de turmas que um professor poderá ministrar aulas (no caso 6) multiplicado pelo número de carteiras de uma sala (35).

Qual foi o resultado?

Os primeiros duzentos e dez alunos foram realmente alocados na escola em questão e os demais alocados em uma turma na Escola Municipal Walfredo da Silva, como pode ser observado no anexo C, seção C.1.7.

Assim sendo, mais uma hipótese ficou comprovada. Além da localização, ordem de entrada no sistema e a possibilidade de uma escola ter turmas apenas de uma mesma série; uma escola mesmo com um número total de vagas grande poderá não alocar alunos se as vagas não forem para o segmento no qual o aluno pretende estudar.

V.2.7 Teste Número 7

Este teste refere-se aos alunos não matriculados em nenhuma escola, o que pode ser visto no anexo C, seção C.1.8.

Foram criados noventa e cinco registros, onde os primeiros setenta alunos foram localizados em uma coordenada dentro da área de influência da Creche Municipal Vila São Miguel e os últimos vinte e cinco alunos localizados em uma coordenada dentro da área de influência da Creche Municipal de Austin.

Isto pode ser visto no anexo C, seção 1.9, onde se verifica que todos os trinta alunos da Creche Municipal Vila São Miguel estão localizados dentro da área de influência. Enquanto que dos quarenta e cinco cadastrados na Creche Municipal de Austin, apenas cinco se encontram na área de influência.

Os demais vinte alunos ficaram sem serem alocados em nenhuma das creches por ter excedido o valor máximo de vagas oferecidas por estas, o que identifica uma escassez de unidade escolar que ofereça a série creche.

Como foi mencionado no teste número 5, mesmo que os alunos estejam dentro da área de influência de uma determinada escola, ou creche, a sua entrada no sistema é que vai ser mais prioritária. Naquele teste tinham sido incluídos sessenta alunos em um ponto e estes preencheram uma das creches e os quinze últimos foram alocados na Creche Municipal de Austin. Comentou-se que dependendo do número de cadastros incluídos antes dos alunos localizados na área de influência desta última creche, fatalmente alguns ou todos os alunos ficariam sem matrícula. No teste atual foram

incluídos mais dez registros, sendo assim, apenas cinco dos quinze alunos matriculados anteriormente na Creche Municipal de Austin que conseguiram uma vaga, os outros dez alunos ficaram sem vaga.

V.2.8 Outros Resultados

Nesta seção será abordado um resultado muito interessante que poderá ser conferido analisando um dos relatórios já apresentado associado ao teste número 2. Neste teste levanta-se a relevância de um dos fatores para alocação dos alunos nas escolas, que é o número de professores disponíveis para ser criada uma turma.

No anexo C, seção 1.3 tem-se os seguintes campos: nome da escola, quantidade de salas e de turnos, o código da escola, quantidade de professores por disciplina, quantidade máxima de turmas e total de turmas possíveis para primeiro e segundo segmento.

A quantidade máxima de turmas constitui em um somatório de todos os professores por disciplina. Por exemplo: o valor apresentado sendo o número seis, não quer dizer que cada professor de dada disciplina poderá ministrar aulas em seis turmas, a menos que a quantidade de professor seja igual a 1. Em outras palavras, para a disciplina inglês, constando apenas um professor e a quantidade máxima de turmas igual a seis, quer dizer que apenas um professor será responsável por seis turmas. Mas se este total também é seis, mas a quantidade de professores for dois ou três, poderá indicar que cada um separadamente seja responsável por três ou duas turmas, mas no relatório será visualizado o número seis.

Emprega-se o termo total de turmas possíveis para primeiro ou segundo segmento, pois não basta uma escola ter um número de professores disponíveis, mas também salas suficientes.

Serão analisadas caso a caso, as escolas registradas no anexo mencionado anteriormente.

a) Creche Municipal de Austin – A quantidade possível de turmas equivale à capacidade de salas multiplicadas pelo turno (apenas um turno para creche). Não foram verificadas falta de professores ou de salas (espaço físico da escola).

b) Creche Municipal Vila São Miguel – A quantidade possível de turmas equivale à capacidade de salas multiplicadas pelo turno (apenas um turno para creche). Não foram verificadas falta de professores ou de salas (espaço físico da escola).

c) Escola Municipal Althair Pimenta de Moraes – nove salas multiplicadas por dois turnos equivale a um total de 18 turmas. Ao ser verificado o número possível de turmas para os dois segmentos, tem-se um total de quinze turmas (nove para o primeiro segmento e seis para o segundo segmento). Para esta escola observa-se uma falta de professores, para poder ter as dezoito turmas possíveis. Existe a mesma quantidade de professores para segundo segmento (seis). Para fins de Planejamento, pode-se dizer que para esta escola trabalhar com seu efetivo físico, seria necessário contratar mais três professores de primeiro segmento, ou então pelo menos mais um professor de cada disciplina de segundo segmento, ficando este único professor (por disciplina) responsável pelas três novas turmas de tal segmento.

d) Escola Municipal Dr. Odir Araújo – quatro salas multiplicadas por dois turnos equivale a um total de 8 turmas. Esta escola trabalha apenas com o primeiro segmento. Ao ser verificado o número possível de turmas, tem-se um total de sete turmas. Para esta escola observa-se uma falta de apenas um professor, para poder ter as oito turmas possíveis. Para fins de Planejamento, pode-se dizer que para esta escola trabalhar com seu efetivo físico, seria necessário contratar mais um professor de primeiro segmento.

e) Escola Municipal José Luiz da Silva – nove salas multiplicadas por dois turnos equivale a um total de 18 turmas. Esta escola trabalha apenas com o primeiro segmento. Ao ser verificado o número possível de turmas para este segmento, tem-se um total de dezoito turmas. Para esta escola observa-se uma sobra de professores, já que ela possui vinte professores, mas capacidade para apenas dezoito turmas. Para fins de Planejamento, pode-se dizer que para esta escola trabalhar com seu efetivo físico, seria necessário deslocar estes dois professores excedentes para uma outra escola, ou os mesmos ficariam sem trabalhar.

f) Escola Municipal Nena Rodrigues – cinco salas multiplicadas por dois turnos equivale a um total de 10 turmas. Esta escola trabalha apenas com o primeiro segmento. Ao ser verificado o número possível de turmas para este segmento, tem-se um total de dez turmas. Esta escola encontra-se trabalhando com o seu total físico, uma vez que a escola possui dez professores cadastrados. Assim, cada um destes dez professores irá assumir uma única turma, das dez possíveis.

g) Escola Municipal Prof. Enilza Barros dos Santos Chiconelli – dezesseis salas multiplicadas por dois turnos equivale a um total de 32 turmas. Ao ser verificado o

número possível de turmas para os dois segmentos, tem-se um total de vinte e uma turmas (vinte e uma para o primeiro segmento e zero para o segundo segmento). Nesta escola nota-se um problema. Para as disciplinas matemática e português, o número máximo de turmas é maior que nas outras disciplinas (onze), respectivamente dezessete e doze turmas. A princípio poderia ser alocado o total físico da escola (vinte e uma turmas de primeiro segmento somado a onze de segundo segmento, daria um total de trinta e dois). Entretanto, as turmas de segundo segmento só serão criadas se houver pelo menos um professor de cada disciplina. Para esta escola percebe-se uma falta de professor de ciências, assim sendo, nenhuma turma poderá ser criada. Para fins de Planejamento, pode-se dizer que para esta escola trabalhar com seu efetivo físico, seria necessário contratar no mínimo mais três professores de ciências, para poder ter as onze turmas. Este valor igual a três no mínimo é dado pelo máximo de turmas que um professor desta disciplina poderá possuir, que é quatro. Pelo levantamento feito através dos dados fornecidos via formulários de professores (anexo B, seção B.1.2) das oito disciplinas obrigatórias apenas educação física, artes e inglês constam como o máximo de turmas igual a seis, as demais disciplinas variam entre três a quatro turmas.

h) Escola Municipal Prof. Marcio Caulino Soares – treze salas multiplicadas por dois turnos equivale a um total de 26 turmas. Ao ser verificado o número possível de turmas para os dois segmentos, tem-se um total de dezesseis turmas (dez para o primeiro segmento e seis para o segundo segmento). Para esta escola observa-se uma falta de professores, para poder ter as vinte e seis turmas possíveis. Observa-se que para a maioria das disciplinas do segundo segmento tem-se um total de quinze turmas, diferente apenas para matemática (dezoito) e inglês (seis). Para fins de Planejamento, pode-se dizer que para esta escola trabalhar com seu efetivo físico, seria necessário contratar mais professores de inglês. Mesmo que a disciplina inglês também tivesse um máximo possível de quinze turmas como a maioria das demais disciplinas, mesmo assim sobraria uma sala, onde poderia ser alocada mais uma turma (levando-se em conta as dez turmas de primeiro segmento somando as quinze possíveis de segundo segmento).

i) Escola Municipal Ruy Barbosa – três salas multiplicadas por dois turnos equivale a um total de 6 turmas. Esta escola trabalha apenas com o primeiro segmento. Ao ser verificado o número possível de turmas para este segmento, tem-se um total de seis turmas. Nesta escola existe um grande excedente de professores. Existem vinte e um cadastrados, mas a escola só comporta um máximo de seis turmas. Sendo assim,

quinze professores cadastrados para esta escola deveriam ser realocados em outras unidades escolares.

j) Escola Municipal Souza e Mello – seis salas multiplicadas por dois turnos equivale a um total de 12 turmas. Ao ser verificado o número possível de turmas para os dois segmentos, tem-se um total de dezessete turmas (doze para o primeiro segmento e cinco para o segundo segmento). Para esta escola observa-se um excesso de professores, pois existem cinco turmas a mais que não teria como serem alocadas nesta escola. Para fins de Planejamento, pode-se dizer que para esta escola trabalhar com seu efetivo físico, o ideal seria manter as cinco possíveis turmas de segundo segmento e subtrair este valor das doze turmas de primeiro segmento. Mas é claro que a eliminação de uma ou outra turma de um dos dois segmentos deverá levar em conta a demanda de alunos para a área.

l) Escola Municipal Walfredo da Silva Lessa – onze salas multiplicadas por dois turnos equivale a um total de 22 turmas. Ao ser verificado o número possível de turmas para os dois segmentos, tem-se um total de vinte e duas turmas (dezesseis para o primeiro segmento e seis para o segundo segmento). Esta escola encontra-se trabalhando com o seu total físico.

V.3 Comentários

Como foi verificado neste capítulo, o **MatriGeo** vem apresentando os resultados esperados. Não é apenas um fator que determina a alocação ou não de um aluno em uma certa escola, e sim um conjunto de fatores pré-determinados antes do processamento das turmas.

Claro que poderiam ter sido realizados uma gama maior de testes, mas os sete aqui apresentados abordam satisfatoriamente questões importantes a serem averiguadas para analisar o perfeito funcionamento do **MatriGeo**.

CAPÍTULO VI

Conclusões

Esta dissertação de mestrado chega ao fim, tendo sido alcançados todos os objetivos traçados como meta para a elaboração da mesma.

Espera-se que este seja um, entre vários outros trabalhos, voltados para Baixada Fluminense. Isto foi visto na Apresentação, como sendo um desejo dos moradores desta região.

O **MatriGeo** é apenas um protótipo e foi desenvolvido para fazer as matrículas de alunos conforme as vagas nas escolas de uma área de estudo do Município de Nova Iguaçu. As vagas das escolas são obtidas a partir do banco de dados principal (arquivo Matrigeo.mdb) de acordo com a capacidade das referidas escolas em termos de número de carteiras, número de salas e turnos de funcionamento. A partir de então, o **MatriGeo** será capaz de definir quantas turmas de cada série deverão ser alocadas em cada escola, após o prazo da pré-matrícula.

O pleno funcionamento do sistema, alcançando todos objetivos traçados, foi visto no capítulo anterior, através dos testes que apresentaram os resultados esperados.

Neste contexto, o **MatriGeo** pretende ir além do que um sistema de informação geográfica para matricular os alunos em escolas próximas às suas residências, mas também pretende-se que o mesmo ofereça uma solução informatizada para auxiliar no planejamento e administração das escolas da rede pública.

Como melhorias futuras, são indicados o acesso à internet; inclusão de funções mais direcionadas a um SIG, por exemplo, análise espacial de pontos; inclusão de outros fatores para alocação dos alunos nas escolas.

A melhor eficiência do **MatriGeo** ocorrerá quando ele puder ser acessado (utilizado) pela Internet, pois a natureza deste processo de se fazer matrículas é notadamente ampla e distribuída, sendo assim somente em uma implementação que atinja rapidamente a uma grande quantidade de pessoas será 100% adequada.

O emprego da solução via Internet esbarra no problema da falta de acesso à rede por uma grande parte da população carente, entretanto isso poderia ser resolvido com um investimento, do governo municipal, de instalações de postos públicos para acesso à Internet. Poderia inclusive manter a estrutura das Escolas Pólo e Satélites, sendo estas escolhidas para instalação de computadores contendo o **MatriGeo**, para proceder com as matrículas.

Para que não haja conflitos, caso o **MatriGeo** venha realmente a ser implementado em plataforma *web*, será necessário incluir senhas de acesso. Isto para evitar que qualquer pessoa possa alterar o calendário, a base de dados, entre outras informações.

Em relação a inclusão de funcionalidades mais direcionadas a um Sistema de Informação Geográfica, cabe ressaltar que neste primeiro momento não foi possível implementá-las pelo tempo disponível para a elaboração do aplicativo. Este foi reduzido ao se levar em conta toda parte de programação, uma vez que sempre serão necessárias revisões dos códigos e entendimento das rotinas, muitas vezes utilizadas pela primeira vez (como foi o caso do MapObjects). Outro fator importante de ser mencionado, diz respeito ao desenvolvimento deste aplicativo não por uma equipe composta por diversos profissionais de várias áreas para suporte do mesmo, e sim desenvolvido em meio acadêmico.

Para mencionar pelo menos uma das possíveis funcionalidades a serem implementadas, citaremos a Análise Espacial de Pontos. Uma vez em que o **MatriGeo** armazena pontualmente escolas e alunos, poderiam ser feitos mapas de superfície, a partir dos pontos dos alunos. Visualmente esta opção seria de melhor entendimento por parte de um analista, do que apenas ver um amontoado de pontos. Através do mapa de superfície poderiam ser levantadas questões referentes à distribuição espacial dos alunos, verificando através destes a necessidade de inclusão de novas unidades escolares.

No tocante aos outros fatores para alocação dos alunos nas escolas, não poderia deixar de ser mencionado outro muito importante. O **MatriGeo** aloca os alunos levando-se em conta principalmente a menor distância entre a residência dos alunos até uma certa escola. Atualmente o sistema não usa um fator de alocação fundamental, que são os acidentes geográficos, que muitas vezes impossibilitam os alunos de se deslocarem a uma escola. Entende-se por acidentes geográficos aqueles naturais (rios, morros etc.), como os artificiais (estradas de rodagem, de ferro, avenidas movimentadas etc.).

Em outras palavras, nos moldes atuais um aluno poderá ser alocado em uma escola mais próxima, mas que esta poderá estar em uma localização que seria muito difícil o deslocamento do aluno. Neste caso, a implementação de lógica *fuzzy* seria um dos caminhos a serem tomados para levar em conta a alocação dos alunos pela menor distância associada aos acidentes geográficos.

A preferência do aluno por estudar em uma determinada escola também não é aboradao nesta versão do **MatriGeo**, mas fica como um outro fator de alocação dos alunos nas escolas.

Espera-se que o sistema **MatriGeo** não tenha fim em si mesmo, e que estas sugestões como melhorias futuras sejam realmente implementadas, seja pela autora ou por pessoas que tenham o real interesse no campo da otimização não somente em sistemas voltados ao Planejamento das Redes Municipais de Ensino, como ao Planejamento Urbano de um modo geral.

ANEXO A

Código Fonte do MatriGeo

A.1 Introdução

Todo o código fonte do **MatriGeo** será descrito neste anexo. Cada tópico a seguir refere-se a uma das *units* do sistema e serão descritas em ordem alfabética e não por ordem de importância. Como já foi mencionado no capítulo IV, a *unit* que contém todo o pilar do **MatriGeo** é *Interface_Usuario*.

A.1.1 *Unit* CadMatri

```
unit CadMatri;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Mask, StdCtrls, Db, ADODB, Interface_Usuario;

type
  TCadastrarMatricula = class(TForm)
    Edit_Nome: TEdit;
    Nemail: TLabel;
    Edit_email: TEdit;
    Edit_Pai: TEdit;
    Naluno: TLabel;
    Npai: TLabel;
    Nmae: TLabel;
    Edit_Mae: TEdit;
    Ndata: TLabel;
    EditData: TMaskEdit;
    Nendereco: TLabel;
    Nrua: TLabel;
    Edit_Rua: TEdit;
    Nnumero: TLabel;
    Ncomplemento: TLabel;
    Edit_Comp: TEdit;
    Nbairro: TLabel;
    Edit_Bairro: TEdit;
    Ncidade: TLabel;
    Edit_Cidade: TEdit;
    Nestado: TLabel;
    Combo_Estado: TComboBox;
    Nserie: TLabel;
    Combo_Serie: TComboBox;
    NCEP: TLabel;
    Edit_CEP: TMaskEdit;
    NTEL1: TLabel;
    Edit_TEL1: TMaskEdit;
    NTEL2: TLabel;
    Edit_TEL2: TMaskEdit;
    Botao_cadastrar: TButton;
    Botao_Localizar: TButton;
    Botao_Cancelar: TButton;
    TabelaMatri: TADOTable;
    Edit_Numero: TEdit;
    Botao_Aplicar: TButton;
    Botao_AtualizarLocalizacao: TButton;
    procedure Botao_cadastrarClick(Sender: TObject);
    procedure Botao_LocalizarClick(Sender: TObject);
    procedure Botao_CancelarClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Botao_AplicarClick(Sender: TObject);
    procedure Botao_AtualizarLocalizacaoClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  CadastrarMatricula: TCadastrarMatricula;
implementation
```

```
{$R *.DFM}

function VerificaCamposOK : boolean;

var
  campoOK : boolean;
  msgerro : string[255];

begin
  msgerro := 'ERRO - Os seguintes campos estão em branco:'
             + #13#10#13#10;
  campoOK := true;
  if CadastrarMatricula.Edit_Nome.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Nome do Aluno' + #13#10;
       end;
  if CadastrarMatricula.Edit_email.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Endereço Eletrônico' + #13#10;
       end;
  if CadastrarMatricula.Edit_Pai.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Nome do Pai' + #13#10;
       end;
  if CadastrarMatricula.Edit_Mae.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Nome da Mãe' + #13#10;
       end;
  if CadastrarMatricula.EditData.Text = '  /  /    '
  then begin
         campoOK := false;
         msgerro := msgerro + 'Data de Nascimento' + #13#10;
       end;
  if CadastrarMatricula.Edit_Rua.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Rua' + #13#10;
       end;
  if CadastrarMatricula.Edit_Numero.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Número' + #13#10;
       end;
  if CadastrarMatricula.Edit_Comp.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Complemento' + #13#10;
       end;
  if CadastrarMatricula.Edit_Bairro.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Bairro' + #13#10;
       end;
  if CadastrarMatricula.Edit_Cidade.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Cidade' + #13#10;
       end;
  if CadastrarMatricula.Combo_Estado.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Estado' + #13#10;
       end;
  if CadastrarMatricula.Combo_Serie.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Série' + #13#10;
       end;
  if CadastrarMatricula.Edit_CEP.Text = '     -   '
  then begin
         campoOK := false;
         msgerro := msgerro + 'CEP' + #13#10;
       end;
  if not campoOK then showmessage (msgerro);
  VerificaCamposOK := campoOK;
end;

procedure CarregaCampos;

begin
  CadastrarMatricula.TabelaMatri.FieldValues['NOME'] :=
     CadastrarMatricula.Edit_Nome.text;
  CadastrarMatricula.TabelaMatri.FieldValues['RUA'] :=
     CadastrarMatricula.Edit_Rua.text;
  CadastrarMatricula.TabelaMatri.FieldValues['NUM'] :=
     CadastrarMatricula.Edit_Numero.text;
  CadastrarMatricula.TabelaMatri.FieldValues['COMPL'] :=
     CadastrarMatricula.Edit_Comp.text;
  CadastrarMatricula.TabelaMatri.FieldValues['BAIRRO'] :=
     CadastrarMatricula.Edit_Bairro.text;
  CadastrarMatricula.TabelaMatri.FieldValues['CIDADE'] :=
     CadastrarMatricula.Edit_Cidade.text;
  CadastrarMatricula.TabelaMatri.FieldValues['ESTADO'] :=
     CadastrarMatricula.Combo_Estado.text;
  CadastrarMatricula.TabelaMatri.FieldValues['CEP'] :=
     CadastrarMatricula.Edit_CEP.text;
  CadastrarMatricula.TabelaMatri.FieldValues['TEL1'] :=
     CadastrarMatricula.Edit_TEL1.text;
  CadastrarMatricula.TabelaMatri.FieldValues['TEL2'] :=
     CadastrarMatricula.Edit_TEL2.text;
  CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] :=
     CadastrarMatricula.Combo_Serie.text;
  CadastrarMatricula.TabelaMatri.FieldValues['PAI'] :=
```

```
     CadastrarMatricula.Edit_Pai.text;
  CadastrarMatricula.TabelaMatri.FieldValues['MAE'] :=
     CadastrarMatricula.Edit_Mae.text;
  CadastrarMatricula.TabelaMatri.FieldValues['NASCIMENTO'] :=
     CadastrarMatricula.EditData.text;
  CadastrarMatricula.TabelaMatri.FieldValues['EMAIL'] :=
     CadastrarMatricula.Edit_email.text;
end;

procedure TCadastrarMatricula.Botao_cadastrarClick(Sender: TObject);

var
   CodigoAluno : cardinal;

begin
 if VerificaCamposOK
 then
 begin
  CodigoAluno := 0;
  if TabelaMatri.RecordCount <> 0
  then TabelaMatri.first;
       while not TabelaMatri.eof do
         begin
           if TabelaMatri.FieldValues['COD_ALUN'] > CodigoAluno
           then CodigoAluno := TabelaMatri.FieldValues['COD_ALUN'];
           TabelaMatri.Next;
         end;
  inc (CodigoAluno);
  TabelaMatri.Append;
  TabelaMatri.FieldValues['COD_ALUN'] := CodigoAluno;
  CarregaCampos;
  Botao_cadastrar.Enabled := false;
  Botao_Cancelar.Enabled := false;
  Botao_Localizar.Enabled := true;
  Form1.Matrcula1.Enabled := false;
  Form1.Consultas.Enabled := false;
  Form1.BD.Enabled := false;
  Form1.Relatrios1.Enabled := false;
 end;
end;

procedure TCadastrarMatricula.Botao_LocalizarClick(Sender: TObject);

begin
  EstouLocalizando := true;
  CadastrarMatricula.Visible := false;
  showmessage ('Acione o botão direito do mouse para localizar.');
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TCadastrarMatricula.Botao_CancelarClick(Sender: TObject);

begin
  CadastrarMatricula.TabelaMatri.first;
  CadastrarMatricula.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TCadastrarMatricula.FormClose(Sender: TObject;
  var Action: TCloseAction);

begin
  if Botao_cadastrar.Enabled = false
  then Action := caNone
  else Action := caFree;
end;

procedure TCadastrarMatricula.Botao_AplicarClick(Sender: TObject);

var
  AlterouLocal : boolean;

begin
 if VerificaCamposOK
 then begin
        AlterouLocal :=
        (CadastrarMatricula.Edit_Rua.Text <>
         CadastrarMatricula.TabelaMatri.FieldValues['RUA']) or
        (CadastrarMatricula.Edit_Numero.Text <>
         CadastrarMatricula.TabelaMatri.FieldValues['NUM']);
        CadastrarMatricula.TabelaMatri.Edit;
        CarregaCampos;
        if AlterouLocal
        then begin
               Botao_Aplicar.Enabled := false;
               Botao_Cancelar.Enabled := false;
               Botao_AtualizarLocalizacao.Enabled := true;
               Form1.Matrcula1.Enabled := false;
               Form1.Consultas.Enabled := false;
               Form1.BD.Enabled := false;
               Form1.Relatrios1.Enabled := false;
               showmessage ('Informações atualizadas.' + #13#10 + #13#10
                           + 'Atualização da localização é obrigatória!');
             end
        else begin
              CadastrarMatricula.TabelaMatri.Post;
              showmessage ('Informações atualizadas.');
             end;
      end;
end;

procedure TCadastrarMatricula.Botao_AtualizarLocalizacaoClick(
  Sender: TObject);

begin
```

```
  EstouAtualizandoLocal := true;
  CadastrarMatricula.Visible := false;
  showmessage ('Acione o botão direito do mouse para atualizar a localização.');
  Form1.Enabled := true;
  Form1.SetFocus;
end;

end.
```

A.1.2 *Unit* ConsultAlun

```
unit ConsultAlun;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, DBGrids, CadMatri, Db, StdCtrls, Mask, DBCtrls, ExtCtrls,
  Interface_Usuario;

type
  TConsultaAluno = class(TForm)
    DataSourceAluno: TDataSource;
    Naluno: TLabel;
    Npai: TLabel;
    Nmae: TLabel;
    Nemail: TLabel;
    NTEL2: TLabel;
    NTEL1: TLabel;
    Nserie: TLabel;
    Ndata: TLabel;
    PainelEndereco: TPanel;
    Nendereco: TLabel;
    NCEP: TLabel;
    Nrua: TLabel;
    Ncomplemento: TLabel;
    Ncidade: TLabel;
    Nbairro: TLabel;
    Nestado: TLabel;
    Nnumero: TLabel;
    DBEditNascimento: TDBEdit;
    DBEditPai: TDBEdit;
    DBEditMae: TDBEdit;
    DBEditEmail: TDBEdit;
    DBEditSerie: TDBEdit;
    DBEditTel1: TDBEdit;
    DBEditTel2: TDBEdit;
    DBEditNumero: TDBEdit;
    DBEditCep: TDBEdit;
    DBEditRua: TDBEdit;
    DBEditComp: TDBEdit;
    DBEditBairro: TDBEdit;
    DBEditCidade: TDBEdit;
    DBEditEstado: TDBEdit;
    DBNavegador: TDBNavigator;
    Ncodigo: TLabel;
    DBEditCodigo: TDBEdit;
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    BotaoPesquisar: TButton;
    BotaoCancelar: TButton;
    DBEditNome: TDBEdit;
    ComboPalavraChave: TComboBox;
    procedure BotaoCancelarClick(Sender: TObject);
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure ComboPalavraChaveChange(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

  function RetiraAcentos (str : string) : string;

var
  ConsultaAluno: TConsultaAluno;

implementation

{$R *.DFM}

procedure TConsultaAluno.BotaoCancelarClick(Sender: TObject);

begin
  ConsultaAluno.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

function RetiraAcentos (str : string) : string;

var
  i              : byte;
  str_sem_acentos : string[255];

begin
   str_sem_acentos := str;
   for i := 1 to length (str) do
       case str[i] of
         'á','à','ã','â' : str_sem_acentos[i] := 'a';
         'é','ê' : str_sem_acentos[i] := 'e';
         'í' : str_sem_acentos[i] := 'i';
         'ó','ô','õ' : str_sem_acentos[i] := 'o';
         'ú','ü' : str_sem_acentos[i] := 'u';
         'ç' : str_sem_acentos[i] := 'c';
```

```
        'Á','À','Ã','Â' : str_sem_acentos[i] := 'A';
        'É','Ê' : str_sem_acentos[i] := 'E';
        'Í' : str_sem_acentos[i] := 'I';
        'Ó','Ô','Õ' : str_sem_acentos[i] := 'O';
        'Ú','Ü' : str_sem_acentos[i] := 'U';
        'Ç' : str_sem_acentos[i] := 'C';
      end;
    RetiraAcentos := str_sem_acentos;
end;

procedure TConsultaAluno.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   CadastrarMatricula.TabelaMatri.first;
   ListaDEnomes := TStringList.Create;
   while not CadastrarMatricula.TabelaMatri.Eof do
      begin
        straux := CadastrarMatricula.TabelaMatri.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (CadastrarMatricula.TabelaMatri.FieldValues['NOME']);
        CadastrarMatricula.TabelaMatri.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure TConsultaAluno.ComboPalavraChaveChange(Sender: TObject);

var
  achei : boolean;

begin
   achei := false;
   CadastrarMatricula.TabelaMatri.first;
   while not (CadastrarMatricula.TabelaMatri.Eof)
         and not achei do
      begin
        if CadastrarMatricula.TabelaMatri.FieldValues['NOME'] =
           ComboPalavraChave.Text
        then achei := true
        else CadastrarMatricula.TabelaMatri.next;
      end;
end;

end.
```

A.1.3 *Unit* ConsultCalen

```
unit ConsultCalen;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, Grids, DBGrids;

type
  TConsultaCalendario = class(TForm)
    DBGrid1: TDBGrid;
    DataSourceCalendario: TDataSource;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ConsultaCalendario: TConsultaCalendario;

implementation

{$R *.DFM}

end.
```

A.1.4 *Unit* ConsultEsc

```
unit ConsultEsc;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, DBCtrls, Db, ADODB, Mask, ExtCtrls,
  ConsultAlun, Interface_Usuario;

type
  TConsultaEscola = class(TForm)
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    BotaoPesquisar: TButton;
    BotaoCancelar: TButton;
    ComboPalavraChave: TComboBox;
    PainelEndereco: TPanel;
    Nendereco: TLabel;
    NCEP: TLabel;
```

```
    Nrua: TLabel;
    Ncomplemento: TLabel;
    Ncidade: TLabel;
    Nbairro: TLabel;
    Nestado: TLabel;
    Nnumero: TLabel;
    DBEditNumero: TDBEdit;
    DBEditCep: TDBEdit;
    DBEditRua: TDBEdit;
    DBEditComp: TDBEdit;
    DBEditBairro: TDBEdit;
    DBEditCidade: TDBEdit;
    DBEditEstado: TDBEdit;
    TabEscola: TADOTable;
    DataSourceEscola: TDataSource;
    DBNavegador: TDBNavigator;
    Nescola: TLabel;
    Ncorrente: TLabel;
    Naval: TLabel;
    Nconst: TLabel;
    Ndiretor: TLabel;
    DBEditNascimento: TDBEdit;
    DBEditPai: TDBEdit;
    DBEditMae: TDBEdit;
    DBEditEmail: TDBEdit;
    DBEditNome: TDBEdit;
    DBEdit1: TDBEdit;
    Nsalas: TLabel;
    DBEdit2: TDBEdit;
    Ncart: TLabel;
    Ncodigo: TLabel;
    DBEditCodigo: TDBEdit;
    DBEnergia: TDBCheckBox;
    DBagua: TDBCheckBox;
    DBesgoto: TDBCheckBox;
    DBeduc_infantil: TDBCheckBox;
    DBensino_fund: TDBCheckBox;
    DBensino_ja: TDBCheckBox;
    NTEL: TLabel;
    DBEditTel: TDBEdit;
    procedure BotaoCancelarClick(Sender: TObject);
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure ComboPalavraChaveChange(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ConsultaEscola: TConsultaEscola;

implementation

{$R *.DFM}

procedure TConsultaEscola.BotaoCancelarClick(Sender: TObject);

begin
  ConsultaEscola.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TConsultaEscola.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   TabEscola.first;
   ListaDEnomes := TStringList.Create;
   while not TabEscola.Eof do
      begin
        straux := TabEscola.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (TabEscola.FieldValues['NOME']);
        TabEscola.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure TConsultaEscola.ComboPalavraChaveChange(Sender: TObject);

var
  achei : boolean;

begin
   achei := false;
   TabEscola.first;
   while not (TabEscola.Eof)
         and not achei do
      begin
        if TabEscola.FieldValues['NOME'] =
           ComboPalavraChave.Text
        then achei := true
        else TabEscola.next;
      end;
end;

end.
```

A.1.5 *Unit* ConsultProf

```
unit ConsultProf;

interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Interface_Usuario, StdCtrls, DBCtrls, Mask, ExtCtrls, Db, ADODB,
  ConsultAlun;

type
  TConsultaProfessor = class(TForm)
    TabProfessor: TADOTable;
    DataSourceProf: TDataSource;
    Nprof: TLabel;
    Nemail: TLabel;
    NTEL: TLabel;
    Ncodigo: TLabel;
    PainelEndereco: TPanel;
    Nendereco: TLabel;
    NCEP: TLabel;
    Nrua: TLabel;
    Ncomplemento: TLabel;
    Ncidade: TLabel;
    Nbairro: TLabel;
    Nestado: TLabel;
    Nnumero: TLabel;
    DBEditNumero: TDBEdit;
    DBEditCep: TDBEdit;
    DBEditRua: TDBEdit;
    DBEditComp: TDBEdit;
    DBEditBairro: TDBEdit;
    DBEditCidade: TDBEdit;
    DBEditEstado: TDBEdit;
    DBEditEmail: TDBEdit;
    DBEditTel: TDBEdit;
    DBNavegador: TDBNavigator;
    DBEditNome: TDBEdit;
    DBEditCodigo: TDBEdit;
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    BotaoPesquisar: TButton;
    BotaoCancelar: TButton;
    ComboPalavraChave: TComboBox;
    DB1seg: TDBCheckBox;
    DB2seg: TDBCheckBox;
    procedure BotaoCancelarClick(Sender: TObject);
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure ComboPalavraChaveChange(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ConsultaProfessor: TConsultaProfessor;

implementation

{$R *.DFM}

procedure TConsultaProfessor.BotaoCancelarClick(Sender: TObject);

begin
  ConsultaProfessor.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TConsultaProfessor.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   TabProfessor.first;
   ListaDEnomes := TStringList.Create;
   while not TabProfessor.Eof do
      begin
        straux := TabProfessor.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (TabProfessor.FieldValues['NOME']);
        TabProfessor.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure TConsultaProfessor.ComboPalavraChaveChange(Sender: TObject);

var
  achei : boolean;

begin
   achei := false;
   TabProfessor.first;
   while not (TabProfessor.Eof)
         and not achei do
      begin
        if TabProfessor.FieldValues['NOME'] =
           ComboPalavraChave.Text
        then achei := true
```

```
       else TabProfessor.next;
     end;
end;

end.
```

A.1.6 *Unit* ConsultTurmProc

```
unit ConsultTurmProc;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, Mask, DBCtrls, Interface_Usuario, Db, ADODB,
  ConsultEsc, ConsultAlun, Grids, DBGrids;

type
  TConsultaTurmasProcessadas = class(TForm)
    DBEditCodigoEsc: TDBEdit;
    Ncodigoesc: TLabel;
    DBEditNomeEsc: TDBEdit;
    Nescola: TLabel;
    DBEditCodigoTurma: TDBEdit;
    Ncodigoturma: TLabel;
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    BotaoPesquisar: TButton;
    BotaoCancelar: TButton;
    ComboPalavraChave: TComboBox;
    QueryAluno: TADOQuery;
    DataSourceQueryAluno: TDataSource;
    ComboNomeTurma: TComboBox;
    NnomeTurma: TLabel;
    Nalunos: TLabel;
    Nprofessores: TLabel;
    DBGridAlunos: TDBGrid;
    Nserie: TLabel;
    DBEditSerie: TDBEdit;
    DBGridProfessores: TDBGrid;
    QueryProfessor: TADOQuery;
    DataSourceQueryProfessor: TDataSource;
    DataSourceEscola: TDataSource;
    DataSourceTurma: TDataSource;
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure ComboPalavraChaveChange(Sender: TObject);
    procedure ComboNomeTurmaChange(Sender: TObject);
    procedure BotaoCancelarClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  ConsultaTurmasProcessadas : TConsultaTurmasProcessadas;

procedure LimpaCampos;

implementation

{$R *.DFM}

procedure TConsultaTurmasProcessadas.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   ConsultEsc.ConsultaEscola.TabEscola.first;
   ListaDEnomes := TStringList.Create;
   while not ConsultEsc.ConsultaEscola.TabEscola.Eof do
      begin
        straux := ConsultEsc.ConsultaEscola.TabEscola.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append
            (ConsultEsc.ConsultaEscola.TabEscola.FieldValues['NOME']);
        ConsultEsc.ConsultaEscola.TabEscola.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure LimpaCampos;

begin
  ConsultaTurmasProcessadas.DBEditNomeEsc.DataField := '';
  ConsultaTurmasProcessadas.DBEditCodigoEsc.DataField := '';
  ConsultaTurmasProcessadas.DBEditCodigoTurma.DataField := '';
  ConsultaTurmasProcessadas.DBEditSerie.DataField := '';
  ConsultaTurmasProcessadas.QueryProfessor.Active := false;
  ConsultaTurmasProcessadas.QueryAluno.Active := false;
end;

procedure TConsultaTurmasProcessadas.ComboPalavraChaveChange(
  Sender: TObject);

var
  achei        : boolean;
  ListaDEnomes : TStrings;
  etapas       : string[12];
```

```
begin
   ComboNomeTurma.Clear;
   achei := false;
   ConsultEsc.ConsultaEscola.TabEscola.first;
   while not (ConsultEsc.ConsultaEscola.TabEscola.Eof)
         and not achei do
      begin
        if ConsultEsc.ConsultaEscola.TabEscola.FieldValues['NOME'] =
           ComboPalavraChave.Text
        then begin
              achei := true;
              LimpaCampos;
              DBEditNomeEsc.DataField := 'NOME';
              DBEditCodigoEsc.DataField := 'COD_ESC';
              Form1.TabTurma.first;
              ListaDEnomes := TStringList.Create;
              while not Form1.TabTurma.Eof do
               begin
                if Form1.TabTurma.FieldValues['COD_ESC'] =
                   ConsultEsc.ConsultaEscola.TabEscola.FieldValues['COD_ESC']
                then begin
                      etapas := '';
                      if Form1.TabTurma.FieldValues['SERIE'] = 'Primeira Etapa'
                      then etapas := ' 1a. Etapa';
                      if Form1.TabTurma.FieldValues['SERIE'] = 'Segunda Etapa'
                      then etapas := ' 2a. Etapa';
                      if Form1.TabTurma.FieldValues['SERIE'] = 'Terceira Etapa'
                      then etapas := ' 3a. Etapa';
                      ListaDEnomes.Append
                           (Form1.TabTurma.FieldValues['NOME'] + etapas);
                     end;
                Form1.TabTurma.next;
               end;
              ComboNomeTurma.Items := ListaDEnomes;
              ListaDEnomes.Free;
             end
        else ConsultEsc.ConsultaEscola.TabEscola.next;
      end;
end;

procedure TConsultaTurmasProcessadas.ComboNomeTurmaChange(Sender: TObject);

var
  achei,AcheiTurma,SeriesEtapas : boolean;
  nome_turma,Serie              : string[20];

begin
   achei := false;
   QueryProfessor.Active := false;
   QueryAluno.Active := false;
   SeriesEtapas := (pos ('Etapa',ComboNomeTurma.Text) <> 0);
   if SeriesEtapas
   then begin
          nome_turma := ComboNomeTurma.Text[1];
          case ComboNomeTurma.Text[3] of
            '1' : Serie := 'Primeira Etapa';
            '2' : Serie := 'Segunda Etapa';
            '3' : Serie := 'Terceira Etapa';
          end;
        end
   else nome_turma := ComboNomeTurma.Text;
   Form1.TabTurma.first;
   while not (Form1.TabTurma.Eof) and not achei do
      begin
        if SeriesEtapas
        then AcheiTurma :=
              (Form1.TabTurma.FieldValues['NOME'] = nome_turma) and
              (ConsultEsc.ConsultaEscola.TabEscola.FieldValues['COD_ESC'] =
               Form1.TabTurma.FieldValues['COD_ESC']) and
              (Serie = Form1.TabTurma.FieldValues['SERIE'])
        else AcheiTurma :=
              (Form1.TabTurma.FieldValues['NOME'] = nome_turma) and
              (ConsultEsc.ConsultaEscola.TabEscola.FieldValues['COD_ESC'] =
               Form1.TabTurma.FieldValues['COD_ESC']);
        if AcheiTurma
        then begin
              achei := true;
              DBEditSerie.DataField := 'SERIE';
              DBEditCodigoTurma.DataField := 'COD_TURM';
              QueryProfessor.Parameters.ParamValues['CT'] :=
                Form1.TabTurma.FieldValues['COD_TURM'];
              QueryProfessor.Active := true;
              QueryAluno.Parameters.ParamValues['CT'] :=
                Form1.TabTurma.FieldValues['COD_TURM'];
              QueryAluno.Active := true;
             end
        else Form1.TabTurma.next;
      end;
end;

procedure TConsultaTurmasProcessadas.BotaoCancelarClick(Sender: TObject);

begin
  ConsultaTurmasProcessadas.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

end.
```

A.1.7 Unit DefCalen

```
unit DefCalen;
```

```
interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Mask, Db, ADODB, Interface_Usuario;

type
  TDefinirCalendario = class(TForm)
    EditDataInicial: TMaskEdit;
    Combo_Descricao: TComboBox;
    EditDataFinal: TMaskEdit;
    Ndescricao: TLabel;
    Ninicial: TLabel;
    Nfinal: TLabel;
    Botao_Aplicar: TButton;
    Botao_Cancelar: TButton;
    procedure Botao_AplicarClick(Sender: TObject);
    procedure Botao_CancelarClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  DefinirCalendario: TDefinirCalendario;

implementation

{$R *.DFM}

procedure TDefinirCalendario.Botao_AplicarClick(Sender: TObject);

var
   inicio, fim : TDateTime;

begin
  if (EditDataInicial.Text = '  /  /    ') or
     (EditDataFinal.Text = '  /  /    ')
  then showmessage ('ERRO - Uma ou mais data(s) está(ão) em branco.')
  else if Combo_Descricao.Text = 'Matrícula'
       then begin
             inicio := StrToDate (EditDataInicial.Text);
             fim := StrToDate (EditDataFinal.Text);
             if fim >= inicio
             then begin
                   Form1.TabCal.first;
                   Form1.TabCal.Edit;
                   Form1.TabCal.FieldValues['INICIO'] := inicio;
                   Form1.TabCal.FieldValues['FIM'] := fim;
                   Form1.TabCal.Post;
                   Form1.Enabled := true;
                   Form1.SetFocus;
                   DefinirCalendario.Visible := false;
                  end
             else showmessage ('Datas inconsistentes!');
            end;
end;

procedure TDefinirCalendario.Botao_CancelarClick(Sender: TObject);

begin
  Form1.Enabled := true;
  Form1.SetFocus;
  DefinirCalendario.Visible := false;
end;

end.
```

A.1.8 *Unit* Interface_Usuario

```
unit Interface_Usuario;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  OleCtrls, MapObjects2_TLB, StdCtrls, Db, ADODB, Menus, Buttons, ComCtrls,
  ToolWin, ComObj, ExtCtrls, ActiveX, ImgList, jpeg, Mask;

type
  TForm1 = class(TForm)
    Mapa_Austin: TMap;
    Matricula: TMainMenu;
    Matrcula1: TMenuItem;
    BD: TMenuItem;
    Consultas: TMenuItem;
    PreMatri: TMenuItem;
    MatriProceTurma: TMenuItem;
    DefinirCalend: TMenuItem;
    Sair1: TMenuItem;
    Exibir: TMenuItem;
    Zoom: TMenuItem;
    ToolBar1: TToolBar;
    StatusBar1: TStatusBar;
    Cadastrar: TMenuItem;
    AtualizarPreMatri: TMenuItem;
    ExcluirCadastro1: TMenuItem;
    EscolaBD: TMenuItem;
    ConsultaAluno: TMenuItem;
    ConsultaEscola: TMenuItem;
    ConsultTurmasProcessadas: TMenuItem;
    Aproximar: TMenuItem;
    Afastar: TMenuItem;
    ExtensoTotal1: TMenuItem;
```

```
    Logradouro: TMenuItem;
    Deslocar: TMenuItem;
    Camadas1: TMenuItem;
    Relatrios1: TMenuItem;
    PreMatriculaRelat: TMenuItem;
    TurmasProcessadasRelat: TMenuItem;
    Ajuda1: TMenuItem;
    Sobre1: TMenuItem;
    AproximarporSel: TMenuItem;
    TabCal: TADOTable;
    BasedeDadosRelat: TMenuItem;
    ConsultaProfessor: TMenuItem;
    ConsultaCalendario: TMenuItem;
    TabTurma: TADOTable;
    TabProfEsc: TADOTable;
    TabProfTurma: TADOTable;
    ProgressBarProcTur: TProgressBar;
    NProcessandoTurmas: TStaticText;
    Panel1: TPanel;
    LogradouroBD: TMenuItem;
    ProfessorBD: TMenuItem;
    BotaoConfSelQuad: TSpeedButton;
    BotaoZoomSelec: TSpeedButton;
    BotaoZoomAproximar: TSpeedButton;
    BotaoZoomAfastar: TSpeedButton;
    SpeedButton1: TSpeedButton;
    BotaoDeslocar: TSpeedButton;
    MenuMapas: TMenuItem;
    MenuNovaIguau: TMenuItem;
    MenuMapaAustin: TMenuItem;
    MenuMapaAlunosEsc: TMenuItem;
    MenuMapaEscolas: TMenuItem;
    ImageList1: TImageList;
    ScrollBox1: TScrollBox;
    Label1: TLabel;
    Label2: TLabel;
    PanelLegenda: TPanel;
    Nlegenda: TLabel;
    TreeView1: TTreeView;
    Image1: TImage;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Panel2: TPanel;
    StaticText1: TStaticText;
    MaskEdit1: TMaskEdit;
    Map1: TMap;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Panel3: TPanel;
    Label15: TLabel;
    TreeView2: TTreeView;
    Panel4: TPanel;
    Label16: TLabel;
    TreeView3: TTreeView;
    Label17: TLabel;
    SpeedButton2: TSpeedButton;
    Panel5: TPanel;
    Label18: TLabel;
    TreeView4: TTreeView;
    Label19: TLabel;
    SpeedButton3: TSpeedButton;
    Label20: TLabel;
    Label21: TLabel;
    procedure zoomtotal(Sender: TObject);
    procedure Camadas1Click(Sender: TObject);
    procedure SairProc(Sender: TObject);
    procedure Iniciar_mapas(Sender: TObject);
    procedure AproximarProc(Sender: TObject);
    procedure AfastarProc(Sender: TObject);
    procedure ZoomPanProc(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure DeslocarProc(Sender: TObject);
    procedure AproximarporSelProc(Sender: TObject);
    procedure ZoomLogradouroProc(Sender: TObject);
    procedure CadastrarProc(Sender: TObject);
    procedure AtualizarPreMatriClick(Sender: TObject);
    procedure ExcluirCadastro1Click(Sender: TObject);
    procedure DefinirCalendClick(Sender: TObject);
    procedure ConsultaAlunoClick(Sender: TObject);
    procedure ConsultaEscolaClick(Sender: TObject);
    procedure ConsultaProfessorClick(Sender: TObject);
    procedure ConsultaCalendarioClick(Sender: TObject);
    procedure MatriProceTurmaClick(Sender: TObject);
    procedure ConsultTurmasProcessadasClick(Sender: TObject);
    procedure PreMatriculaRelatClick(Sender: TObject);
    procedure BasedeDadosRelatClick(Sender: TObject);
    procedure TurmasProcessadasRelatClick(Sender: TObject);
    procedure Panel1Resize(Sender: TObject);
    procedure EscolaBDClick(Sender: TObject);
    procedure ProfessorBDClick(Sender: TObject);
    procedure LogradouroBDClick(Sender: TObject);
    procedure Mapa_AustinAfterLayerDraw(Sender: TObject; index: Smallint;
      canceled: WordBool; hDC: Cardinal);
    procedure BotaoConfSelQuadClick(Sender: TObject);
    procedure BotaoZoomSelecClick(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure BotaoZoomAproximarClick(Sender: TObject);
    procedure BotaoZoomAfastarClick(Sender: TObject);
    procedure BotaoDeslocarClick(Sender: TObject);
```

```
    procedure FormResize(Sender: TObject);
    procedure MenuMapaAustinClick(Sender: TObject);
    procedure MenuNovaIguauClick(Sender: TObject);
    procedure MenuMapaEscolasClick(Sender: TObject);
    procedure FormClick(Sender: TObject);
    procedure MenuMapaAlunosEscClick(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

procedure ApagarProcessamentoAnterior;

var
  Form1                   : TForm1;
  Qt_quadras_por_linha    : array [1..10] of byte;
  EstouLocalizando        : boolean;
  EstouAtualizandoLocal   : boolean;
  EstouSelecionandoQuadra : boolean;
  sym                     : IMoSymbol;
  shp                     : IMoPolygon;
  tLayer,tLayer2          : IMoTrackingLayer;
  unidade                 : string[20];
  SelecioneiQuadra        : boolean;
  SelecioneiEscola        : boolean;
  Form1Iniciado           : boolean;
  ApagarPontos            : boolean;
  DadosPARAmatricular     : record
                              TurmaAberta          : boolean;
                              CodigoTurma          : cardinal;
                              Distancia            : double;
                              AreaDEinfluencia     : integer;
                              CodigoProfPrimSeg    : cardinal;
                              CodigoProfMatematica : cardinal;
                              CodigoProfPortugues  : cardinal;
                              CodigoProfCiencias   : cardinal;
                              CodigoProfHistoria   : cardinal;
                              CodigoProfGeografia  : cardinal;
                              CodigoProfEduFisica  : cardinal;
                              CodigoProfIngles     : cardinal;
                              CodigoProfArtes      : cardinal;
                            end;

implementation

uses SelCamadas, Selog, CadMatri, SelCad, DefCalen, ConsultAlun,
     ConsultEsc, ConsultProf, ConsultCalen, ConsultTurmProc,
     RelPreMatri, RelBD, RelTurProc, UnitEscolaBD, UnitProfessorBD,
     UnitLogradouroBD, UnitDescEsc, SelEscSerie;

{$R *.DFM}

procedure TForm1.zoomtotal(Sender: TObject);

begin
 Mapa_Austin.Extent := Mapa_Austin.FullExtent;
 if SpeedButton3.Enabled
 then begin
        SelecioneiEscola := true;
        Mapa_Austin.Refresh;
      end;
end;

procedure TForm1.Camadas1Click(Sender: TObject);

begin
  SelecCamadas.Visible := true;
end;

procedure TForm1.SairProc(Sender: TObject);

begin
  halt;
end;

procedure IniciarMapaNI;

var
 strings,strings2 : IMoStrings;
 ly               : IMoMapLayer;
 recs             : IMoRecordset;
 s                : String;
 i                : integer;
 lyrs             : IMoLayers;
 flds             : IMoFields;
 fld              : IMoField;
 ren              : IMoValueMapRenderer;

begin
 lyrs := IMoLayers(form1.map1.layers);
 strings := IMoStrings(CreateOleObject('MapObjects2.Strings'));
 ly := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
 ly := IMoMapLayer(lyrs.item('Bairros'));
 recs := IMoRecordset(ly.records);
 while not recs.eof do
  begin
   flds := IMoFields(recs.fields);
   fld := IMoField(flds.item('Setores'));
   s := fld.Value;
   strings.add(s);
   recs.movenext;
  end;
 ren := IMoValueMapRenderer(CreateOleObject('MapObjects2.ValueMapRenderer'));
 ren.Field :='Setores';
```

```
 ren.ValueCount :=strings.count;
 for i :=0 to Strings.Count-1 do
  begin
    ren.value[i]:= strings.Item (i);
    case i of
      0: ren.Symbol[i].Color := moBlue;
      1: ren.Symbol[i].Color := clred;
      2: ren.Symbol[i].Color := clyellow;
      3: ren.Symbol[i].Color := moDarkGreen;
      4: ren.Symbol[i].Color := moCyan;
      5: ren.Symbol[i].Color := moMagenta;
    end;
  end;
 ly.renderer := ren;
 strings2 := IMoStrings(CreateOleObject('MapObjects2.Strings'));
 ly := IMoMapLayer(lyrs.item('norteni'));
 recs := IMoRecordset(ly.records);
 while not recs.eof do
  begin
   flds := IMoFields(recs.fields);
   fld := IMoField(flds.item('ID'));
   s := fld.Value;
   strings2.add(s);
   recs.movenext;
  end;
 ren := IMoValueMapRenderer(CreateOleObject('MapObjects2.ValueMapRenderer'));
 ren.Field :='ID';
 ren.ValueCount :=strings2.count;
 for i :=0 to Strings2.Count-1 do
   begin
    ren.value[i]:= strings2.Item (i);
    case i of
      0: ren.Symbol[i].Color := clBlack;
      1: ren.Symbol[i].Color := moLightGray;
      2: ren.Symbol[i].Color := clBlack;
    end;
   end;
 ly.renderer := ren;
 form1.Map1.Refresh;
end;

procedure ColorirAustin (Monocromatico : boolean);

var
  Camadas    : IMoLayers;
  Camada     : IMoMapLayer;
  i          : byte;
  strings    : IMoStrings;
  recs       : IMoRecordset;
  s          : String;
  flds       : IMoFields;
  fld        : IMoField;
  ren        : IMoValueMapRenderer;

begin
  camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  camadas := Form1.Mapa_Austin.Layers;
  Camada := IMoMapLayer(Camadas.Item('austin'));
  strings := IMoStrings(CreateOleObject('MapObjects2.Strings'));
  recs := IMoRecordset(camada.records);
  while not recs.eof do
    begin
      flds := IMoFields(recs.fields);
      fld := IMoField(flds.item('Bairro'));
      s := fld.Value;
      strings.add(s);
      recs.movenext;
    end;
  ren := IMoValueMapRenderer(CreateOleObject('MapObjects2.ValueMapRenderer'));
  ren.Field :='Bairro';
  ren.ValueCount :=strings.count;
  for i :=0 to Strings.Count-1 do
    begin
      ren.value[i]:= strings.Item (i);
      if Monocromatico
      then case i of
             0: ren.Symbol[i].Color := clyellow;
             1: ren.Symbol[i].Color := clyellow;
             2: ren.Symbol[i].Color := clyellow;
             3: ren.Symbol[i].Color := clyellow;
             4: ren.Symbol[i].Color := clyellow;
             5: ren.Symbol[i].Color := clyellow;
             6: ren.Symbol[i].Color := clyellow;
             7: ren.Symbol[i].Color := clyellow;
           end
      else case i of
             0: ren.Symbol[i].Color := moCyan;
             1: ren.Symbol[i].Color := moMagenta;
             2: ren.Symbol[i].Color := clred;
             3: ren.Symbol[i].Color := clyellow;
             4: ren.Symbol[i].Color := moDarkGreen;
             5: ren.Symbol[i].Color := moGreen;
             6: ren.Symbol[i].Color := moLightGray;
             7: ren.Symbol[i].Color := moBlue;
           end;
    end;
  camada.renderer := ren;
end;

procedure TForm1.Iniciar_mapas(Sender: TObject);

var
  Camadas    : IMoLayers;
  Camada     : IMoMapLayer;
  DCtexto,dc : IMoDataConnection;
  simbolo    : IMoSymbol;
  rend       : IMoLabelRenderer;
```

```
  i          : byte;
  strings2   : IMoStrings;
  recs       : IMoRecordset;
  s          : String;
  flds       : IMoFields;
  fld        : IMoField;
  ren        : IMoValueMapRenderer;

begin
  IniciarMapaNI;
  dc := IMoDataConnection(CreateOleObject('MapObjects2.DataConnection'));
  dc.Database := 'd:\mestrado\tese\aplicativo' ;
  if (dc.Connect) then
    begin
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := IMoGeoDataset(dc.FindGeoDataset('austin'));
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       ColorirAustin (false);
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := dc.FindGeoDataset('area');
       simbolo := camada.Symbol;
       simbolo.style := 1;
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := IMoGeoDataset(dc.FindGeoDataset('rios_inter'));
       simbolo := camada.Symbol;
       simbolo.Color := moBlue;
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := IMoGeoDataset(dc.FindGeoDataset('ferrovia_inter'));
       simbolo := camada.Symbol;
       simbolo.Color := clBlack;
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := IMoGeoDataset(dc.FindGeoDataset('lograd_inter'));
       simbolo := camada.Symbol;
       simbolo.Color := clMaroon;
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := IMoGeoDataset(dc.FindGeoDataset('escolas'));
       simbolo := camada.Symbol;
       simbolo.Color := moOrange;
       simbolo.Size := 8;
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := IMoGeoDataset(dc.FindGeoDataset('buff2'));
       simbolo := camada.Symbol;
       simbolo.style := 1;
       simbolo.Size := 2;
       camada.Visible := false;
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       camada.GeoDataset := IMoGeoDataset(dc.FindGeoDataset('norte'));
       camadas := Mapa_Austin.Layers;
       camadas.Add(camada);
       strings2 := IMoStrings(CreateOleObject('MapObjects2.Strings'));
       recs := IMoRecordset(camada.records);
       while not recs.eof do
        begin
         flds := IMoFields(recs.fields);
         fld := IMoField(flds.item('ID'));
         s := fld.Value;
         strings2.add(s);
         recs.movenext;
        end;
       ren := IMoValueMapRenderer(CreateOleObject('MapObjects2.ValueMapRenderer'));
       ren.Field :='ID';
       ren.ValueCount :=strings2.count;
       for i :=0 to Strings2.Count-1 do
        begin
         ren.value[i]:= strings2.Item (i);
         case i of
          0: ren.Symbol[i].Color := clBlack;
          1: ren.Symbol[i].Color := moLightGray;
          2: ren.Symbol[i].Color := clBlack;
         end;
        end;
       camada.renderer := ren;
    end
   else
       raise Exception.Create('Arquivo(s) não encontrado(s)!');
  Camadas := Interface_Usuario.Form1.Mapa_Austin.Layers;
  Camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  rend := IMoLabelRenderer(CreateOleObject('MapObjects2.LabelRenderer'));
  DCtexto := IMoDataConnection(CreateOleObject('MapObjects2.DataConnection'));
  DCtexto.Database := '[CADText]d:\mestrado\tese\Aplicativo';
  if (DCtexto.Connect)
  then begin
         Camada.GeoDataset := IMoGeoDataset(DCtexto.FindGeoDataset('text_log_reg_austin.dwg'));
         simbolo := Camada.Symbol;
         simbolo.Color := clBlack;
         Camadas.Add(Camada);
        end
  else raise Exception.Create('Arquivo da camada texto não encontrado!');
  Camada := IMoMapLayer(Camadas.Item('area'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('escolas'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('ferrovia_inter'));
  Camada.Visible := false;
```

```
  Camada := IMoMapLayer(Camadas.Item('rios_inter'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('text_log_reg_austin.dwg'));
  rend.HeightField := 'HeightText';
  rend.Field := 'Text';
  Camada.Renderer := rend;
  Camada.Visible := false;
  Mapa_Austin.refresh;
{ iniciar quantidade de quadras por linha }
  Qt_quadras_por_linha[1] := 5;
  Qt_quadras_por_linha[2] := 6;
  Qt_quadras_por_linha[3] := 7;
  for i := 4 to 8 do Qt_quadras_por_linha[i] := 8;
  Qt_quadras_por_linha[9] := 7;
  Qt_quadras_por_linha[10] := 5;
  EstouLocalizando := false;
  EstouAtualizandoLocal := false;
  EstouSelecionandoQuadra := false;
  SelecioneiQuadra := false;
  SelecioneiEscola := false;
  Form1Iniciado := false;
end;

procedure Aprox;

begin
  if not Form1.Aproximar.Checked
  then begin
         Form1.Aproximar.Checked := true;
         Form1.Afastar.Enabled := false;
         Form1.Deslocar.Enabled := false;
         Form1.AproximarporSel.Enabled := false;
         Form1.BotaoZoomAfastar.Enabled := false;
         Form1.BotaoDeslocar.Enabled := false;
         Form1.BotaoZoomSelec.Enabled := false;
       end
  else begin
         Form1.Aproximar.Checked := false;
         Form1.Afastar.Enabled := true;
         Form1.Deslocar.Enabled := true;
         Form1.AproximarporSel.Enabled := true;
         Form1.BotaoZoomAfastar.Enabled := true;
         Form1.BotaoDeslocar.Enabled := true;
         Form1.BotaoZoomSelec.Enabled := true;
       end;
end;

procedure TForm1.AproximarProc(Sender: TObject);

begin
  Aprox;
end;

procedure Afast;

begin
  if not Form1.Afastar.Checked
  then begin
         Form1.Afastar.Checked := true;
         Form1.Aproximar.Enabled := false;
         Form1.AproximarporSel.Enabled := false;
         Form1.Deslocar.Enabled := false;
         Form1.BotaoZoomAproximar.Enabled := false;
         Form1.BotaoDeslocar.Enabled := false;
         Form1.BotaoZoomSelec.Enabled := false;
       end
  else begin
         Form1.Afastar.Checked := false;
         Form1.Aproximar.Enabled := true;
         Form1.AproximarporSel.Enabled := true;
         Form1.Deslocar.Enabled := true;
         Form1.BotaoZoomAproximar.Enabled := true;
         Form1.BotaoDeslocar.Enabled := true;
         Form1.BotaoZoomSelec.Enabled := true;
       end;
end;

procedure TForm1.AfastarProc(Sender: TObject);

begin
  Afast;
end;

procedure TForm1.ZoomPanProc(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);

var
  pmapa : Point;
  ret   : IMoRectangle;
  l     : IMoMapLayer;
  recs  : IMoRecordset;
  flds  : IMoFields;
  fld   : IMoField;

begin
 if Button = mbLeft
 then if AproximarporSel.Checked
      then Mapa_Austin.Extent := Mapa_Austin.TrackRectangle
      else if Deslocar.Checked
           then Mapa_Austin.Pan
           else if Afastar.Checked
                then begin
                       ret := Mapa_Austin.Extent;
                       ret.ScaleRectangle (1.25);
                       Mapa_Austin.Extent := ret;
                     end
                else if Aproximar.Checked
```

```
                    then begin
                          ret := Mapa_Austin.Extent;
                          ret.ScaleRectangle (0.85);
                          Mapa_Austin.Extent := ret;
                         end;
 if ((Button = mbRight) and EstouLocalizando) or
    ((Button = mbRight) and EstouAtualizandoLocal)
 then begin
        pmapa := Mapa_Austin.ToMapPoint(X, Y);
        CadastrarMatricula.TabelaMatri.FieldValues['COORD_X'] := pmapa.x;
        CadastrarMatricula.TabelaMatri.FieldValues['COORD_Y'] := pmapa.y;
        CadastrarMatricula.TabelaMatri.Post;
        CadastrarMatricula.TabelaMatri.first;
        CadastrarMatricula.Botao_Cancelar.Enabled := true;
        if EstouAtualizandoLocal
        then begin
              CadastrarMatricula.Botao_Aplicar.Enabled := true;
              CadastrarMatricula.Botao_AtualizarLocalizacao.Enabled := false;
              showmessage ('Localização do aluno atualizada.');
              EstouAtualizandoLocal := false;
              CadastrarMatricula.Botao_Aplicar.Visible := false;
              CadastrarMatricula.Botao_AtualizarLocalizacao.Visible := false;
              CadastrarMatricula.Botao_Cancelar.Caption := 'Cancelar';
              CadastrarMatricula.Caption := 'Cadastrar Pré-Matrícula';
              CadastrarMatricula.Botao_Cancelar.Visible := true;
              CadastrarMatricula.Botao_cadastrar.Visible := true;
             end
        else begin
              CadastrarMatricula.Botao_cadastrar.Enabled := true;
              CadastrarMatricula.Botao_Localizar.Enabled := false;
              showmessage ('Aluno localizado.');
              EstouLocalizando := false;
             end;
        Matrcula1.Enabled := true;
        Consultas.Enabled := true;
        BD.Enabled := true;
        Relatrios1.Enabled := true;
       end;
 if (Button = mbLeft) and (ssCtrl in shift)
    and EstouSelecionandoQuadra
 then begin
       l := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
       l := IMoMapLayer(Mapa_Austin.layers.item('area'));
       pmapa := IMoPoint(CreateOleObject('MapObjects2.Point'));
       pmapa := Mapa_Austin.ToMapPoint(x,y);
       recs := l.SearchShape(pmapa,12,'');
       if not recs.eof
       then begin
             sym := IMoSymbol(CreateComObject(Class_Symbol));
//             sym.OutlineColor := clgray;
             sym.Color := clgray;
             flds := recs.Fields;
             shp := IMoPolygon(CreateOleObject('MapObjects2.Polygon'));
             shp := IMoPolygon(IDispatch(flds.Item('Shape').Value));
             fld := IMoField(flds.item('UNIDADE'));
             unidade := fld.value;
             SelecioneiQuadra := true;
             Mapa_Austin.Refresh;
            end;
      end;
 if (Button = mbRight) and (ssCtrl in shift)
    and EstouSelecionandoQuadra
 then Mapa_Austin.Refresh;
 if SpeedButton2.Enabled and SpeedButton2.Down
 then begin
       UnitDescEsc.DescritorEscola.BorderIcons := UnitDescEsc.DescritorEscola.BorderIcons - [biMaximize];
       UnitDescEsc.DescritorEscola.BorderIcons := UnitDescEsc.DescritorEscola.BorderIcons - [biHelp];
       UnitDescEsc.DescritorEscola.BorderIcons := UnitDescEsc.DescritorEscola.BorderIcons - [biMinimize];
       UnitDescEsc.DescritorEscola.Show;
       UnitDescEsc.DescritorEscola.StringGrid1.DefaultDrawing := false;
       MostraInfoEscola (X,Y);
      end;
 if SpeedButton3.Enabled
 then begin
        SelecioneiEscola := true;
        Mapa_Austin.Refresh;
       end;
end;

procedure Desloc;

begin
  if not Form1.Deslocar.Checked
  then begin
         Form1.Deslocar.Checked := true;
         Form1.Aproximar.Enabled := false;
         Form1.Afastar.Enabled := false;
         Form1.AproximarporSel.Enabled := false;
         Form1.BotaoZoomAfastar.Enabled := false;
         Form1.BotaoZoomAproximar.Enabled := false;
         Form1.BotaoZoomSelec.Enabled := false;
       end
  else begin
         Form1.Deslocar.Checked := false;
         Form1.Aproximar.Enabled := true;
         Form1.Afastar.Enabled := true;
         Form1.AproximarporSel.Enabled := true;
         Form1.BotaoZoomAfastar.Enabled := true;
         Form1.BotaoZoomAproximar.Enabled := true;
         Form1.BotaoZoomSelec.Enabled := true;
       end;
end;

procedure TForm1.DeslocarProc(Sender: TObject);

begin
  Desloc;
```

```
end;

procedure AproxporSelec;

begin
  if not Form1.AproximarporSel.Checked
  then begin
         Form1.Deslocar.Enabled := false;
         Form1.Aproximar.Enabled := false;
         Form1.Afastar.Enabled := false;
         Form1.AproximarporSel.Checked := true;
         Form1.BotaoZoomAfastar.Enabled := false;
         Form1.BotaoDeslocar.Enabled := false;
         Form1.BotaoZoomAproximar.Enabled := false;
       end
  else begin
         Form1.Deslocar.Enabled := true;
         Form1.Aproximar.Enabled := true;
         Form1.Afastar.Enabled := true;
         Form1.AproximarporSel.Checked := false;
         Form1.BotaoZoomAfastar.Enabled := true;
         Form1.BotaoDeslocar.Enabled := true;
         Form1.BotaoZoomAproximar.Enabled := true;
       end;
end;

procedure TForm1.AproximarporSelProc (Sender: TObject);

begin
  AproxporSelec;
end;

procedure TForm1.ZoomLogradouroProc(Sender: TObject);

begin
  SelecLog.Visible := true;
end;

procedure TForm1.CadastrarProc(Sender: TObject);

var
   inicio, fim : TDateTime;

begin
 inicio := TabCal.FieldValues['INICIO'];
 fim := TabCal.FieldValues['FIM'];
 if (date >= inicio) and (date <= fim)
 then begin
       CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biMaximize];
       CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biHelp];
       CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biMinimize];
       CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biSystemMenu];
       CadastrarMatricula.Edit_Nome.Clear;
       CadastrarMatricula.Edit_email.Text := 'usuario@dominio';
       CadastrarMatricula.Edit_Pai.Clear;
       CadastrarMatricula.Edit_Mae.Clear;
       CadastrarMatricula.EditData.Clear;
       CadastrarMatricula.Edit_Rua.Clear;
       CadastrarMatricula.Edit_Numero.Clear;
       CadastrarMatricula.Edit_Comp.Text := 'sem complemento';
       CadastrarMatricula.Edit_Bairro.Clear;
       CadastrarMatricula.Edit_Cidade.Text := 'Nova Iguaçu';
       CadastrarMatricula.Combo_Estado.Text := 'Rio de Janeiro - RJ';
       CadastrarMatricula.Combo_Serie.Text := '';
       CadastrarMatricula.Edit_CEP.Clear;
       CadastrarMatricula.Edit_TEL1.Clear;
       CadastrarMatricula.Edit_TEL2.Clear;
       CadastrarMatricula.Visible := true;
       Form1.Enabled := false;
      end
 else showmessage ('Fora do prazo para fazer a matrícula.');
end;

procedure TForm1.AtualizarPreMatriClick(Sender: TObject);

var
   inicio, fim : TDateTime;

begin
 inicio := TabCal.FieldValues['INICIO'];
 fim := TabCal.FieldValues['FIM'];
 if (date >= inicio) and (date <= fim)
 then begin
        ExcluindoPreMatri := false;
        SelecCad.Visible := true;
      end
 else showmessage ('Fora do prazo para atualizar a matrícula.');
end;

procedure TForm1.ExcluirCadastro1Click(Sender: TObject);

var
   inicio, fim : TDateTime;

begin
 inicio := TabCal.FieldValues['INICIO'];
 fim := TabCal.FieldValues['FIM'];
 if (date >= inicio) and (date <= fim)
 then begin
        ExcluindoPreMatri := true;
        SelecCad.Visible := true;
      end
 else showmessage ('Fora do prazo para excluir a matrícula.');
end;

procedure TForm1.DefinirCalendClick(Sender: TObject);
```

```
begin
  DefinirCalendario.BorderIcons := DefinirCalendario.BorderIcons - [biMaximize];
  DefinirCalendario.BorderIcons := DefinirCalendario.BorderIcons - [biHelp];
  DefinirCalendario.BorderIcons := DefinirCalendario.BorderIcons - [biMinimize];
  DefinirCalendario.BorderIcons := DefinirCalendario.BorderIcons - [biSystemMenu];
  Form1.Enabled := false;
  DefinirCalendario.EditDataInicial.Clear;
  DefinirCalendario.EditDataFinal.Clear;
  DefinirCalendario.Visible := true;
end;

procedure TForm1.ConsultaAlunoClick(Sender: TObject);

begin
  ConsultAlun.ConsultaAluno.ComboPalavraChave.Clear;
  ConsultAlun.ConsultaAluno.BorderIcons := ConsultAlun.ConsultaAluno.BorderIcons - [biMaximize];
  ConsultAlun.ConsultaAluno.BorderIcons := ConsultAlun.ConsultaAluno.BorderIcons - [biHelp];
  ConsultAlun.ConsultaAluno.BorderIcons := ConsultAlun.ConsultaAluno.BorderIcons - [biMinimize];
  ConsultAlun.ConsultaAluno.BorderIcons := ConsultAlun.ConsultaAluno.BorderIcons - [biSystemMenu];
  ConsultAlun.ConsultaAluno.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.ConsultaEscolaClick(Sender: TObject);

begin
  ConsultEsc.ConsultaEscola.ComboPalavraChave.Clear;
  ConsultEsc.ConsultaEscola.BorderIcons := ConsultEsc.ConsultaEscola.BorderIcons - [biMaximize];
  ConsultEsc.ConsultaEscola.BorderIcons := ConsultEsc.ConsultaEscola.BorderIcons - [biHelp];
  ConsultEsc.ConsultaEscola.BorderIcons := ConsultEsc.ConsultaEscola.BorderIcons - [biMinimize];
  ConsultEsc.ConsultaEscola.BorderIcons := ConsultEsc.ConsultaEscola.BorderIcons - [biSystemMenu];
  ConsultEsc.ConsultaEscola.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.ConsultaProfessorClick(Sender: TObject);

begin
  ConsultProf.ConsultaProfessor.ComboPalavraChave.Clear;
  ConsultProf.ConsultaProfessor.BorderIcons := ConsultProf.ConsultaProfessor.BorderIcons - [biMaximize];
  ConsultProf.ConsultaProfessor.BorderIcons := ConsultProf.ConsultaProfessor.BorderIcons - [biHelp];
  ConsultProf.ConsultaProfessor.BorderIcons := ConsultProf.ConsultaProfessor.BorderIcons - [biMinimize];
  ConsultProf.ConsultaProfessor.BorderIcons := ConsultProf.ConsultaProfessor.BorderIcons - [biSystemMenu];
  ConsultProf.ConsultaProfessor.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.ConsultaCalendarioClick(Sender: TObject);

begin
  ConsultCalen.ConsultaCalendario.BorderIcons := ConsultCalen.ConsultaCalendario.BorderIcons - [biMaximize];
  ConsultCalen.ConsultaCalendario.BorderIcons := ConsultCalen.ConsultaCalendario.BorderIcons - [biHelp];
  ConsultCalen.ConsultaCalendario.BorderIcons := ConsultCalen.ConsultaCalendario.BorderIcons - [biMinimize];
  ConsultCalen.ConsultaCalendario.BorderIcons := ConsultCalen.ConsultaCalendario.BorderIcons - [biSystemMenu];
  ConsultCalen.ConsultaCalendario.Visible := true;
  Form1.Enabled := false;
end;

function ExisteTurmaAberta (escola : string) : boolean;

var
   acheiAberta : boolean;

begin
  Form1.TabTurma.first;
  acheiAberta := false;
  while not acheiAberta and not Form1.TabTurma.Eof do
    begin
      if (Form1.TabTurma.FieldValues['COD_ESC'] = escola) and
         (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] =
          Form1.TabTurma.FieldValues['SERIE']) and
         (Form1.TabTurma.FieldValues['ABERTA'] = -1)
      then begin
            acheiAberta := true;
            DadosPARAmatricular.CodigoTurma :=
              Form1.TabTurma.FieldValues['COD_TURM'];
           end
      else Form1.TabTurma.next;
    end;
  DadosPARAmatricular.TurmaAberta := acheiAberta;
  ExisteTurmaAberta := acheiAberta;
end;

function PrimeiroSegSerie (Serie : string) : boolean;

var
   SegundoSeg : boolean;

begin
  SegundoSeg := (Serie = 'Quinta Série') or (Serie = 'Sexta Série') or
                (Serie = 'Sétima Série') or (Serie = 'Oitava Série');
  PrimeiroSegSerie := not SegundoSeg;
end;

function TemProfDisc
         (escola, disciplina : string; var cod_prof : cardinal) : boolean;

var
   AcheiProf : boolean;

begin
  Form1.TabProfEsc.first;
  AcheiProf := false;
  while not AcheiProf and not Form1.TabProfEsc.Eof do
    begin
      if (Form1.TabProfEsc.FieldValues['Cod_Esc'] = escola) and
         (Form1.TabProfEsc.FieldValues['Disciplina'] = disciplina) and
```

```
        (Form1.TabProfEsc.FieldValues['Turmas_Aloc'] <
         Form1.TabProfEsc.FieldValues['Max_Turmas'])
      then begin
             AcheiProf := true;
             cod_prof := Form1.TabProfEsc.FieldValues['Cod_Prof'];
           end
      else Form1.TabProfEsc.Next;
    end;
  TemProfDisc := AcheiProf;
end;

function TemProfPrimSeg (escola : string) : boolean;

begin
  TemProfPrimSeg :=
    TemProfDisc
    (escola,'Primeiro Segmento',DadosPARAmatricular.CodigoProfPrimSeg);
end;

function TemProfsSegSeg (escola : string) : boolean;

var
   TemProfs : boolean;

begin
  TemProfs := TemProfDisc
              (escola,'Matemática',DadosPARAmatricular.CodigoProfMatematica);
  if TemProfs
  then TemProfs := TemProfDisc
                   (escola,'Português',DadosPARAmatricular.CodigoProfPortugues);
  if TemProfs
  then TemProfs := TemProfDisc
                   (escola,'Ciências',DadosPARAmatricular.CodigoProfCiencias);
  if TemProfs
  then TemProfs := TemProfDisc
                   (escola,'História',DadosPARAmatricular.CodigoProfHistoria);
  if TemProfs
  then TemProfs := TemProfDisc
                   (escola,'Geografia',DadosPARAmatricular.CodigoProfGeografia);
  if TemProfs
  then TemProfs := TemProfDisc
                   (escola,'Inglês',DadosPARAmatricular.CodigoProfIngles);
  if TemProfs
  then TemProfs := TemProfDisc
              (escola,'Educação Física',DadosPARAmatricular.CodigoProfEduFisica);
  if TemProfs
  then TemProfs := TemProfDisc
                   (escola,'Artes',DadosPARAmatricular.CodigoProfArtes);
  TemProfsSegSeg := TemProfs;
end;

function PossoCriarTurma (escola : string) : boolean;

var
   posso : boolean;

begin
  if not ConsultEsc.ConsultaEscola.TabEscola.Locate ('COD_ESC',escola,[])
  then begin
         showmessage ('Erro no banco de dados ESCOLA');
         halt;
       end;
  posso :=
    ConsultEsc.ConsultaEscola.TabEscola.FieldValues['SALAS_ALOC'] <
    (ConsultEsc.ConsultaEscola.TabEscola.FieldValues['NUM_TURNOS']
     *ConsultEsc.ConsultaEscola.TabEscola.FieldValues['NUM_SALAS']);
//    2*ConsultEsc.ConsultaEscola.TabEscola.FieldValues['NUM_SALAS'];
  if posso
  then if PrimeiroSegSerie (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'])
       then posso := TemProfPrimSeg (escola)
       else posso := TemProfsSegSeg (escola);
  PossoCriarTurma := posso;
end;

function EscolaValida (escola : string) : boolean;

var
   valida : boolean;

begin
  if not ConsultEsc.ConsultaEscola.TabEscola.Locate ('COD_ESC',escola,[])
  then begin
         showmessage ('Erro no banco de dados ESCOLA');
         halt;
       end;
  valida := true;
  if (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Creche')
     and (escola[1] <> 'C')
  then valida := false;
  if valida and
     (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] <> 'Creche')
     and (escola[1] = 'C')
  then valida := false;
  if valida and
     (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Pré-Escolar')
     and (ConsultEsc.ConsultaEscola.TabEscola.FieldValues['EDU_INFANTIL'] = 0)
  then valida := false;
  if valida
  then if not ExisteTurmaAberta (escola)
       then if not PossoCriarTurma (escola)
            then valida := false;
  EscolaValida := valida;
end;

function GeraNomeTurma (escola : string) : string;

var
```

```
   nome_maior    : string[4];
   num_nome, cod : integer;
   PrimeiraTurma : boolean;

begin
  PrimeiraTurma := true;
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Creche'
  then nome_maior := 'c10';
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Pré-Escolar'
  then nome_maior := '51';
  if (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Primeira Etapa') or
     (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Segunda Etapa') or
     (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Terceira Etapa')
  then nome_maior := 'A';
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Terceira Série'
  then nome_maior := '301';
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Quarta Série'
  then nome_maior := '401';
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Quinta Série'
  then nome_maior := '501';
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Sexta Série'
  then nome_maior := '601';
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Sétima Série'
  then nome_maior := '701';
  if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Oitava Série'
  then nome_maior := '801';
  Form1.TabTurma.first;
  while not Form1.TabTurma.eof do
   begin
    if (Form1.TabTurma.FieldValues['SERIE'] =
        CadastrarMatricula.TabelaMatri.FieldValues['SERIE']) and
       (Form1.TabTurma.FieldValues['COD_ESC'] = escola)
    then begin
          PrimeiraTurma := false;
          if Form1.TabTurma.FieldValues['NOME'] > nome_maior
          then nome_maior := Form1.TabTurma.FieldValues['NOME'];
         end;
    Form1.TabTurma.Next;
   end;
  if not PrimeiraTurma
  then if CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Creche'
       then begin
             val (copy (nome_maior,2,2),num_nome,cod);
             inc (num_nome);
             str (num_nome,nome_maior);
             nome_maior := 'c' + nome_maior;
            end
       else if (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Primeira Etapa') or
               (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Segunda Etapa') or
               (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'] = 'Terceira Etapa')
            then begin
                  cod := ord (nome_maior[1]);
                  inc (cod);
                  nome_maior[1] := chr (cod);
                 end
            else begin
                  val (nome_maior,num_nome,cod);
                  inc (num_nome);
                  str (num_nome,nome_maior);
                 end;
  GeraNomeTurma := nome_maior;
end;

procedure AtualizarTabTurma (escola : string);

var
   Cart_Aloc, CodigoTurma : cardinal;
   nome                   : string[4];

begin
  if not ConsultEsc.ConsultaEscola.TabEscola.Locate ('COD_ESC',escola,[])
  then begin
        showmessage ('Erro no banco de dados ESCOLA');
        halt;
       end;
  if DadosPARAmatricular.TurmaAberta
  then begin
       if not Form1.TabTurma.Locate
              ('COD_TURM',DadosPARAmatricular.CodigoTurma,[])
       then begin
             showmessage ('Erro no banco de dados TURMA');
             halt;
            end;
       Cart_Aloc := Form1.TabTurma.FieldValues['CART_ALOC'];
       inc (Cart_Aloc);
       Form1.TabTurma.Edit;
       Form1.TabTurma.FieldValues['CART_ALOC'] := Cart_Aloc;
       if Cart_Aloc = ConsultEsc.ConsultaEscola.TabEscola.FieldValues['NUM_CART']
       then Form1.TabTurma.FieldValues['ABERTA'] := 0;
      end
  else begin
       nome := GeraNomeTurma (escola);
       CodigoTurma := 0;
       if Form1.TabTurma.RecordCount <> 0
       then Form1.TabTurma.first;
            while not Form1.TabTurma.eof do
             begin
              if Form1.TabTurma.FieldValues['COD_TURM'] > CodigoTurma
              then CodigoTurma := Form1.TabTurma.FieldValues['COD_TURM'];
              Form1.TabTurma.Next;
             end;
       inc (CodigoTurma);
       DadosPARAmatricular.CodigoTurma := CodigoTurma;
       Form1.TabTurma.Append;
       Form1.TabTurma.FieldValues['COD_TURM'] := CodigoTurma;
       Form1.TabTurma.FieldValues['SERIE'] :=
         CadastrarMatricula.TabelaMatri.FieldValues['SERIE'];
```

```
        Form1.TabTurma.FieldValues['COD_ESC'] := escola;
        Form1.TabTurma.FieldValues['NOME'] := nome;
        Form1.TabTurma.FieldValues['ABERTA'] := -1;
        Form1.TabTurma.FieldValues['CART_ALOC'] := 1;
       end;
  Form1.TabTurma.Post;
end;

procedure AtualizarTabEscola (escola : string);

var
   Salas_Aloc : cardinal;

begin
 if not DadosPARAmatricular.TurmaAberta
 then begin
       if not ConsultEsc.ConsultaEscola.TabEscola.Locate ('COD_ESC',escola,[])
       then begin
             showmessage ('Erro no banco de dados ESCOLA');
             halt;
       end;
       ConsultEsc.ConsultaEscola.TabEscola.Edit;
       Salas_Aloc := ConsultEsc.ConsultaEscola.TabEscola.FieldValues['SALAS_ALOC'];
       inc (Salas_Aloc);
       ConsultEsc.ConsultaEscola.TabEscola.FieldValues['SALAS_ALOC'] := Salas_Aloc;
       ConsultEsc.ConsultaEscola.TabEscola.Post;
      end;
end;

procedure AtualizarTurmasAlocProf (escola : string; prof : cardinal);

var
   AcheiProfEsc : boolean;
   Turmas_Aloc  : cardinal;

begin
  Form1.TabProfEsc.first;
  AcheiProfEsc := false;
  while not AcheiProfEsc and not Form1.TabProfEsc.Eof do
    begin
      if (Form1.TabProfEsc.FieldValues['Cod_Esc'] = escola) and
         (Form1.TabProfEsc.FieldValues['Cod_Prof'] = prof)
      then begin
             AcheiProfEsc := true;
             Turmas_Aloc := Form1.TabProfEsc.FieldValues['Turmas_Aloc'];
             inc (Turmas_Aloc);
             Form1.TabProfEsc.Edit;
             Form1.TabProfEsc.FieldValues['Turmas_Aloc'] := Turmas_Aloc;
             Form1.TabProfEsc.Post;
           end
      else Form1.TabProfEsc.Next;
    end;
end;

procedure AtualizarTabProfEsc (escola : string);

begin
  if PrimeiroSegSerie (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'])
  then AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfPrimSeg)
  else begin
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfMatematica);
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfPortugues);
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfCiencias);
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfHistoria);
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfGeografia);
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfEduFisica);
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfIngles);
        AtualizarTurmasAlocProf (escola,DadosPARAmatricular.CodigoProfArtes);
       end;
end;

procedure AtualizarTabProfTurma;

begin
  if not DadosPARAmatricular.TurmaAberta
  then begin
         if PrimeiroSegSerie (CadastrarMatricula.TabelaMatri.FieldValues['SERIE'])
         then begin
                Form1.TabProfTurma.Append;
                Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                  DadosPARAmatricular.CodigoProfPrimSeg;
                Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                  DadosPARAmatricular.CodigoTurma;
              end
         else begin
                Form1.TabProfTurma.Append;
                Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                  DadosPARAmatricular.CodigoProfMatematica;
                Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                  DadosPARAmatricular.CodigoTurma;
                Form1.TabProfTurma.Append;
                Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                  DadosPARAmatricular.CodigoProfPortugues;
                Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                  DadosPARAmatricular.CodigoTurma;
                Form1.TabProfTurma.Append;
                Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                  DadosPARAmatricular.CodigoProfCiencias;
                Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                  DadosPARAmatricular.CodigoTurma;
                Form1.TabProfTurma.Append;
                Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                  DadosPARAmatricular.CodigoProfHistoria;
                Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                  DadosPARAmatricular.CodigoTurma;
                Form1.TabProfTurma.Append;
                Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
```

```
                DadosPARAmatricular.CodigoProfGeografia;
              Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                DadosPARAmatricular.CodigoTurma;
              Form1.TabProfTurma.Append;
              Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                DadosPARAmatricular.CodigoProfEduFisica;
              Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                DadosPARAmatricular.CodigoTurma;
              Form1.TabProfTurma.Append;
              Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                DadosPARAmatricular.CodigoProfIngles;
              Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                DadosPARAmatricular.CodigoTurma;
              Form1.TabProfTurma.Append;
              Form1.TabProfTurma.FieldValues['Cod_Prof'] :=
                DadosPARAmatricular.CodigoProfArtes;
              Form1.TabProfTurma.FieldValues['Cod_Turma'] :=
                DadosPARAmatricular.CodigoTurma;
            end;
        Form1.TabProfTurma.Post;
      end;
end;

procedure AtualizarTabAluno;

begin
  CadastrarMatricula.TabelaMatri.Edit;
  CadastrarMatricula.TabelaMatri.FieldValues['COD_TURM'] :=
    DadosPARAmatricular.CodigoTurma;
  CadastrarMatricula.TabelaMatri.FieldValues['DISTANCIA'] :=
    DadosPARAmatricular.Distancia;
  CadastrarMatricula.TabelaMatri.FieldValues['AREA_INFLU'] :=
    DadosPARAmatricular.AreaDEinfluencia;
  CadastrarMatricula.TabelaMatri.Post;
end;

procedure MatricularAlunoAtual (escola : string);

begin
  AtualizarTabTurma (escola);
  AtualizarTabEscola (escola);
  if not DadosPARAmatricular.TurmaAberta
  then AtualizarTabProfEsc (escola);
  AtualizarTabProfTurma;
  AtualizarTabAluno;
end;

procedure ApagarProcessamentoAnterior;

var
   i,nregs : cardinal;

begin
  nregs := Form1.TabTurma.RecordCount;
  if nregs > 0
  then begin
         Form1.TabTurma.first;
         for i := 1 to nregs do
           begin
             Form1.TabTurma.Delete;
             Form1.TabTurma.next;
           end;
       end;
  nregs := Form1.TabProfTurma.RecordCount;
  if nregs > 0
  then begin
         Form1.TabProfTurma.first;
         for i := 1 to nregs do
           begin
             Form1.TabProfTurma.Delete;
             Form1.TabProfTurma.next;
           end;
       end;
  nregs := ConsultEsc.ConsultaEscola.TabEscola.RecordCount;
  if nregs > 0
  then begin
         ConsultEsc.ConsultaEscola.TabEscola.first;
         for i := 1 to nregs do
           begin
             ConsultEsc.ConsultaEscola.TabEscola.Edit;
             ConsultEsc.ConsultaEscola.TabEscola.FieldValues['SALAS_ALOC'] := 0;
             ConsultEsc.ConsultaEscola.TabEscola.Post;
             ConsultEsc.ConsultaEscola.TabEscola.Next;
           end;
       end;
  nregs := CadastrarMatricula.TabelaMatri.RecordCount;
  if nregs > 0
  then begin
         CadastrarMatricula.TabelaMatri.first;
         for i := 1 to nregs do
           begin
             CadastrarMatricula.TabelaMatri.Edit;
             CadastrarMatricula.TabelaMatri.FieldValues['COD_TURM'] := 0;
             CadastrarMatricula.TabelaMatri.FieldValues['DISTANCIA'] := 0;
             CadastrarMatricula.TabelaMatri.FieldValues['AREA_INFLU'] := 0;
             CadastrarMatricula.TabelaMatri.Post;
             CadastrarMatricula.TabelaMatri.Next;
           end;
       end;
  nregs := Form1.TabProfEsc.RecordCount;
  if nregs > 0
  then begin
         Form1.TabProfEsc.first;
         for i := 1 to nregs do
           begin
             Form1.TabProfEsc.Edit;
             Form1.TabProfEsc.FieldValues['Turmas_Aloc'] := 0;
```

```
            Form1.TabProfEsc.Post;
            Form1.TabProfEsc.Next;
          end;
      end;
end;

procedure TForm1.MatriProceTurmaClick(Sender: TObject);

var
   fim                    : TDateTime;
   ListaDEescolasInvalidas : TStrings;
   matriculei             : boolean;
   escolaMaisPerto        : string[8];
   MenorDistancia         : double;
   PontoAluno,PontoEscola  : Point;

begin
 fim := TabCal.FieldValues['FIM'];
 if date > fim
 then begin
       ProgressBarProcTur.Visible := true;
       NProcessandoTurmas.Visible := true;
       showmessage ('Se houver, o processamento de turmas anterior será apagado.');
       ApagarProcessamentoAnterior;
       PontoAluno := Point(CreateOleObject('MapObjects2.Point'));
       PontoEscola := Point(CreateOleObject('MapObjects2.Point'));
       ListaDEescolasInvalidas := TStringList.Create;
       CadastrarMatricula.TabelaMatri.First;
       ProgressBarProcTur.Max := CadastrarMatricula.TabelaMatri.RecordCount;
       ProgressBarProcTur.Min := 0;
       while not CadastrarMatricula.TabelaMatri.Eof do
         begin
           ListaDEescolasInvalidas.Clear;
           matriculei := false;
           PontoAluno.X := CadastrarMatricula.TabelaMatri.FieldValues['COORD_X'];
           PontoAluno.Y := CadastrarMatricula.TabelaMatri.FieldValues['COORD_Y'];
           repeat
            MenorDistancia :=
             sqrt (sqr(Mapa_Austin.FullExtent.Top - Mapa_Austin.FullExtent.Bottom)
                 + sqr(Mapa_Austin.FullExtent.Right - Mapa_Austin.FullExtent.Left));
            ConsultEsc.ConsultaEscola.TabEscola.first;
            while not ConsultEsc.ConsultaEscola.TabEscola.eof do
              begin
                if ListaDEescolasInvalidas.IndexOf
                   (ConsultEsc.ConsultaEscola.TabEscola.FieldValues['COD_ESC']) = -1
                then begin
                      PontoEscola.X :=
                       ConsultEsc.ConsultaEscola.TabEscola.FieldValues['COORD_X'];
                      PontoEscola.Y :=
                       ConsultEsc.ConsultaEscola.TabEscola.FieldValues['COORD_Y'];
                      if PontoAluno.DistanceTo(PontoEscola) < MenorDistancia
                      then begin
                            MenorDistancia := PontoAluno.DistanceTo(PontoEscola);
                            escolaMaisPerto :=
                             ConsultEsc.ConsultaEscola.TabEscola.FieldValues['COD_ESC'];
                           end;
                    end;
                ConsultEsc.ConsultaEscola.TabEscola.next;
              end;
            if EscolaValida (escolaMaisPerto)
            then begin
                   DadosPARAmatricular.Distancia := MenorDistancia;
                   if MenorDistancia <= 1000
                   then DadosPARAmatricular.AreaDEinfluencia := -1
                   else DadosPARAmatricular.AreaDEinfluencia := 0;
                   MatricularAlunoAtual (escolaMaisPerto);
                   Matriculei := true;
                 end
            else ListaDEescolasInvalidas.Append (escolaMaisPerto);
           until matriculei or
                (ConsultEsc.ConsultaEscola.TabEscola.RecordCount =
                 ListaDEescolasInvalidas.Count);
           CadastrarMatricula.TabelaMatri.next;
           ProgressBarProcTur.Position := ProgressBarProcTur.Position + 1;
         end;
        showmessage ('Turmas processadas!');
        ProgressBarProcTur.Visible := false;
        NProcessandoTurmas.Visible := false;
      end
 else showmessage ('Ainda não está no prazo para processar as turmas');
end;

procedure TForm1.ConsultTurmasProcessadasClick(Sender: TObject);

begin
  if Form1.TabTurma.RecordCount = 0
  then showmessage('Não há turmas processadas.')
  else begin
         ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons :=
           ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons - [biMaximize];
         ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons :=
           ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons - [biHelp];
         ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons :=
           ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons - [biMinimize];
         ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons :=
           ConsultTurmProc.ConsultaTurmasProcessadas.BorderIcons - [biSystemMenu];
         ConsultTurmProc.ConsultaTurmasProcessadas.ComboPalavraChave.Clear;
         ConsultTurmProc.LimpaCampos;
         ConsultTurmProc.ConsultaTurmasProcessadas.ComboNomeTurma.Clear;
         ConsultTurmProc.ConsultaTurmasProcessadas.Visible := true;
         Form1.Enabled := false;
       end;
end;

procedure TForm1.PreMatriculaRelatClick(Sender: TObject);

begin
```

```
  RelPreMatri.RelatoriosPreMatricula.EditDe.Clear;
  RelPreMatri.RelatoriosPreMatricula.EditAte.Clear;
  RelPreMatri.RelatoriosPreMatricula.RadioGroupRelat.ItemIndex := 0;
  RelPreMatri.RelatoriosPreMatricula.BorderIcons :=
    RelPreMatri.RelatoriosPreMatricula.BorderIcons - [biMaximize];
  RelPreMatri.RelatoriosPreMatricula.BorderIcons :=
    RelPreMatri.RelatoriosPreMatricula.BorderIcons - [biHelp];
  RelPreMatri.RelatoriosPreMatricula.BorderIcons :=
    RelPreMatri.RelatoriosPreMatricula.BorderIcons - [biMinimize];
  RelPreMatri.RelatoriosPreMatricula.BorderIcons :=
    RelPreMatri.RelatoriosPreMatricula.BorderIcons - [biSystemMenu];
  RelPreMatri.RelatoriosPreMatricula.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.BasedeDadosRelatClick(Sender: TObject);

begin
  RelBD.RelatoriosBD.EditDe.Clear;
  RelBD.RelatoriosBD.EditAte.Clear;
  RelBD.RelatoriosBD.RadioGroupRelat.ItemIndex := 0;
  RelBD.RelatoriosBD.BorderIcons := RelBD.RelatoriosBD.BorderIcons - [biMaximize];
  RelBD.RelatoriosBD.BorderIcons := RelBD.RelatoriosBD.BorderIcons - [biHelp];
  RelBD.RelatoriosBD.BorderIcons := RelBD.RelatoriosBD.BorderIcons - [biMinimize];
  RelBD.RelatoriosBD.BorderIcons := RelBD.RelatoriosBD.BorderIcons - [biSystemMenu];
  RelBD.RelatoriosBD.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.TurmasProcessadasRelatClick(Sender: TObject);

begin
  if Form1.TabTurma.RecordCount = 0
  then showmessage('Não há turmas processadas.')
  else begin
        RelTurProc.RelatoriosTurmasProc.EditDe.Clear;
        RelTurProc.RelatoriosTurmasProc.EditAte.Clear;
        RelTurProc.RelatoriosTurmasProc.EditDeAluno.Clear;
        RelTurProc.RelatoriosTurmasProc.EditAteAluno.Clear;
        RelTurProc.RelatoriosTurmasProc.RadioGroupRelat.ItemIndex := 0;
        RelTurProc.RelatoriosTurmasProc.BorderIcons
          := RelTurProc.RelatoriosTurmasProc.BorderIcons - [biMaximize];
        RelTurProc.RelatoriosTurmasProc.BorderIcons
          := RelTurProc.RelatoriosTurmasProc.BorderIcons - [biHelp];
        RelTurProc.RelatoriosTurmasProc.BorderIcons
          := RelTurProc.RelatoriosTurmasProc.BorderIcons - [biMinimize];
        RelTurProc.RelatoriosTurmasProc.BorderIcons
          := RelTurProc.RelatoriosTurmasProc.BorderIcons - [biSystemMenu];
        RelTurProc.RelatoriosTurmasProc.Visible := true;
        Form1.Enabled := false;
       end;
end;

procedure TForm1.Panel1Resize(Sender: TObject);

begin
  Mapa_Austin.Width  := Panel1.Width;
  Mapa_Austin.Height := Panel1.Height;
  map1.Width := Panel1.Width;
  map1.Height := Panel1.Height;
end;

procedure TForm1.EscolaBDClick(Sender: TObject);

begin
  UnitEscolaBD.FormEscolaBD.ComboPalavraChave.Clear;
  UnitEscolaBD.EstouIncluindo := false;
  UnitEscolaBD.FormEscolaBD.BorderIcons := UnitEscolaBD.FormEscolaBD.BorderIcons - [biMaximize];
  UnitEscolaBD.FormEscolaBD.BorderIcons := UnitEscolaBD.FormEscolaBD.BorderIcons - [biHelp];
  UnitEscolaBD.FormEscolaBD.BorderIcons := UnitEscolaBD.FormEscolaBD.BorderIcons - [biMinimize];
  UnitEscolaBD.FormEscolaBD.BorderIcons := UnitEscolaBD.FormEscolaBD.BorderIcons - [biSystemMenu];
  UnitEscolaBD.FormEscolaBD.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.ProfessorBDClick(Sender: TObject);

begin
  UnitProfessorBD.FormProfessorBD.ComboPalavraChave.Clear;
  UnitProfessorBD.EstouIncluindo := false;
  UnitProfessorBD.FormProfessorBD.BorderIcons := UnitProfessorBD.FormProfessorBD.BorderIcons - [biMaximize];
  UnitProfessorBD.FormProfessorBD.BorderIcons := UnitProfessorBD.FormProfessorBD.BorderIcons - [biHelp];
  UnitProfessorBD.FormProfessorBD.BorderIcons := UnitProfessorBD.FormProfessorBD.BorderIcons - [biMinimize];
  UnitProfessorBD.FormProfessorBD.BorderIcons := UnitProfessorBD.FormProfessorBD.BorderIcons - [biSystemMenu];
  UnitProfessorBD.FormProfessorBD.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.LogradouroBDClick(Sender: TObject);

begin
  UnitLogradouroBD.FormLogradouroBD.ComboPalavraChave.Clear;
  UnitLogradouroBD.EstouIncluindo := false;
  UnitLogradouroBD.FormLogradouroBD.BorderIcons := UnitLogradouroBD.FormLogradouroBD.BorderIcons - [biMaximize];
  UnitLogradouroBD.FormLogradouroBD.BorderIcons := UnitLogradouroBD.FormLogradouroBD.BorderIcons - [biHelp];
  UnitLogradouroBD.FormLogradouroBD.BorderIcons := UnitLogradouroBD.FormLogradouroBD.BorderIcons - [biMinimize];
  UnitLogradouroBD.FormLogradouroBD.BorderIcons := UnitLogradouroBD.FormLogradouroBD.BorderIcons - [biSystemMenu];
  UnitLogradouroBD.FormLogradouroBD.Visible := true;
  Form1.Enabled := false;
end;

procedure TForm1.Mapa_AustinAfterLayerDraw(Sender: TObject;
  index: Smallint; canceled: WordBool; hDC: Cardinal);

begin
  if SelecioneiQuadra or SelecioneiEscola
  then begin
```

```
        Mapa_Austin.DrawShape (shp,sym);
        SelecioneiQuadra := false;
        SelecioneiEscola := false;
       end;
end;

procedure TForm1.BotaoConfSelQuadClick(Sender: TObject);

begin
  Form1.Enabled := false;
  UnitLogradouroBD.FormLogradouroBD.Enabled := true;
  UnitLogradouroBD.FormLogradouroBD.SetFocus;
  Form1.Matrcula1.Enabled := true;
  Form1.Consultas.Enabled := true;
  Form1.MenuMapas.Enabled := true;
  Form1.Exibir.Enabled := true;
  Form1.BD.Enabled := true;
  Form1.Relatrios1.Enabled := true;
  BotaoZoomSelec.Enabled := true;
  SpeedButton1.Enabled := true;
  BotaoZoomAproximar.Enabled := true;
  BotaoZoomAfastar.Enabled := true;
  BotaoDeslocar.Enabled := true;
  EstouSelecionandoQuadra := false;
  Selog.SelecLog.ADOTable1.Edit;
  Selog.SelecLog.ADOTable1.FieldValues['UNIDADE'] := unidade;
  Form1.BotaoConfSelQuad.Enabled := false;
end;

procedure TForm1.BotaoZoomSelecClick(Sender: TObject);

begin
  AproxporSelec;
end;

procedure TForm1.SpeedButton1Click(Sender: TObject);

begin
 Mapa_Austin.Extent := Mapa_Austin.FullExtent;
 if SpeedButton3.Enabled
 then begin
        SelecioneiEscola := true;
        Mapa_Austin.Refresh;
      end;
end;

procedure TForm1.BotaoZoomAproximarClick(Sender: TObject);

begin
  Aprox;
end;

procedure TForm1.BotaoZoomAfastarClick(Sender: TObject);

begin
  Afast;
end;

procedure TForm1.BotaoDeslocarClick(Sender: TObject);

begin
  Desloc;
end;

procedure TForm1.FormResize(Sender: TObject);

begin
  scrollbox1.Height := panel1.Height;
  scrollbox1.Left := panel1.Left + panel1.Width + 7;
  scrollbox1.Top := panel1.Top;
end;

procedure ApagarMapas;

begin
  Form1.Map1.Visible := false;
  Form1.label1.Visible := false;
  Form1.label2.Visible := false;
  Form1.PanelLegenda.Visible := false;
  Form1.Mapa_Austin.Visible := false;
  Form1.Exibir.Enabled := false;
  Form1.label14.Visible := false;
  Form1.panel3.Visible := false;
  Form1.BotaoZoomSelec.Enabled := false;
  Form1.SpeedButton1.Enabled := false;
  Form1.BotaoZoomAproximar.Enabled := false;
  Form1.BotaoZoomAfastar.Enabled := false;
  Form1.BotaoDeslocar.Enabled := false;
  Form1.Panel4.Visible := false;
  Form1.SpeedButton2.Enabled := false;
  Form1.label17.Visible := false;
  Form1.PreMatri.Enabled := false;
  Form1.LogradouroBD.Enabled := false;
  Form1.Panel5.Visible := false;
  Form1.SpeedButton3.Enabled := false;
  Form1.label19.Visible := false;
  Form1.label20.Visible := false;
  Form1.label21.Visible := false;
  if ApagarPontos
  then begin
        tLayer2 := Form1.Mapa_Austin.TrackingLayer;
        tLayer2.ClearEvents;
       end;
end;

procedure TForm1.MenuMapaAustinClick(Sender: TObject);
```

```
begin
  ApagarMapas;
  ColorirAustin (false);
  ExibirCamadasPadrao;
  Mapa_Austin.Visible := true;
  Exibir.Enabled := true;
  Camadas1.Enabled := true;
  label14.Visible := true;
  panel3.Visible := true;
  BotaoZoomSelec.Enabled := true;
  SpeedButton1.Enabled := true;
  BotaoZoomAproximar.Enabled := true;
  BotaoZoomAfastar.Enabled := true;
  BotaoDeslocar.Enabled := true;
  Form1.PreMatri.Enabled := true;
  Form1.LogradouroBD.Enabled := true;
end;

procedure TForm1.MenuNovaIguauClick(Sender: TObject);

begin
  ApagarMapas;
  Map1.Visible := true;
  PanelLegenda.Visible := true;
  label1.Visible := true;
  label2.Visible := true;
end;

procedure TForm1.MenuMapaEscolasClick(Sender: TObject);

var
  Camadas : IMoLayers;
  Camada  : IMoMapLayer;

begin
  ApagarMapas;
  ColorirAustin (false);
  Camadas := Form1.Mapa_Austin.Layers;
  Camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  Camada := IMoMapLayer(Camadas.Item('text_log_reg_austin.dwg'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('austin'));
  Camada.Visible := true;
  Camada := IMoMapLayer(Camadas.Item('lograd_inter'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('area'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('escolas'));
  Camada.Visible := true;
  Camada := IMoMapLayer(Camadas.Item('ferrovia_inter'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('rios_inter'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('buff2'));
  Camada.Visible := false;
  Mapa_Austin.Visible := true;
  Mapa_Austin.refresh;
  label17.Visible := true;
  panel4.Visible := true;
  SpeedButton2.Enabled := true;
end;

procedure TForm1.MenuMapaAlunosEscClick(Sender: TObject);

var
  Camadas : IMoLayers;
  Camada  : IMoMapLayer;

begin
  ApagarMapas;
  Camadas := Form1.Mapa_Austin.Layers;
  Camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  Camada := IMoMapLayer(Camadas.Item('text_log_reg_austin.dwg'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('austin'));
  Camada.Visible := true;
  Camada := IMoMapLayer(Camadas.Item('lograd_inter'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('area'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('escolas'));
  Camada.Visible := true;
  Camada := IMoMapLayer(Camadas.Item('ferrovia_inter'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('rios_inter'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('buff2'));
  Camada.Visible := false;
  ColorirAustin (true);
  Mapa_Austin.Visible := true;
  Mapa_Austin.refresh;
  Form1.Panel5.Visible := true;
  Form1.SpeedButton3.Enabled := true;
  Form1.label19.Visible := true;
  Form1.label20.Visible := true;
  Form1.label21.Visible := true;
  Form1.label20.Caption := 'Escola';
  Form1.label21.Caption := 'Série';
  Exibir.Enabled := true;
  Camadas1.Enabled := false;
  BotaoZoomSelec.Enabled := true;
  SpeedButton1.Enabled := true;
  BotaoZoomAproximar.Enabled := true;
  BotaoZoomAfastar.Enabled := true;
  BotaoDeslocar.Enabled := true;
end;
```

```
procedure TForm1.FormClick(Sender: TObject);

begin
  DescritorEscola.Visible := false;
  if Form1Iniciado
  then begin
        tLayer := Mapa_Austin.TrackingLayer;
        tLayer.ClearEvents;
       end;
end;

procedure TForm1.SpeedButton3Click(Sender: TObject);

begin
  SelEscSerie.SelecEscolaSerie.ComboPalavraChave.Clear;
  SelEscSerie.SelecEscolaSerie.Visible := true;
  Form1.Enabled := false;
end;

begin
  Form1Iniciado := false;
  ApagarPontos := false;
end.
```

A.1.9 RelAlunosNaoMatri

```
unit RelAlunosNaoMatri;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Interface_Usuario;

type
  TRelatorioAlunosNaoMatriculados = class(TForm)
    Nnome: TLabel;
    Nde: TLabel;
    Nate: TLabel;
    EditDe: TEdit;
    EditAte: TEdit;
    Botao_Mostrar: TButton;
    Botao_Cancelar: TButton;
    procedure Botao_CancelarClick(Sender: TObject);
    procedure Botao_MostrarClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  RelatorioAlunosNaoMatriculados: TRelatorioAlunosNaoMatriculados;

implementation

{$R *.DFM}

procedure TRelatorioAlunosNaoMatriculados.Botao_CancelarClick(
  Sender: TObject);

begin
  RelatorioAlunosNaoMatriculados.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TRelatorioAlunosNaoMatriculados.Botao_MostrarClick(
  Sender: TObject);

begin
  UnitRelAluNaoMatri.QRMDFormAluNaoMatri.ADOQuery1.Active := false;
  UnitRelAluNaoMatri.QRMDFormAluNaoMatri.ADOQuery1.Parameters.ParamValues['DE']
    := EditDe.Text;
  UnitRelAluNaoMatri.QRMDFormAluNaoMatri.ADOQuery1.Parameters.ParamValues['ATE']
    := EditAte.Text;
  if EditDe.Text = ''
  then UnitRelAluNaoMatri.QRMDFormAluNaoMatri.ADOQuery1.Parameters.ParamValues['DE']
         := chr (1);
  if EditAte.Text = ''
  then UnitRelAluNaoMatri.QRMDFormAluNaoMatri.ADOQuery1.Parameters.ParamValues['ATE']
         := chr (255);
  UnitRelAluNaoMatri.QRMDFormAluNaoMatri.ADOQuery1.Active := true;
  QRMDFormAluNaoMatri.QuickRep1.Preview;
end;

end.
```

A.1.10 *Unit* RelBD

```
unit RelBD;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, UnitRelBD1, UnitRelBD2, {UnitRelBD3,} Interface_Usuario;

type
  TRelatoriosBD = class(TForm)
    Nnome: TLabel;
```

```
    Nde: TLabel;
    Nate: TLabel;
    RadioGroupRelat: TRadioGroup;
    EditDe: TEdit;
    EditAte: TEdit;
    Botao_Mostrar: TButton;
    Botao_Cancelar: TButton;
    procedure Botao_CancelarClick(Sender: TObject);
    procedure Botao_MostrarClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  RelatoriosBD: TRelatoriosBD;

implementation

{$R *.DFM}

procedure TRelatoriosBD.Botao_CancelarClick(Sender: TObject);

begin
  RelatoriosBD.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TRelatoriosBD.Botao_MostrarClick(Sender: TObject);

begin
  case RadioGroupRelat.ItemIndex of
    0 : begin
         UnitRelBD1.QRMDFormBD1.ADOQuery1.Active := false;
         UnitRelBD1.QRMDFormBD1.ADOQuery1.Parameters.ParamValues['DE']
           := EditDe.Text;
         UnitRelBD1.QRMDFormBD1.ADOQuery1.Parameters.ParamValues['ATE']
           := EditAte.Text;
         if EditDe.Text = ''
         then UnitRelBD1.QRMDFormBD1.ADOQuery1.Parameters.ParamValues['DE']
                := chr (1);
         if EditAte.Text = ''
         then UnitRelBD1.QRMDFormBD1.ADOQuery1.Parameters.ParamValues['ATE']
                := chr (255);
         UnitRelBD1.QRMDFormBD1.ADOQuery1.Active := true;
         UnitRelBD1.QRMDFormBD1.QuickRep1.Preview;
        end;
    1 : begin
         UnitRelBD2.QRMDFormBD2.ADOQuery1.Active := false;
         UnitRelBD2.QRMDFormBD2.ADOQuery1.Parameters.ParamValues['DE']
           := EditDe.Text;
         UnitRelBD2.QRMDFormBD2.ADOQuery1.Parameters.ParamValues['ATE']
           := EditAte.Text;
         if EditDe.Text = ''
         then UnitRelBD2.QRMDFormBD2.ADOQuery1.Parameters.ParamValues['DE']
                := chr (1);
         if EditAte.Text = ''
         then UnitRelBD2.QRMDFormBD2.ADOQuery1.Parameters.ParamValues['ATE']
                := chr (255);
         UnitRelBD2.QRMDFormBD2.ADOQuery1.Active := true;
         UnitRelBD2.QRMDFormBD2.QuickRep1.Preview;
        end;
   end;
end;

end.
```

A.1.11 *Unit* RelPreMatri

```
unit RelPreMatri;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, Interface_Usuario, UnitRelCadAluno1,
  UnitRelCadAluno2, UnitRelCadAluno3, CadMatri;

type
  TRelatoriosPreMatricula = class(TForm)
    RadioGroupRelat: TRadioGroup;
    EditDe: TEdit;
    EditAte: TEdit;
    Nnome: TLabel;
    Nde: TLabel;
    Nate: TLabel;
    Botao_Mostrar: TButton;
    Botao_Cancelar: TButton;
    procedure Botao_CancelarClick(Sender: TObject);
    procedure Botao_MostrarClick(Sender: TObject);
    procedure RadioGroupRelatClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  RelatoriosPreMatricula: TRelatoriosPreMatricula;

implementation

{$R *.DFM}
```

```
procedure TRelatoriosPreMatricula.Botao_CancelarClick(Sender: TObject);

begin
  RelatoriosPreMatricula.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TRelatoriosPreMatricula.Botao_MostrarClick(Sender: TObject);

begin
  case RadioGroupRelat.ItemIndex of
    0 : begin
         UnitRelCadAluno1.QRMDFormCadAluno1.ADOQuery1.Active := false;
         UnitRelCadAluno1.QRMDFormCadAluno1.ADOQuery1.Parameters.ParamValues['DE']
           := EditDe.Text;
         UnitRelCadAluno1.QRMDFormCadAluno1.ADOQuery1.Parameters.ParamValues['ATE']
           := EditAte.Text;
         if EditDe.Text = ''
         then UnitRelCadAluno1.QRMDFormCadAluno1.ADOQuery1.Parameters.ParamValues['DE']
                := chr (1);
         if EditAte.Text = ''
         then UnitRelCadAluno1.QRMDFormCadAluno1.ADOQuery1.Parameters.ParamValues['ATE']
                := chr (255);
         UnitRelCadAluno1.QRMDFormCadAluno1.ADOQuery1.Active := true;
         QRMDFormCadAluno1.QuickRep1.Preview;
        end;
    1 : QRMDFormCadAluno2.QuickRep1.Preview;
    2 : QRMDFormCadAluno3.QuickRep1.Preview;
  end;
end;

procedure TRelatoriosPreMatricula.RadioGroupRelatClick(Sender: TObject);

begin
  case RadioGroupRelat.ItemIndex of
    0 : begin
          Nnome.Enabled := true;
          Nde.Enabled := true;
          Nate.Enabled := true;
          EditDe.Enabled := true;
          EditAte.Enabled := true;
        end;
    1,2 : begin
          Nnome.Enabled := false;
          Nde.Enabled := false;
          Nate.Enabled := false;
          EditDe.Enabled := false;
          EditAte.Enabled := false;
        end;
  end;
end;

end.
```

A.1.12 *Unit* RelTurProc

```
unit RelTurProc;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, Interface_Usuario, UnitRelTurProc1, UnitRelTurProc2,
  UnitRelTurProc3;

type
  TRelatoriosTurmasProc = class(TForm)
    Nnome: TLabel;
    Nde: TLabel;
    Nate: TLabel;
    RadioGroupRelat: TRadioGroup;
    EditDe: TEdit;
    EditAte: TEdit;
    Botao_Mostrar: TButton;
    Botao_Cancelar: TButton;
    NnomeAluno: TLabel;
    NdeAluno: TLabel;
    NateAluno: TLabel;
    EditDeAluno: TEdit;
    EditAteAluno: TEdit;
    procedure Botao_CancelarClick(Sender: TObject);
    procedure Botao_MostrarClick(Sender: TObject);
    procedure RadioGroupRelatClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  RelatoriosTurmasProc: TRelatoriosTurmasProc;

implementation

{$R *.DFM}

procedure TRelatoriosTurmasProc.Botao_CancelarClick(Sender: TObject);

begin
  RelatoriosTurmasProc.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
```

```
end;

procedure TRelatoriosTurmasProc.Botao_MostrarClick(Sender: TObject);

begin
  case RadioGroupRelat.ItemIndex of
    0 : begin
         UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Active := false;
         UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['DE']
           := EditDe.Text;
         UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['ATE']
           := EditAte.Text;
         if EditDe.Text = ''
         then UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['DE']
                := chr (1);
         if EditAte.Text = ''
         then UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['ATE']
                := chr (255);
         UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['DEALUNO']
           := EditDeAluno.Text;
         UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['ATEALUNO']
           := EditAteAluno.Text;
         if EditDeAluno.Text = ''
         then UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['DEALUNO']
                := chr (1);
         if EditAteAluno.Text = ''
         then UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Parameters.ParamValues['ATEALUNO']
                := chr (255);
         UnitRelTurProc1.QRMDFormTurProc1.ADOQuery1.Active := true;
         UnitRelTurProc1.QRMDFormTurProc1.QuickRep1.Preview;
        end;
    1 : begin
         UnitRelTurProc2.QRMDFormTurProc2.ADOQuery1.Active := false;
         UnitRelTurProc2.QRMDFormTurProc2.ADOQuery1.Parameters.ParamValues['DE']
           := EditDe.Text;
         UnitRelTurProc2.QRMDFormTurProc2.ADOQuery1.Parameters.ParamValues['ATE']
           := EditAte.Text;
         if EditDe.Text = ''
         then UnitRelTurProc2.QRMDFormTurProc2.ADOQuery1.Parameters.ParamValues['DE']
                := chr (1);
         if EditAte.Text = ''
         then UnitRelTurProc2.QRMDFormTurProc2.ADOQuery1.Parameters.ParamValues['ATE']
                := chr (255);
         UnitRelTurProc2.QRMDFormTurProc2.ADOQuery1.Active := true;
         UnitRelTurProc2.QRMDFormTurProc2.QuickRep1.Preview;
        end;
{    2 : begin
         UnitRelTurProc3.QRMDFormTurProc3.ADOQuery1.Active := false;
         UnitRelTurProc3.QRMDFormTurProc3.ADOQuery1.Parameters.ParamValues['DE']
           := EditDe.Text;
         UnitRelTurProc3.QRMDFormTurProc3.ADOQuery1.Parameters.ParamValues['ATE']
           := EditAte.Text;
         if EditDe.Text = ''
         then UnitRelTurProc3.QRMDFormTurProc3.ADOQuery1.Parameters.ParamValues['DE']
                := chr (1);
         if EditAte.Text = ''
         then UnitRelTurProc3.QRMDFormTurProc3.ADOQuery1.Parameters.ParamValues['ATE']
                := chr (255);
         UnitRelTurProc3.QRMDFormTurProc3.ADOQuery1.Active := true;
         UnitRelTurProc3.QRMDFormTurProc3.QuickRep1.Preview;
        end;
 }  end;
end;

procedure TRelatoriosTurmasProc.RadioGroupRelatClick(Sender: TObject);

begin
  case RadioGroupRelat.ItemIndex of
    0 : begin
          NnomeAluno.Enabled := true;
          NdeAluno.Enabled := true;
          NateAluno.Enabled := true;
          EditDeAluno.Enabled := true;
          EditAteAluno.Enabled := true;
        end;
    1,2 : begin
          NnomeAluno.Enabled := false;
          NdeAluno.Enabled := false;
          NateAluno.Enabled := false;
          EditDeAluno.Enabled := false;
          EditAteAluno.Enabled := false;
        end;
  end;
end;

end.
```

A.1.13 *Unit* SelCad

```
unit SelCad;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, MapObjects2_TLB, ComObj, CadMatri, Interface_Usuario;

type
  TSelecCad = class(TForm)
    digitar_aluno: TLabel;
    Edit1: TEdit;
    ListBox1: TListBox;
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
    procedure EscolheuAlunoProc(Sender: TObject);
    procedure EscolheuAlunoComKeyPress(Sender: TObject; var Key: Char);
```

```
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  SelecCad          : TSelecCad;
  ExcluindoPreMatri : boolean;

implementation

{$R *.DFM}

procedure TSelecCad.Edit1KeyPress(Sender: TObject; var Key: Char);

var
  nstr : string[60];

begin
  if ord (Key) = 13
  then begin
        CadastrarMatricula.TabelaMatri.TableName := 'Aluno';
        CadastrarMatricula.TabelaMatri.first;
        SelecCad.Edit1.Text := UpperCase (SelecCad.Edit1.Text);
        ListBox1.Items.clear;
        while not CadastrarMatricula.TabelaMatri.Eof do
         begin
          nstr := CadastrarMatricula.TabelaMatri.FieldValues['NOME'];
          nstr := UpperCase (nstr);
          if pos (SelecCad.Edit1.Text,nstr) <> 0
          then ListBox1.Items.Add
               (CadastrarMatricula.TabelaMatri.FieldValues['NOME']);
          CadastrarMatricula.TabelaMatri.next;
         end;
        CadastrarMatricula.TabelaMatri.first;
       end;
end;

procedure EscolhendoAlunoPARAexcluir;

var
  i,selecionado : word;
  achei         : boolean;
  msgconfir     : string[255];

begin
  CadastrarMatricula.TabelaMatri.first;
  for i := 0 to SelecCad.ListBox1.Items.count - 1 do
     if SelecCad.ListBox1.selected[i] then selecionado := i;
  achei := false;
  while not (CadastrarMatricula.TabelaMatri.Eof) and not achei do
      if CadastrarMatricula.TabelaMatri.FieldValues['NOME'] =
         SelecCad.ListBox1.Items.strings[selecionado]
      then achei := true
      else CadastrarMatricula.TabelaMatri.next;
  msgconfir := 'Confirmar a exclusão do seguinte cadastro:'
            + #13#10#13#10;
  msgconfir := msgconfir + 'Nome do Aluno: ';
  msgconfir := msgconfir +
               CadastrarMatricula.TabelaMatri.FieldValues['NOME']
               + #13#10;
  msgconfir := msgconfir + 'Nome do Pai: ';
  msgconfir := msgconfir +
               CadastrarMatricula.TabelaMatri.FieldValues['PAI']
               + #13#10;
  msgconfir := msgconfir + 'Nome da Mãe: ';
  msgconfir := msgconfir +
               CadastrarMatricula.TabelaMatri.FieldValues['MAE']
               + #13#10;
  msgconfir := msgconfir + 'Data de Nascimento: ';
  msgconfir := msgconfir + DateTimeToStr
               (CadastrarMatricula.TabelaMatri.FieldValues['NASCIMENTO'])
               + #13#10;
  if MessageDlg (msgconfir,mtConfirmation,mbOKCancel,0) = mrOK
  then CadastrarMatricula.TabelaMatri.Delete;
  {Form1.Enabled := false;}
  SelecCad.ListBox1.Items.clear;
  SelecCad.Edit1.Clear;
  CadastrarMatricula.TabelaMatri.first;

end;

procedure EscolhendoAlunoPARAatualizar;

var
  i,selecionado : word;
  achei         : boolean;

begin
  SelecCad.Visible := false;
  CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biMaximize];
  CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biHelp];
  CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biMinimize];
  CadastrarMatricula.BorderIcons := CadastrarMatricula.BorderIcons - [biSystemMenu];
  CadastrarMatricula.Botao_Cancelar.Caption := 'Fechar';
  CadastrarMatricula.Botao_cadastrar.Visible := false;
  CadastrarMatricula.Botao_Localizar.Visible := false;
  CadastrarMatricula.Botao_Aplicar.Visible := true;
  CadastrarMatricula.Botao_AtualizarLocalizacao.Enabled := false;
  CadastrarMatricula.Botao_AtualizarLocalizacao.Visible := true;
  for i := 0 to SelecCad.ListBox1.Items.count - 1 do
     if SelecCad.ListBox1.selected[i] then selecionado := i;
  achei := false;
  while not (CadastrarMatricula.TabelaMatri.Eof) and not achei do
      if CadastrarMatricula.TabelaMatri.FieldValues['NOME'] =
         SelecCad.ListBox1.Items.strings[selecionado]
```

```
      then achei := true
      else CadastrarMatricula.TabelaMatri.next;
  CadastrarMatricula.Edit_Nome.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['NOME'];
  CadastrarMatricula.Edit_email.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['EMAIL'];
  CadastrarMatricula.Edit_Pai.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['PAI'];
  CadastrarMatricula.Edit_Mae.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['MAE'];
  CadastrarMatricula.EditData.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['NASCIMENTO'];
  CadastrarMatricula.Edit_Rua.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['RUA'];
  CadastrarMatricula.Edit_Numero.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['NUM'];
  CadastrarMatricula.Edit_Comp.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['COMPL'];
  CadastrarMatricula.Edit_Bairro.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['BAIRRO'];
  CadastrarMatricula.Edit_Cidade.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['CIDADE'];
  CadastrarMatricula.Combo_Estado.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['ESTADO'];
  CadastrarMatricula.Combo_Serie.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['SERIE'];
  CadastrarMatricula.Edit_CEP.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['CEP'];
  CadastrarMatricula.Edit_TEL1.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['TEL1'];
  CadastrarMatricula.Edit_TEL2.Text :=
     CadastrarMatricula.TabelaMatri.FieldValues['TEL2'];
  CadastrarMatricula.Caption := 'Atualizar Pré-Matrícula';
  CadastrarMatricula.Visible := true;
  Form1.Enabled := false;
  SelecCad.ListBox1.Items.clear;
  SelecCad.Edit1.Clear;
end;

procedure TSelecCad.EscolheuAlunoProc(Sender: TObject);

begin
  if not ExcluindoPreMatri then EscolhendoAlunoPARAatualizar;
  if ExcluindoPreMatri then EscolhendoAlunoPARAexcluir;
end;

procedure TSelecCad.EscolheuAlunoComKeyPress(Sender: TObject;
  var Key: Char);

begin
  if (ord (Key) = 13) and not ExcluindoPreMatri
  then EscolhendoAlunoPARAatualizar;
  if (ord (Key) = 13) and ExcluindoPreMatri
  then EscolhendoAlunoPARAexcluir;
end;

end.
```

A.1.14 *Unit* SelCamadas

```
unit SelCamadas;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  OleCtrls, StdCtrls, MapObjects2_TLB, ComObj, Interface_Usuario, ActiveX,
  ComCtrls, ToolWin;

type
  TSelecCamadas = class(TForm)
    AjudaSelecCamadas: TLabel;
    Bairros: TCheckBox;
    Logradouros: TCheckBox;
    Area: TCheckBox;
    Escolas: TCheckBox;
    ExibirCamadas: TButton;
    ExibirCamadasDefault: TButton;
    CancelarExibCamadas: TButton;
    Ferrovias: TCheckBox;
    Rios: TCheckBox;
    Texto: TCheckBox;
    procedure CancelarSelCamadasProc(Sender: TObject);
    procedure ExibirCamadasProc(Sender: TObject);
    procedure ExibirCamadasDefaultProc(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

procedure ExibirCamadasPadrao;

var
  SelecCamadas: TSelecCamadas;

implementation

{$R *.DFM}

procedure TSelecCamadas.CancelarSelCamadasProc(Sender: TObject);

begin
  SelecCamadas.Visible := false;
```

```
end;

procedure TSelecCamadas.ExibirCamadasProc(Sender: TObject);

var
  Camadas : IMoLayers;
  Camada  : IMoMapLayer;
  ly      : IMoMapLayer;
  lblren  : IMoLabelPlacer;
  lyrs    : Layers;
  ft      : TFont;
  oleFt   : variant;


begin
  Camadas := Interface_Usuario.Form1.Mapa_Austin.Layers;
  Camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  Camada := IMoMapLayer(Camadas.Item('text_log_reg_austin.dwg'));
  if Texto.State = cbChecked
  then Camada.Visible := true
  else Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('austin'));
  if Bairros.State = cbChecked
  then Camada.Visible := true
  else Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('lograd_inter'));
  if Logradouros.State = cbChecked
  then Camada.Visible := true
  else Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('area'));
  if Area.State = cbChecked
  then begin
        Camada.Visible := true;
        ft := TFont.Create;
        ft.name := 'MS Sans Serif';
        ft.size := 35;
        oleFt := FontToOleFont(ft);
        lyrs := IMoLayers(Form1.Mapa_Austin.layers);
        ly := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
        lblren := IMoLabelPlacer(CreateOleObject('MapObjects2.LabelPlacer'));
        lblren.defaultsymbol.height := Form1.Mapa_Austin.extent.height/25;
        lblren.field :='ID';
        lblren.defaultsymbol.font := IFontDisp(IDispatch(oleFt));
        lblren.AllowDuplicates :=false;
        ly := IMoMapLayer(lyrs.item('area'));
        ly.renderer := lblren;
        ft.Free;
       end
  else Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('escolas'));
  if Escolas.State = cbChecked
  then Camada.Visible := true
  else Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('ferrovia_inter'));
  if Ferrovias.State = cbChecked
  then Camada.Visible := true
  else Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('rios_inter'));
  if Rios.State = cbChecked
  then Camada.Visible := true
  else Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('austin_mc'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('buff2'));
  Camada.Visible := false;
  SelecCamadas.Visible := false;
  Interface_Usuario.Form1.Mapa_Austin.refresh;
end;

procedure ExibirCamadasPadrao;

var
  Camadas : IMoLayers;
  Camada  : IMoMapLayer;

begin
  Camadas := Interface_Usuario.Form1.Mapa_Austin.Layers;
  Camada := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  Camada := IMoMapLayer(Camadas.Item('text_log_reg_austin.dwg'));
  SelecCamadas.Texto.State := cbUnChecked;
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('austin'));
  SelecCamadas.Bairros.State := cbChecked;
  Camada.Visible := true;
  Camada := IMoMapLayer(Camadas.Item('lograd_inter'));
  SelecCamadas.Logradouros.State := cbChecked;
  Camada.Visible := true;
  Camada := IMoMapLayer(Camadas.Item('area'));
  SelecCamadas.Area.State := cbUnChecked;
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('escolas'));
  SelecCamadas.Escolas.State := cbUnChecked;
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('ferrovia_inter'));
  SelecCamadas.Ferrovias.State := cbUnChecked;
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('rios_inter'));
  SelecCamadas.Rios.State := cbUnChecked;
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('austin_mc'));
  Camada.Visible := false;
  Camada := IMoMapLayer(Camadas.Item('buff2'));
  Camada.Visible := false;
  Interface_Usuario.Form1.Mapa_Austin.refresh;
  SelecCamadas.Visible := false;
end;
```

```
procedure TSelecCamadas.ExibirCamadasDefaultProc(Sender: TObject);

begin
  ExibirCamadasPadrao;
end;

end.
```

A.1.15 *Unit* SelEscSerie

```
unit SelEscSerie;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, Mask, DBCtrls, Interface_Usuario, ConsultEsc,
  ConsultAlun, CadMatri, MapObjects2_TLB, ComObj, OleCtrls;

type
  TSelecEscolaSerie = class(TForm)
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    BotaoPesquisar: TButton;
    ComboPalavraChave: TComboBox;
    Nescola: TLabel;
    DBEditNome: TDBEdit;
    RadioGroup1: TRadioGroup;
    BotaoAceitarSelec: TButton;
    BotaoCancelar: TButton;
    procedure BotaoCancelarClick(Sender: TObject);
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure ComboPalavraChaveClick(Sender: TObject);
    procedure BotaoAceitarSelecClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  SelecEscolaSerie: TSelecEscolaSerie;

implementation

{$R *.DFM}

procedure TSelecEscolaSerie.BotaoCancelarClick(Sender: TObject);

begin
  SelecEscolaSerie.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TSelecEscolaSerie.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   ConsultaEscola.TabEscola.first;
   ListaDEnomes := TStringList.Create;
   while not ConsultaEscola.TabEscola.Eof do
      begin
        straux := ConsultaEscola.TabEscola.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (ConsultaEscola.TabEscola.FieldValues['NOME']);
        ConsultaEscola.TabEscola.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure TSelecEscolaSerie.ComboPalavraChaveClick(Sender: TObject);

var
  achei : boolean;

begin
   achei := false;
   ConsultaEscola.TabEscola.first;
   while not (ConsultaEscola.TabEscola.Eof)
         and not achei do
      begin
        if ConsultaEscola.TabEscola.FieldValues['NOME'] =
           ComboPalavraChave.Text
        then achei := true
        else ConsultaEscola.TabEscola.next;
      end;
end;

procedure TSelecEscolaSerie.BotaoAceitarSelecClick(Sender: TObject);

var
  ponto   : Point;
  simbolo : IMoSymbol;
  l       : IMoMapLayer;
  recs    : IMoRecordset;
  flds    : IMoFields;
  lys     : IMoLayers;
  exp     : string;
```

```
  id_str  : string[5];
  id      : cardinal;

begin
  Form1.Label20.Caption := DBEditNome.Text;
  Form1.Label21.Caption := RadioGroup1.Items[RadioGroup1.ItemIndex];
  tLayer2 := Form1.Mapa_Austin.TrackingLayer;
  tLayer2.ClearEvents;
  tLayer2.SymbolCount := 2;
  lys := Form1.Mapa_Austin.Layers;
  exp :='Escola =''' + ConsultaEscola.TabEscola.FieldValues['COD_ESC'] + '''';
  l := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  l := IMoMapLayer(lys.Item('Buff2'));
  recs := l.SearchExpression(exp);
  if not recs.eof
  then begin
         flds := recs.Fields;
         sym := IMoSymbol(CreateComObject(Class_Symbol));
         sym.Color := clBlack;
         sym.Size := 2;
         sym.Style := 1;
         shp := IMoPolygon(CreateOleObject('MapObjects2.Polygon'));
         shp := IMoPolygon(IDispatch(flds.Item('Shape').Value));
         SelecioneiEscola := true;
       end;
  id := ConsultaEscola.TabEscola.FieldValues['ID'];
  str (id,id_str);
  exp :='Id =' + id_str;
  lys := Form1.Mapa_Austin.Layers;
  l := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  l := IMoMapLayer(lys.Item('escolas'));
  l.Visible := false;
  recs := l.SearchExpression(exp);
  if not recs.eof
  then begin
         flds := recs.Fields;
         simbolo := IMoSymbol(CreateComObject(Class_Symbol));
         ponto := IMoPoint(CreateOleObject('MapObjects2.Point'));
         ponto := IMoPoint(IDispatch(flds.Item('Shape').Value));
         simbolo := tLayer2.Symbol[0];
         simbolo.Color := moOrange;
         simbolo.Size := 10;
         tLayer2.AddEvent(ponto,0);
       end;

//----------- plotar alunos ---------
  simbolo := IMoSymbol(CreateComObject(Class_Symbol));
  ponto := IMoPoint(CreateOleObject('MapObjects2.Point'));
  simbolo := tLayer2.Symbol[1];
  simbolo.Color := moBlue;
  simbolo.Size := 4;
  simbolo.CharacterIndex := 1;
  CadMatri.CadastrarMatricula.TabelaMatri.First;
  while not CadMatri.CadastrarMatricula.TabelaMatri.Eof do
   begin
     if (Radiogroup1.ItemIndex = 11) or
        (RadioGroup1.Items[RadioGroup1.ItemIndex] =
         CadMatri.CadastrarMatricula.TabelaMatri.FieldValues['SERIE'])
     then begin
           if not Form1.TabTurma.Locate
               ('COD_TURM',
                 CadastrarMatricula.TabelaMatri.FieldValues['COD_TURM'],[])
           then begin
                 showmessage ('Erro no banco de dados TURMA');
                 halt;
                end;
           if Form1.TabTurma.FieldValues['COD_ESC'] =
              ConsultaEscola.TabEscola.FieldValues['COD_ESC']
           then begin
                 ponto.X := CadastrarMatricula.TabelaMatri.FieldValues['COORD_X'];
                 ponto.Y := CadastrarMatricula.TabelaMatri.FieldValues['COORD_Y'];
                 tLayer2.AddEvent(ponto,1);
                end;
         end;
     CadMatri.CadastrarMatricula.TabelaMatri.Next;
   end;
//----------- fim plotar alunos -----

  Form1.Mapa_Austin.Refresh;
  SelecEscolaSerie.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
  ApagarPontos := true;
end;

end.
```

A.1.16 *Unit* Selog

```
unit Selog;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, StdCtrls, MapObjects2_TLB, ComObj, Interface_Usuario;

type
  TSelecLog = class(TForm)
    ADOTable1: TADOTable;
    Edit1: TEdit;
    ListBox1: TListBox;
    digitar_log: TLabel;
    procedure Edit1KeyPress(Sender: TObject; var Key: Char);
```

```
    procedure EscolheuLogProc(Sender: TObject);
    procedure EscolheuLogComKeyPress(Sender: TObject; var Key: Char);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  SelecLog     : TSelecLog;

implementation

{$R *.DFM}

procedure zoom_quadra (quadra : integer);

var
  linha, coluna                      : byte;
  ret                                : IMoRectangle;
  largura_da_coluna, altura_da_linha : double;

begin
  largura_da_coluna :=
    (Form1.Mapa_Austin.FullExtent.Right - Form1.Mapa_Austin.FullExtent.Left)/8.0;
  altura_da_linha :=
    (Form1.Mapa_Austin.FullExtent.Top - Form1.Mapa_Austin.FullExtent.Bottom)/10.0;
  linha := 1;
  while Qt_quadras_por_linha[linha] < quadra do
    begin
      quadra := quadra - Qt_quadras_por_linha[linha];
      inc (linha);
    end;
  case linha of
     9 : coluna := quadra + 1;
    10 : coluna := quadra + 3;
  else  coluna := quadra;
  end;
  ret := Form1.Mapa_Austin.FullExtent;
  ret.left := ret.Left + (coluna - 1)*largura_da_coluna;
  ret.right := ret.left + largura_da_coluna;
  ret.bottom := ret.bottom + (linha - 1)*altura_da_linha;
  ret.top := ret.bottom + altura_da_linha;
  Form1.Mapa_Austin.Extent := ret;
end;

procedure TSelecLog.Edit1KeyPress(Sender: TObject; var Key: Char);

var
  nstr : string[60];

begin
  if ord (Key) = 13
  then begin
        SelecLog.Edit1.Text := UpperCase (SelecLog.Edit1.Text);
        ListBox1.Items.clear;
        while not ADOTable1.Eof do
         begin
          nstr := ADOTable1.FieldValues['Logradouro'];
          if pos (SelecLog.Edit1.Text,nstr) <> 0
          then ListBox1.Items.Add (nstr);
          ADOTable1.next;
         end;
        ADOTable1.first;
       end;
end;

procedure EscolhendoLog;

var
  i,selecionado : word;
  achei         : boolean;
  unidade       : string [15];
  quadra,codigo : integer;

begin
  for i := 0 to SelecLog.ListBox1.Items.count -1 do
     if SelecLog.ListBox1.selected[i] then selecionado := i;
  achei := false;
  while not (SelecLog.ADOTable1.Eof) and not achei do
      if SelecLog.ADOTable1.FieldValues['LOGRADOURO'] =
         SelecLog.ListBox1.Items.strings[selecionado]
      then achei := true
      else SelecLog.ADOTable1.next;
  if achei
  then begin
         unidade := SelecLog.ADOTable1.FieldValues['UNIDADE'];
         delete (unidade,1,7);
         val (unidade,quadra,codigo);
         zoom_quadra (quadra);
         SelecLog.ADOTable1.first;
         SelecLog.ListBox1.Items.clear;
         SelecLog.Edit1.Clear;
         SelecLog.Visible := false;
       end
  else showmessage ('Erro: Logradouro não encontrado');
end;

procedure TSelecLog.EscolheuLogProc(Sender: TObject);

begin
  EscolhendoLog;
end;

procedure TSelecLog.EscolheuLogComKeyPress(Sender: TObject; var Key: Char);

begin
```

```
  if ord (Key) = 13 then EscolhendoLog;
end;

end.
```

A.1.17 *Unit* UnitDescEsc

```
unit UnitDescEsc;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Grids, Interface_Usuario, MapObjects2_TLB, ComObj;

type
  TDescritorEscola = class(TForm)
    StringGrid1: TStringGrid;
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

procedure MostraInfoEscola (X, Y : integer);

var
  DescritorEscola: TDescritorEscola;

implementation

{$R *.DFM}


procedure MostraInfoEscola (X, Y : integer);

var
  l         : IMoMapLayer;
  p         : IMoPoint;
  shp_Point : IMoPoint;
  recs      : IMoRecordset;
  fields    : IMoFields;
  fld       : IMoField;
  tdesc     : IMoTableDesc;
  i         : integer;

begin
  l := IMoMapLayer(CreateOleObject('MapObjects2.MapLayer'));
  l := IMoMapLayer(Form1.Mapa_Austin.Layers.item('escolas'));
  p := IMoPoint(CreateOleObject('MapObjects2.Point'));
  p := Form1.Mapa_Austin.ToMapPoint(x,y);
  tLayer := Form1.Mapa_Austin.TrackingLayer;
  tLayer.ClearEvents;
  if l.ShapeType=23
  then recs :=l.SearchShape(p,12,'')
  else recs :=l.SearchByDistance(p,Form1.Mapa_Austin.ToMapDistance(10),'');
      // if the search returned something, diaplay the fields and values
  if not recs.eof
  then begin
        DescritorEscola.StringGrid1.DefaultDrawing := true;
        fields := recs.Fields;
        tdesc := recs.TableDesc ;
       //fld := IMoField(CreateOleObject('MapObjects2.Field'));
        for i := 0 to tdesc.FieldCount - 1  do
         begin
          DescritorEscola.StringGrid1.cells[0,0] :=' Campo';
          DescritorEscola.StringGrid1.cells[1,0] :=' Valor';
          fld := fields.Item(tdesc.FieldName[i]);
          DescritorEscola.StringGrid1.cells[0,i+1] := fld.name;
          DescritorEscola.StringGrid1.Cells[1,i+1] :=fld.valueasstring;
         end;
        sym := IMoSymbol(CreateComObject(Class_Symbol));
        //sym.OutlineColor := clGray;
        shp_Point := IMoPoint(CreateOleObject('MapObjects2.Point'));
        shp_Point := IMoPoint(IDispatch(fields.Item('Shape').Value));
        sym := tLayer.Symbol[0];
        sym.Color := clGray;
        sym.Size := 8;
        tLayer.AddEvent(shp_Point,0);
       end;
end;

procedure TDescritorEscola.FormClose(Sender: TObject;
  var Action: TCloseAction);

begin
  Form1.SetFocus;
  tLayer := Form1.Mapa_Austin.TrackingLayer;
  tLayer.ClearEvents;
end;

end.
```

A.1.18 *Unit* UnitEscolaBD

```
unit UnitEscolaBD;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
```

```
  Interface_Usuario, Grids, DBGrids, Db, ADODB, StdCtrls, DBCtrls, Mask,
  ExtCtrls, ConsultEsc, ConsultAlun;

type
  TFormEscolaBD = class(TForm)
    Nescola: TLabel;
    Ncorrente: TLabel;
    Naval: TLabel;
    Nconst: TLabel;
    Ndiretor: TLabel;
    Nsalas: TLabel;
    Ncart: TLabel;
    Ncodigo: TLabel;
    NcoordX: TLabel;
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    BotaoIncluir: TButton;
    ComboPalavraChave: TComboBox;
    PainelEndereco: TPanel;
    Nendereco: TLabel;
    NCEP: TLabel;
    Nrua: TLabel;
    Ncomplemento: TLabel;
    Ncidade: TLabel;
    Nbairro: TLabel;
    Nestado: TLabel;
    Nnumero: TLabel;
    DBEditNumero: TDBEdit;
    DBEditCep: TDBEdit;
    DBEditRua: TDBEdit;
    DBEditComp: TDBEdit;
    DBEditBairro: TDBEdit;
    DBEditCidade: TDBEdit;
    DBEditEstado: TDBEdit;
    DBNavegador: TDBNavigator;
    DBEditDiretor: TDBEdit;
    DBEditCorrente: TDBEdit;
    DBEditAval: TDBEdit;
    DBEditAnoConstru: TDBEdit;
    DBEditNome: TDBEdit;
    DBEditNsalas: TDBEdit;
    DBEditNcart: TDBEdit;
    DBEditCodigo: TDBEdit;
    DBEnergia: TDBCheckBox;
    DBagua: TDBCheckBox;
    DBesgoto: TDBCheckBox;
    DBeduc_infantil: TDBCheckBox;
    DBensino_fund: TDBCheckBox;
    DBensino_ja: TDBCheckBox;
    DBEditCoordX: TDBEdit;
    DataSourceEscola: TDataSource;
    DBEditCoordY: TDBEdit;
    NcoordY: TLabel;
    Label2: TLabel;
    DBEditTel: TDBEdit;
    BotaoPesquisar: TButton;
    BotaoAtualizar: TButton;
    BotaoAplicar: TButton;
    BotaoExcluir: TButton;
    BotaoFechar: TButton;
    procedure ComboPalavraChaveChange(Sender: TObject);
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure BotaoFecharClick(Sender: TObject);
    procedure BotaoAtualizarClick(Sender: TObject);
    procedure BotaoAplicarClick(Sender: TObject);
    procedure BotaoIncluirClick(Sender: TObject);
    procedure BotaoExcluirClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormEscolaBD   : TFormEscolaBD;
  EstouIncluindo : boolean;

implementation

{$R *.DFM}

procedure TFormEscolaBD.ComboPalavraChaveChange(Sender: TObject);

var
  achei : boolean;

begin
   achei := false;
   ConsultaEscola.TabEscola.first;
   while not (ConsultaEscola.TabEscola.Eof)
         and not achei do
      begin
        if ConsultaEscola.TabEscola.FieldValues['NOME'] =
           ComboPalavraChave.Text
        then achei := true
        else ConsultaEscola.TabEscola.next;
      end;
end;

procedure TFormEscolaBD.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   ConsultaEscola.TabEscola.first;
```

```
   ListaDEnomes := TStringList.Create;
   while not ConsultaEscola.TabEscola.Eof do
      begin
        straux := ConsultaEscola.TabEscola.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (ConsultaEscola.TabEscola.FieldValues['NOME']);
        ConsultaEscola.TabEscola.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure TFormEscolaBD.BotaoFecharClick(Sender: TObject);

begin
  FormEscolaBD.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure ModoDEedicao (modo : boolean);

begin
    FormEscolaBD.DBEditNumero.ReadOnly := not modo;
    FormEscolaBD.DBEditCep.ReadOnly := not modo;
    FormEscolaBD.DBEditRua.ReadOnly := not modo;
    FormEscolaBD.DBEditComp.ReadOnly := not modo;
    FormEscolaBD.DBEditBairro.ReadOnly := not modo;
    FormEscolaBD.DBEditCidade.ReadOnly := not modo;
    FormEscolaBD.DBEditEstado.ReadOnly := not modo;
    FormEscolaBD.DBEditDiretor.ReadOnly := not modo;
    FormEscolaBD.DBEditAval.ReadOnly := not modo;
    FormEscolaBD.DBEditCorrente.ReadOnly := not modo;
    FormEscolaBD.DBEditAnoConstru.ReadOnly := not modo;
    FormEscolaBD.DBEditNome.ReadOnly := not modo;
    FormEscolaBD.DBEditCoordX.ReadOnly := not modo;
    FormEscolaBD.DBEditCoordY.ReadOnly := not modo;
    FormEscolaBD.DBEditCodigo.ReadOnly := not modo;
    FormEscolaBD.DBEnergia.ReadOnly := not modo;
    FormEscolaBD.DBagua.ReadOnly := not modo;
    FormEscolaBD.DBesgoto.ReadOnly := not modo;
    FormEscolaBD.DBeduc_infantil.ReadOnly := not modo;
    FormEscolaBD.DBensino_fund.ReadOnly := not modo;
    FormEscolaBD.DBensino_ja.ReadOnly := not modo;
    FormEscolaBD.DBEditTel.ReadOnly := not modo;
    FormEscolaBD.DBEditNsalas.ReadOnly := not modo;
    FormEscolaBD.DBEditNcart.ReadOnly := not modo;
end;

procedure TFormEscolaBD.BotaoAtualizarClick(Sender: TObject);

begin
  ModoDEedicao (true);
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
  BotaoIncluir.Enabled := false;
  BotaoExcluir.Enabled := false;
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  BotaoAplicar.Enabled := true;
end;

function VerificaCamposOK : boolean;

var
  campoOK : boolean;
  msgerro : string[255];

begin
  msgerro := 'ERRO - Os seguintes campos estão em branco:'
             + #13#10#13#10;
  campoOK := true;
  if FormEscolaBD.DBEditNome.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Nome da Escola' + #13#10;
       end;
  if FormEscolaBD.DBEditDiretor.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Diretor' + #13#10;
       end;
  if FormEscolaBD.DBEditAval.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Avaliação' + #13#10;
       end;
  if FormEscolaBD.DBEditCorrente.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Corrente Pedagógica' + #13#10;
       end;
  if FormEscolaBD.DBEditAnoConstru.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Ano da Construção' + #13#10;
       end;
  if FormEscolaBD.DBEditCoordX.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Coordenada X' + #13#10;
       end;
  if FormEscolaBD.DBEditCoordY.Text = ''
  then begin
         campoOK := false;
```

```
        msgerro := msgerro + 'Coordenada Y' + #13#10;
      end;
  if FormEscolaBD.DBEditCodigo.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Código' + #13#10;
      end;
  if FormEscolaBD.DBEditTel.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Telefone' + #13#10;
      end;
  if FormEscolaBD.DBEditNsalas.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Número de Salas' + #13#10;
      end;
  if FormEscolaBD.DBEditNcart.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Número de Carteiras' + #13#10;
      end;
  if FormEscolaBD.DBEditRua.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Rua' + #13#10;
      end;
  if FormEscolaBD.DBEditNumero.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Número' + #13#10;
      end;
  if FormEscolaBD.DBEditComp.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Complemento' + #13#10;
      end;
  if FormEscolaBD.DBEditBairro.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Bairro' + #13#10;
      end;
  if FormEscolaBD.DBEditCidade.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Cidade' + #13#10;
      end;
  if FormEscolaBD.DBEditEstado.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'Estado' + #13#10;
      end;
  if FormEscolaBD.DBEditCep.Text = ''
  then begin
        campoOK := false;
        msgerro := msgerro + 'CEP' + #13#10;
      end;
  if not campoOK then showmessage (msgerro);
  {Veirifcar código}
  VerificaCamposOK := campoOK;
end;

procedure TFormEscolaBD.BotaoAplicarClick(Sender: TObject);

begin
  if (EstouIncluindo and VerificaCamposOK) or not EstouIncluindo
  then begin
        EstouIncluindo := false;
        ModoDEedicao (false);
        BotaoPesquisar.Enabled := true;
        BotaoFechar.Enabled := true;
        BotaoIncluir.Enabled := true;
        BotaoExcluir.Enabled := true;
        BotaoAtualizar.Enabled := true;
        ComboPalavraChave.Enabled := true;
        Npalavrachave.Enabled := true;
        DBnavegador.Enabled := true;
        if Form1.TabTurma.RecordCount <> 0
        then begin
              showmessage ('O processamento anterior das turmas será apagado.');
              ApagarProcessamentoAnterior;
              showmessage ('O processamento anterior das turmas foi apagado.');
             end;
      end;
  BotaoAplicar.Enabled := false;
end;

procedure TFormEscolaBD.BotaoIncluirClick(Sender: TObject);

begin
  EstouIncluindo := true;
  ConsultaEscola.TabEscola.Append;
  ModoDEedicao (true);
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
  BotaoAtualizar.Enabled := false;
  BotaoIncluir.Enabled := false;
  BotaoExcluir.Enabled := false;
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  BotaoAplicar.Enabled := true;
end;

procedure TFormEscolaBD.BotaoExcluirClick(Sender: TObject);

begin
```

```
  if MessageDlg ('Confirma a exclusão do registro atual?',
                 mtConfirmation,mbOKCancel,0) = mrOK
  then begin
        ConsultaEscola.TabEscola.Delete;
        if Form1.TabTurma.RecordCount <> 0
        then begin
              showmessage ('O processamento anterior das turmas será apagado.');
              ApagarProcessamentoAnterior;
              showmessage ('O processamento anterior das turmas foi apagado.');
             end;
       end;
{excluir relacionamentos com a tabela prof-esc}
end;

end.
```

A.1.19 *Unit* UnitLogradouroBD

```
unit UnitLogradouroBD;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBCtrls, StdCtrls, ExtCtrls, Mask, OleCtrls, MapObjects2_TLB,
  Selog, ConsultAlun, Interface_Usuario;

type
  TFormLogradouroBD = class(TForm)
    DBEditLog: TDBEdit;
    DBEditQuad: TDBEdit;
    Nlog: TLabel;
    Nquad: TLabel;
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    BotaoIncluir: TButton;
    ComboPalavraChave: TComboBox;
    BotaoPesquisar: TButton;
    BotaoAtualizar: TButton;
    BotaoAplicar: TButton;
    BotaoExcluir: TButton;
    BotaoFechar: TButton;
    DBNavegador: TDBNavigator;
    DataSourceLog: TDataSource;
    BotaoSelQuad: TButton;
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure BotaoFecharClick(Sender: TObject);
    procedure ComboPalavraChaveClick(Sender: TObject);
    procedure BotaoExcluirClick(Sender: TObject);
    procedure BotaoIncluirClick(Sender: TObject);
    procedure BotaoAtualizarClick(Sender: TObject);
    procedure BotaoAplicarClick(Sender: TObject);
    procedure BotaoSelQuadClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormLogradouroBD : TFormLogradouroBD;
  EstouIncluindo   : boolean;

implementation

{$R *.DFM}

procedure TFormLogradouroBD.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   SelecLog.ADOTable1.first;
   ListaDEnomes := TStringList.Create;
   while not SelecLog.ADOTable1.Eof do
      begin
        straux := SelecLog.ADOTable1.FieldValues['LOGRADOURO'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (SelecLog.ADOTable1.FieldValues['LOGRADOURO']);
        SelecLog.ADOTable1.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure TFormLogradouroBD.BotaoFecharClick(Sender: TObject);

begin
  FormLogradouroBD.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TFormLogradouroBD.ComboPalavraChaveClick(Sender: TObject);

var
  achei : boolean;

begin
   achei := false;
   SelecLog.ADOTable1.first;
```

```
   while not (SelecLog.ADOTable1.Eof)
         and not achei do
       begin
         if SelecLog.ADOTable1.FieldValues['LOGRADOURO'] =
            ComboPalavraChave.Text
         then achei := true
         else SelecLog.ADOTable1.next;
       end;
end;

procedure TFormLogradouroBD.BotaoExcluirClick(Sender: TObject);

begin
  if MessageDlg ('Confirma a exclusão do registro atual?',
                 mtConfirmation,mbOKCancel,0) = mrOK
  then begin
        EstouIncluindo := false;
        DBEditLog.ReadOnly := true;
        BotaoPesquisar.Enabled := true;
        BotaoFechar.Enabled := true;
        BotaoIncluir.Enabled := true;
        BotaoAtualizar.Enabled := true;
        ComboPalavraChave.Enabled := true;
        Npalavrachave.Enabled := true;
        DBnavegador.Enabled := true;
        BotaoAplicar.Enabled := false;
        BotaoSelQuad.Enabled := false;
        SelecLog.ADOTable1.Delete;
       end;
end;

procedure TFormLogradouroBD.BotaoIncluirClick(Sender: TObject);
begin
  EstouIncluindo := true;
  SelecLog.ADOTable1.Append;
  DBEditLog.ReadOnly := false;
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
  BotaoAtualizar.Enabled := false;
  BotaoIncluir.Enabled := false;
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  BotaoAplicar.Enabled := true;
  BotaoSelQuad.Enabled := true;
end;

procedure TFormLogradouroBD.BotaoAtualizarClick(Sender: TObject);

begin
  DBEditLog.ReadOnly := false;
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
  BotaoIncluir.Enabled := false;
  BotaoExcluir.Enabled := false;
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  BotaoAplicar.Enabled := true;
  BotaoSelQuad.Enabled := true;
end;

function VerificaCamposOK : boolean;

var
  campoOK : boolean;

begin
  campoOK := not (FormLogradouroBD.DBEditLog.Text = '');
  if not campoOK
  then showmessage ('ERRO - O campo Logradouro está em branco!');
  VerificaCamposOK := campoOK;
end;

procedure TFormLogradouroBD.BotaoAplicarClick(Sender: TObject);

begin
  if (EstouIncluindo and VerificaCamposOK) or not EstouIncluindo
  then begin
        EstouIncluindo := false;
        DBEditLog.ReadOnly := true;
        BotaoPesquisar.Enabled := true;
        BotaoFechar.Enabled := true;
        BotaoIncluir.Enabled := true;
        BotaoExcluir.Enabled := true;
        BotaoAtualizar.Enabled := true;
        ComboPalavraChave.Enabled := true;
        Npalavrachave.Enabled := true;
        DBnavegador.Enabled := true;
        BotaoSelQuad.Enabled := false;
        Form1.Mapa_Austin.Refresh;
       end;
  BotaoAplicar.Enabled := false;
  FormLogradouroBD.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure TFormLogradouroBD.BotaoSelQuadClick(Sender: TObject);

begin
  showmessage ('Aperte a tecla Ctrl e o'+#13#10+
               'botão esquerdo do mouse para'+#13#10+
               'selecionar uma quadra no mapa.');
  FormLogradouroBD.Enabled := false;
  Form1.Enabled := true;
  Form1.SetFocus;
```

```
  Form1.Matrcula1.Enabled := false;
  Form1.Consultas.Enabled := false;
  Form1.BD.Enabled := false;
  Form1.Relatrios1.Enabled := false;
  Form1.BotaoConfSelQuad.Enabled := true;
  EstouSelecionandoQuadra := true;
end;

end.
```

A.1.20 *Unit* UnitProfEscBD

```
unit UnitProfEscBD;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, Db, ADODB, DBCtrls, Mask, ExtCtrls, UnitProfessorBD,
  ConsultAlun, Interface_Usuario, ConsultEsc, ConsultProf;

type
  TFormProfEscBD = class(TForm)
    DataSourceEscola: TDataSource;
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    ComboPalavraChave: TComboBox;
    DBEditNomeEsc: TDBEdit;
    Nescola: TLabel;
    DBEditCodigoEsc: TDBEdit;
    NcodigoEsc: TLabel;
    DBNavegador: TDBNavigator;
    QueryProfEsc: TADOQuery;
    BotaoIncluir: TButton;
    BotaoPesquisar: TButton;
    BotaoAplicar: TButton;
    BotaoAtualizar: TButton;
    BotaoExcluir: TButton;
    BotaoFechar: TButton;
    DataSourceProfEsc: TDataSource;
    Ndisciplina: TLabel;
    QueryProfEscDISCIPLINA: TWideStringField;
    QueryProfEscMAX_TURMAS: TIntegerField;
    DBEditMaxTur: TDBEdit;
    Nmaxtur: TLabel;
    Label1: TLabel;
    DBEditNomeProf: TDBEdit;
    NcodigoProf: TLabel;
    DBEditCodigoProf: TDBEdit;
    DBComboDisciplina: TDBComboBox;
    BotaoConcluir: TButton;
    procedure BotaoFecharClick(Sender: TObject);
    procedure BotaoAtualizarClick(Sender: TObject);
    procedure ComboPalavraChaveClick(Sender: TObject);
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure BotaoAplicarClick(Sender: TObject);
    procedure BotaoIncluirClick(Sender: TObject);
    procedure BotaoExcluirClick(Sender: TObject);
    procedure BotaoConcluirClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormProfEscBD  : TFormProfEscBD;
  EstouIncluindo : boolean;

implementation

{$R *.DFM}

procedure TFormProfEscBD.BotaoFecharClick(Sender: TObject);

begin
  FormProfEscBD.Visible := false;
  FormProfessorBD.Enabled := true;
  FormProfessorBD.SetFocus;
end;

procedure TFormProfEscBD.BotaoAtualizarClick(Sender: TObject);
begin
  DBComboDisciplina.ReadOnly := false;
  DBEditMaxTur.ReadOnly := false;
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
  BotaoIncluir.Enabled := false;
  BotaoExcluir.Enabled := false;
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  Form1.TabProfEsc.First;
  while not ((Form1.TabProfEsc.FieldValues['Cod_Esc'] = DBEditCodigoEsc.Text) and
             (Form1.TabProfEsc.FieldValues['Cod_Prof'] = DBEditCodigoProf.Text) and
             (Form1.TabProfEsc.FieldValues['Disciplina'] = DBComboDisciplina.Text))
    do Form1.TabProfEsc.next;
  BotaoAplicar.Enabled := true;
end;

procedure TFormProfEscBD.ComboPalavraChaveClick(Sender: TObject);

begin
  if not ConsultEsc.ConsultaEscola.TabEscola.Locate
```

```
        ('NOME',ComboPalavraChave.Text,[])
  then begin
         showmessage ('Erro no banco de dados ESCOLA');
         halt;
       end;
  QueryProfEsc.Active := false;
  QueryProfEsc.Parameters.ParamValues['CODPROF'] := DBEditCodigoProf.Text;
  QueryProfEsc.Parameters.ParamValues['CODESC'] := DBEditCodigoEsc.Text;
  QueryProfEsc.Active := true;
end;

procedure TFormProfEscBD.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   ConsultaEscola.TabEscola.first;
   ListaDEnomes := TStringList.Create;
   while not ConsultaEscola.TabEscola.Eof do
      begin
        straux := ConsultaEscola.TabEscola.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (ConsultaEscola.TabEscola.FieldValues['NOME']);
        ConsultaEscola.TabEscola.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

function VerificaCamposOK : boolean;

var
  campoOK : boolean;
  msgerro : string[255];

begin
  msgerro := 'ERRO - Os seguintes campos estão em branco:'
             + #13#10#13#10;
  campoOK := true;
  if FormProfEscBD.DBComboDisciplina.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Disciplina' + #13#10;
       end;
  if FormProfEscBD.DBEditMaxTur.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Número Máximo de Turmas' + #13#10;
       end;
  if not campoOK then showmessage (msgerro);
  VerificaCamposOK := campoOK;
end;

procedure TFormProfEscBD.BotaoAplicarClick(Sender: TObject);

begin
  if (EstouIncluindo and VerificaCamposOK) or not EstouIncluindo
  then begin
        if not EstouIncluindo then Form1.TabProfEsc.Edit;
        Form1.TabProfEsc.FieldValues['Disciplina'] := DBComboDisciplina.Text;
        Form1.TabProfEsc.FieldValues['Max_Turmas'] := DBEditMaxTur.Text;
        Form1.TabProfEsc.Post;
        DBComboDisciplina.Visible := false;
        DBEditMaxTur.Visible := false;
        Ndisciplina.Visible := false;
        Nmaxtur.Visible := false;
        BotaoConcluir.Visible := true;
        QueryProfEsc.Active := false;
        QueryProfEsc.Parameters.ParamValues['CODPROF'] := DBEditCodigoProf.Text;
        QueryProfEsc.Parameters.ParamValues['CODESC'] := DBEditCodigoEsc.Text;
{        QueryProfEsc.Active := true;}
        DBComboDisciplina.ReadOnly := true;
        DBEditMaxTur.ReadOnly := true;
        BotaoPesquisar.Enabled := true;
        BotaoFechar.Enabled := true;
        BotaoIncluir.Enabled := true;
        BotaoExcluir.Enabled := true;
        BotaoAtualizar.Enabled := true;
        ComboPalavraChave.Enabled := true;
        Npalavrachave.Enabled := true;
        DBnavegador.Enabled := true;
        if (Form1.TabTurma.RecordCount <> 0) and EstouIncluindo
        then begin
              showmessage ('O processamento anterior das turmas será apagado.');
              ApagarProcessamentoAnterior;
              showmessage ('O processamento anterior das turmas foi apagado.');
             end;
        EstouIncluindo := false;
       end;
  BotaoAplicar.Enabled := false;
end;

procedure TFormProfEscBD.BotaoIncluirClick(Sender: TObject);

begin
  EstouIncluindo := true;
  Form1.TabProfEsc.Append;
  DBComboDisciplina.ReadOnly := false;
  DBEditMaxTur.ReadOnly := false;
  Form1.TabProfEsc.FieldValues['Cod_Esc'] := DBEditCodigoEsc.Text;
  Form1.TabProfEsc.FieldValues['Cod_Prof'] := DBEditCodigoProf.Text;
  Form1.TabProfEsc.FieldValues['Turmas_Aloc'] := 0;
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
```

```
  BotaoAtualizar.Enabled := false;
  BotaoIncluir.Enabled := false;
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  BotaoAplicar.Enabled := true;
end;

procedure TFormProfEscBD.BotaoExcluirClick(Sender: TObject);

begin
  if MessageDlg ('Confirma a exclusão do registro atual?',
                 mtConfirmation,mbOKCancel,0) = mrOK
  then begin
        Form1.TabProfEsc.First;
        while not ((Form1.TabProfEsc.FieldValues['Cod_Esc'] = DBEditCodigoEsc.Text) and
                   (Form1.TabProfEsc.FieldValues['Cod_Prof'] = DBEditCodigoProf.Text) and
                   (Form1.TabProfEsc.FieldValues['Disciplina'] = DBComboDisciplina.Text))
          do Form1.TabProfEsc.next;
        Form1.TabProfEsc.Delete;
        if Form1.TabTurma.RecordCount <> 0
        then begin
              showmessage ('O processamento anterior das turmas será apagado.');
              ApagarProcessamentoAnterior;
              showmessage ('O processamento anterior das turmas foi apagado.');
             end;
       end;
end;

procedure TFormProfEscBD.BotaoConcluirClick(Sender: TObject);

begin
  if not ConsultEsc.ConsultaEscola.TabEscola.Locate
         ('NOME',ComboPalavraChave.Text,[])
  then begin
         showmessage ('Erro no banco de dados ESCOLA');
         halt;
       end;
  QueryProfEsc.Active := false;
  QueryProfEsc.Parameters.ParamValues['CODPROF'] := DBEditCodigoProf.Text;
  QueryProfEsc.Parameters.ParamValues['CODESC'] := DBEditCodigoEsc.Text;
  QueryProfEsc.Active := true;
  BotaoConcluir.Visible := false;
  DBComboDisciplina.Visible := true;
  DBEditMaxTur.Visible := true;
  Ndisciplina.Visible := true;
  Nmaxtur.Visible := true;
end;

end.
```

A.1.21 *Unit* UnitProfessorBD

```
unit UnitProfessorBD;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, StdCtrls, DBCtrls, Mask, ExtCtrls, ConsultProf,
  ConsultAlun, Interface_Usuario;

type
  TFormProfessorBD = class(TForm)
    Nprof: TLabel;
    Nemail: TLabel;
    NTEL: TLabel;
    Ncodigo: TLabel;
    PainelEndereco: TPanel;
    Nendereco: TLabel;
    NCEP: TLabel;
    Nrua: TLabel;
    Ncomplemento: TLabel;
    Ncidade: TLabel;
    Nbairro: TLabel;
    Nestado: TLabel;
    Nnumero: TLabel;
    DBEditNumero: TDBEdit;
    DBEditCep: TDBEdit;
    DBEditRua: TDBEdit;
    DBEditComp: TDBEdit;
    DBEditBairro: TDBEdit;
    DBEditCidade: TDBEdit;
    DBEditEstado: TDBEdit;
    DBEditEmail: TDBEdit;
    DBEditTel: TDBEdit;
    DBNavegador: TDBNavigator;
    DBEditNome: TDBEdit;
    DBEditCodigo: TDBEdit;
    PainelPrinc: TPanel;
    Npalavrachave: TLabel;
    ComboPalavraChave: TComboBox;
    DB1seg: TDBCheckBox;
    DB2seg: TDBCheckBox;
    DataSourceProf: TDataSource;
    BotaoIncluir: TButton;
    BotaoPesquisar: TButton;
    BotaoAplicar: TButton;
    BotaoAtualizar: TButton;
    BotaoExcluir: TButton;
    BotaoFechar: TButton;
    BotaoProfEsc: TButton;
    procedure BotaoPesquisarClick(Sender: TObject);
    procedure ComboPalavraChaveChange(Sender: TObject);
```

```
    procedure BotaoFecharClick(Sender: TObject);
    procedure BotaoAtualizarClick(Sender: TObject);
    procedure BotaoAplicarClick(Sender: TObject);
    procedure BotaoExcluirClick(Sender: TObject);
    procedure BotaoIncluirClick(Sender: TObject);
    procedure BotaoProfEscClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FormProfessorBD: TFormProfessorBD;
  EstouIncluindo : boolean;

implementation

uses
   UnitProfEscBD;

{$R *.DFM}

procedure TFormProfessorBD.BotaoPesquisarClick(Sender: TObject);

var
  straux       : string[255];
  ListaDEnomes : TStrings;

begin
   ConsultaProfessor.TabProfessor.first;
   ListaDEnomes := TStringList.Create;
   while not ConsultaProfessor.TabProfessor.Eof do
      begin
        straux := ConsultaProfessor.TabProfessor.FieldValues['NOME'];
        straux := UpperCase (RetiraAcentos (straux));
        if pos (UpperCase (RetiraAcentos (ComboPalavraChave.Text)),straux) <> 0
        then ListaDEnomes.Append (ConsultaProfessor.TabProfessor.FieldValues['NOME']);
        ConsultaProfessor.TabProfessor.next;
      end;
   ComboPalavraChave.Items := ListaDEnomes;
   ListaDEnomes.Free;
end;

procedure TFormProfessorBD.ComboPalavraChaveChange(Sender: TObject);

var
  achei : boolean;

begin
   achei := false;
   ConsultaProfessor.TabProfessor.first;
   while not (ConsultaProfessor.TabProfessor.Eof)
         and not achei do
      begin
        if ConsultaProfessor.TabProfessor.FieldValues['NOME'] =
           ComboPalavraChave.Text
        then achei := true
        else ConsultaProfessor.TabProfessor.next;
      end;
end;

procedure TFormProfessorBD.BotaoFecharClick(Sender: TObject);

begin
  FormProfessorBD.Visible := false;
  Form1.Enabled := true;
  Form1.SetFocus;
end;

procedure ModoDEedicao (modo : boolean);

begin
    FormProfessorBD.DBEditNumero.ReadOnly := not modo;
    FormProfessorBD.DBEditCep.ReadOnly := not modo;
    FormProfessorBD.DBEditRua.ReadOnly := not modo;
    FormProfessorBD.DBEditComp.ReadOnly := not modo;
    FormProfessorBD.DBEditBairro.ReadOnly := not modo;
    FormProfessorBD.DBEditCidade.ReadOnly := not modo;
    FormProfessorBD.DBEditEstado.ReadOnly := not modo;
    FormProfessorBD.DBEditNome.ReadOnly := not modo;
    FormProfessorBD.DBEditCodigo.ReadOnly := not modo;
    FormProfessorBD.DBEditTel.ReadOnly := not modo;
    FormProfessorBD.DBEditEmail.ReadOnly := not modo;
    FormProfessorBD.DB1seg.ReadOnly := not modo;
    FormProfessorBD.DB2seg.ReadOnly := not modo;
end;

procedure TFormProfessorBD.BotaoAtualizarClick(Sender: TObject);

begin
  ModoDEedicao (true);
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
  BotaoIncluir.Enabled := false;
  BotaoExcluir.Enabled := false;
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  BotaoAplicar.Enabled := true;
end;

function VerificaCamposOK : boolean;

var
  campoOK : boolean;
  msgerro : string[255];
```

```
begin
  msgerro := 'ERRO - Os seguintes campos estão em branco:'
             + #13#10#13#10;
  campoOK := true;
  if FormProfessorBD.DBEditNome.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Nome do Professor' + #13#10;
       end;
  if FormProfessorBD.DBEditEmail.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Endereço Eletrônico' + #13#10;
       end;
  if FormProfessorBD.DBEditCodigo.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Código' + #13#10;
       end;
  if FormProfessorBD.DBEditTel.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Telefone' + #13#10;
       end;
  if FormProfessorBD.DBEditRua.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Rua' + #13#10;
       end;
  if FormProfessorBD.DBEditNumero.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Número' + #13#10;
       end;
  if FormProfessorBD.DBEditComp.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Complemento' + #13#10;
       end;
  if FormProfessorBD.DBEditBairro.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Bairro' + #13#10;
       end;
  if FormProfessorBD.DBEditCidade.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Cidade' + #13#10;
       end;
  if FormProfessorBD.DBEditEstado.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'Estado' + #13#10;
       end;
  if FormProfessorBD.DBEditCep.Text = ''
  then begin
         campoOK := false;
         msgerro := msgerro + 'CEP' + #13#10;
       end;
  if not campoOK then showmessage (msgerro);
  {Veirifcar código}
  VerificaCamposOK := campoOK;
end;

procedure TFormProfessorBD.BotaoAplicarClick(Sender: TObject);

begin
  if (EstouIncluindo and VerificaCamposOK) or not EstouIncluindo
  then begin
        ModoDEedicao (false);
        BotaoPesquisar.Enabled := true;
        BotaoFechar.Enabled := true;
        BotaoIncluir.Enabled := true;
        BotaoExcluir.Enabled := true;
        BotaoAtualizar.Enabled := true;
        ComboPalavraChave.Enabled := true;
        Npalavrachave.Enabled := true;
        DBnavegador.Enabled := true;
        if (Form1.TabTurma.RecordCount <> 0) and EstouIncluindo
        then begin
              showmessage ('O processamento anterior das turmas será apagado.');
              ApagarProcessamentoAnterior;
              showmessage ('O processamento anterior das turmas foi apagado.');
             end;
        EstouIncluindo := false;
       end;
  BotaoAplicar.Enabled := false;
end;

procedure TFormProfessorBD.BotaoExcluirClick(Sender: TObject);

begin
  if MessageDlg ('Confirma a exclusão do registro atual?',
                 mtConfirmation,mbOKCancel,0) = mrOK
  then begin
        ConsultaProfessor.TabProfessor.Delete;
        if Form1.TabTurma.RecordCount <> 0
        then begin
              showmessage ('O processamento anterior das turmas será apagado.');
              ApagarProcessamentoAnterior;
              showmessage ('O processamento anterior das turmas foi apagado.');
             end;
{excluir relacionamentos com a tabela prof-esc}
       end;
end;
```

```
procedure TFormProfessorBD.BotaoIncluirClick(Sender: TObject);

var
   CodigoProfessor : cardinal;

begin
  EstouIncluindo := true;
  CodigoProfessor := 0;
  if ConsultaProfessor.TabProfessor.RecordCount <> 0
  then ConsultaProfessor.TabProfessor.first;
       while not ConsultaProfessor.TabProfessor.eof do
         begin
           if ConsultaProfessor.TabProfessor.FieldValues['COD_PROF']
              > CodigoProfessor
           then CodigoProfessor
                := ConsultaProfessor.TabProfessor.FieldValues['COD_PROF'];
           ConsultaProfessor.TabProfessor.Next;
         end;
  inc (CodigoProfessor);
  ConsultaProfessor.TabProfessor.Append;
  ModoDEedicao (true);
  ConsultaProfessor.TabProfessor.FieldValues['COD_PROF'] := CodigoProfessor;
  DBEditCodigo.ReadOnly := true;
  BotaoPesquisar.Enabled := false;
  BotaoFechar.Enabled := false;
  BotaoAtualizar.Enabled := false;
  BotaoIncluir.Enabled := false;
{  BotaoExcluir.Enabled := false;}
  ComboPalavraChave.Enabled := false;
  Npalavrachave.Enabled := false;
  DBnavegador.Enabled := false;
  BotaoAplicar.Enabled := true;
end;

procedure TFormProfessorBD.BotaoProfEscClick(Sender: TObject);

begin
  UnitProfEscBD.FormProfEscBD.ComboPalavraChave.Clear;
  UnitProfEscBD.FormProfEscBD.QueryProfEsc.Active := false;
  UnitProfEscBD.EstouIncluindo := false;
  UnitProfEscBD.FormProfEscBD.BorderIcons := UnitProfEscBD.FormProfEscBD.BorderIcons - [biMaximize];
  UnitProfEscBD.FormProfEscBD.BorderIcons := UnitProfEscBD.FormProfEscBD.BorderIcons - [biHelp];
  UnitProfEscBD.FormProfEscBD.BorderIcons := UnitProfEscBD.FormProfEscBD.BorderIcons - [biMinimize];
  UnitProfEscBD.FormProfEscBD.BorderIcons := UnitProfEscBD.FormProfEscBD.BorderIcons - [biSystemMenu];
  UnitProfEscBD.FormProfEscBD.Visible := true;
  FormProfessorBD.Enabled := false;
end;

end.
```

A.1.22 *Unit* UnitRelAluNaoMatri

```
unit UnitRelAluNaoMatri;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, Qrctrls, QuickRpt, ExtCtrls;

type
  TQRMDFormAluNaoMatri = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape2: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRDBText2: TQRDBText;
    QRLabel3: TQRLabel;
    QRDBText3: TQRDBText;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel9: TQRLabel;
    QRLabel10: TQRLabel;
    QRLabel11: TQRLabel;
    QRLabel12: TQRLabel;
    QRLabel13: TQRLabel;
    QRLabel14: TQRLabel;
    QRLabel15: TQRLabel;
    QRLabel16: TQRLabel;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText8: TQRDBText;
    QRDBText9: TQRDBText;
    QRDBText10: TQRDBText;
    QRDBText11: TQRDBText;
    QRDBText12: TQRDBText;
    QRDBText13: TQRDBText;
    QRDBText14: TQRDBText;
    QRDBText15: TQRDBText;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
    ADOQuery1: TADOQuery;
    ADOQuery1NOME: TWideStringField;
```

```
    ADOQuery1COD_TURM: TIntegerField;
    ADOQuery1COD_ALUN: TIntegerField;
    ADOQuery1RUA: TWideStringField;
    ADOQuery1NUM: TWideStringField;
    ADOQuery1COMPL: TWideStringField;
    ADOQuery1BAIRRO: TWideStringField;
    ADOQuery1CIDADE: TWideStringField;
    ADOQuery1ESTADO: TWideStringField;
    ADOQuery1CEP: TWideStringField;
    ADOQuery1TEL1: TWideStringField;
    ADOQuery1SERIE: TWideStringField;
    ADOQuery1PAI: TWideStringField;
    ADOQuery1MAE: TWideStringField;
    ADOQuery1NASCIMENTO: TDateField;
    ADOQuery1TEL2: TWideStringField;
    ADOQuery1EMAIL: TWideStringField;
    ADOQuery1COORD_X: TIntegerField;
    ADOQuery1COORD_Y: TIntegerField;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormAluNaoMatri: TQRMDFormAluNaoMatri;

implementation

{$R *.DFM}

end.
```

A.1.23 *Unit* UnitRelBD1

```
unit UnitRelBD1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, Qrctrls, QuickRpt, ExtCtrls;

type
  TQRMDFormBD1 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape1: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
    ADOQuery1: TADOQuery;
    QRDBText2: TQRDBText;
    ADOQuery1NOME: TWideStringField;
    ADOQuery1DISCIPLINA: TWideStringField;
    QRGroup1: TQRGroup;
    QRLabel1: TQRLabel;
    QRDBText1: TQRDBText;
    QRBand2: TQRBand;
    ADOQuery1QPROF: TIntegerField;
    QRDBText3: TQRDBText;
    ADOQuery1MAXT: TFloatField;
    QRDBText4: TQRDBText;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    ADOQuery1NUM_SALAS: TIntegerField;
    QRDBText5: TQRDBText;
    QRExpr1: TQRExpr;
    QRLabel8: TQRLabel;
    QRLabel9: TQRLabel;
    ADOQuery1COD_ESC: TWideStringField;
    QRDBText6: TQRDBText;
    QRLabel10: TQRLabel;
    QRExpr2: TQRExpr;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  QRMDFormBD1: TQRMDFormBD1;

implementation

{$R *.DFM}

end.
```

A.1.24 *Unit* UnitRelBD2

```
unit UnitRelBD2;

interface
```

```
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, Qrctrls, QuickRpt, ExtCtrls;

type
  TQRMDFormBD2 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape2: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRLabel1: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel10: TQRLabel;
    QRLabel11: TQRLabel;
    QRLabel12: TQRLabel;
    QRLabel13: TQRLabel;
    QRLabel14: TQRLabel;
    QRLabel15: TQRLabel;
    QRLabel16: TQRLabel;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText9: TQRDBText;
    QRDBText10: TQRDBText;
    QRDBText11: TQRDBText;
    QRDBText12: TQRDBText;
    QRDBText13: TQRDBText;
    QRDBText14: TQRDBText;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
    ADOQuery1: TADOQuery;
    QRLabel3: TQRLabel;
    ADOQuery1COD_PROF: TIntegerField;
    ADOQuery1NOME: TWideStringField;
    ADOQuery1RUA: TWideStringField;
    ADOQuery1NUM: TWideStringField;
    ADOQuery1COMPL: TWideStringField;
    ADOQuery1BAIRRO: TWideStringField;
    ADOQuery1CIDADE: TWideStringField;
    ADOQuery1ESTADO: TWideStringField;
    ADOQuery1CEP: TWideStringField;
    ADOQuery1TEL: TWideStringField;
    ADOQuery1EMAIL: TWideStringField;
    ADOQuery1DSDesigner1SEGM: TBooleanField;
    ADOQuery1DSDesigner2SEGM: TBooleanField;
    QRExpr1: TQRExpr;
    QRExpr2: TQRExpr;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormBD2: TQRMDFormBD2;

implementation

{$R *.DFM}

end.
```

A.1.25 *Unit* UnitRelBD3

```
unit UnitRelBD3;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, Qrctrls, QuickRpt, ExtCtrls;

type
  TQRMDFormBD3 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape2: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRLabel1: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel10: TQRLabel;
    QRLabel11: TQRLabel;
    QRLabel12: TQRLabel;
    QRLabel13: TQRLabel;
    QRLabel14: TQRLabel;
    QRLabel15: TQRLabel;
    QRLabel16: TQRLabel;
    QRDBText5: TQRDBText;
    QRDBText6: TQRDBText;
```

```
    QRDBText7: TQRDBText;
    QRDBText9: TQRDBText;
    QRDBText10: TQRDBText;
    QRDBText11: TQRDBText;
    QRDBText12: TQRDBText;
    QRDBText13: TQRDBText;
    QRDBText14: TQRDBText;
    QRLabel3: TQRLabel;
    QRExpr1: TQRExpr;
    QRExpr2: TQRExpr;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
    ADOQuery1: TADOQuery;
    ADOQuery1COD_ESC: TWideStringField;
    ADOQuery1NOME: TWideStringField;
    ADOQuery1RUA: TWideStringField;
    ADOQuery1NUM: TWideStringField;
    ADOQuery1COMPL: TWideStringField;
    ADOQuery1BAIRRO: TWideStringField;
    ADOQuery1CIDADE: TWideStringField;
    ADOQuery1UF: TWideStringField;
    ADOQuery1CEP: TWideStringField;
    ADOQuery1TEL: TWideStringField;
    ADOQuery1DIRETOR: TWideStringField;
    ADOQuery1ANO_CONST: TWideStringField;
    ADOQuery1NUM_SALAS: TIntegerField;
    ADOQuery1NUM_CART: TIntegerField;
    ADOQuery1ESGOTO: TBooleanField;
    ADOQuery1AGUA: TBooleanField;
    ADOQuery1ENERGIA: TBooleanField;
    ADOQuery1AVALIACAO: TWideStringField;
    ADOQuery1CORRENT_PEDA: TWideStringField;
    ADOQuery1EDU_INFANTIL: TBooleanField;
    ADOQuery1ENS_FUNDA: TBooleanField;
    ADOQuery1ENS_JOVENS_ADULT: TBooleanField;
    ADOQuery1SALAS_ALOC: TIntegerField;
    QRLabel2: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel9: TQRLabel;
    QRLabel17: TQRLabel;
    QRLabel18: TQRLabel;
    QRLabel19: TQRLabel;
    QRLabel20: TQRLabel;
    QRLabel21: TQRLabel;
    QRLabel22: TQRLabel;
    QRLabel23: TQRLabel;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRDBText8: TQRDBText;
    QRDBText15: TQRDBText;
    QRDBText16: TQRDBText;
    QRExpr3: TQRExpr;
    QRExpr4: TQRExpr;
    QRExpr5: TQRExpr;
    QRExpr6: TQRExpr;
    ADOQuery1ID: TIntegerField;
    ADOQuery1COORD_X: TFloatField;
    ADOQuery1COORD_Y: TFloatField;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormBD3: TQRMDFormBD3;

implementation

{$R *.DFM}

end.
```

A.1.26 *Unit* UnitRelCadAluno1

```
unit UnitRelCadAluno1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, Db, DBTables, QuickRpt, Qrctrls, ExtCtrls,
ADODB;

type
  TQRMDFormCadAluno1 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    ColumnHeaderBand1: TQRBand;
    ADOQuery1: TADOQuery;
    ADOQuery1NOME: TWideStringField;
    ADOQuery1COD_TURM: TIntegerField;
    QRSysData1: TQRSysData;
    QRBand1: TQRBand;
    QRDBText1: TQRDBText;
    QRShape2: TQRShape;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    QRDBText2: TQRDBText;
    QRLabel3: TQRLabel;
    ADOQuery1COD_ALUN: TIntegerField;
    ADOQuery1RUA: TWideStringField;
    ADOQuery1NUM: TWideStringField;
```

```
    ADOQuery1COMPL: TWideStringField;
    ADOQuery1BAIRRO: TWideStringField;
    ADOQuery1CIDADE: TWideStringField;
    ADOQuery1ESTADO: TWideStringField;
    ADOQuery1CEP: TWideStringField;
    ADOQuery1TEL1: TWideStringField;
    ADOQuery1SERIE: TWideStringField;
    ADOQuery1PAI: TWideStringField;
    ADOQuery1MAE: TWideStringField;
    ADOQuery1NASCIMENTO: TDateField;
    ADOQuery1TEL2: TWideStringField;
    ADOQuery1EMAIL: TWideStringField;
    ADOQuery1COORD_X: TIntegerField;
    ADOQuery1COORD_Y: TIntegerField;
    QRLabel4: TQRLabel;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel8: TQRLabel;
    QRLabel9: TQRLabel;
    QRLabel10: TQRLabel;
    QRLabel11: TQRLabel;
    QRLabel12: TQRLabel;
    QRLabel13: TQRLabel;
    QRLabel14: TQRLabel;
    QRLabel15: TQRLabel;
    QRLabel16: TQRLabel;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText6: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText8: TQRDBText;
    QRDBText9: TQRDBText;
    QRDBText10: TQRDBText;
    QRDBText11: TQRDBText;
    QRDBText12: TQRDBText;
    QRDBText13: TQRDBText;
    QRDBText14: TQRDBText;
    QRDBText15: TQRDBText;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormCadAluno1: TQRMDFormCadAluno1;

implementation

{$R *.DFM}


end.
```

A.1.27 *Unit* UnitRelCadAluno2

```
unit UnitRelCadAluno2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, DBTables, QuickRpt, Qrctrls, ExtCtrls, ADODB;

type
  TQRMDFormCadAluno2 = class(TForm)
    ADOQuery1: TADOQuery;
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape1: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel2: TQRLabel;
    ADOQuery1SERIE: TWideStringField;
    ADOQuery1QUANT: TIntegerField;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText1: TQRDBText;
    ADOQuery1NASCIMENTO: TDateField;
    QRShape2: TQRShape;
    QRShape3: TQRShape;
    ADOQuery2: TADOQuery;
    ADOQuery2TOTAL: TIntegerField;
    QRBand2: TQRBand;
    QRDBText4: TQRDBText;
    QRLabel4: TQRLabel;
    QRShape4: TQRShape;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
    QRLabel3: TQRLabel;
  private
    { Private declarations }
  public
    { Public declarations }
  end;
```

```
var
  QRMDFormCadAluno2: TQRMDFormCadAluno2;

implementation

{$R *.DFM}

end.
```

A.1.28 *Unit* UnitRelCadAluno3

```
unit UnitRelCadAluno3;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, Qrctrls, QuickRpt, ExtCtrls;

type
  TQRMDFormCadAluno3 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape1: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRLabel1: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel2: TQRLabel;
    QRShape2: TQRShape;
    QRShape3: TQRShape;
    QRBand1: TQRBand;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText1: TQRDBText;
    QRBand2: TQRBand;
    QRDBText4: TQRDBText;
    QRLabel4: TQRLabel;
    QRShape4: TQRShape;
    ADOQuery1: TADOQuery;
    ADOQuery1SERIE: TWideStringField;
    ADOQuery1QUANT: TIntegerField;
    ADOQuery2: TADOQuery;
    ADOQuery2TOTAL: TIntegerField;
    ADOQuery1BAIRRO: TWideStringField;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormCadAluno3: TQRMDFormCadAluno3;

implementation

{$R *.DFM}

end.
```

A.1.29 *Unit* UnitRelTurProc1

```
unit UnitRelTurProc1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Qrctrls, QuickRpt, ExtCtrls, Db, ADODB;

type
  TQRMDFormTurProc1 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape1: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
    QRGroup1: TQRGroup;
    QRLabel1: TQRLabel;
    QRDBText1: TQRDBText;
    QRLabel9: TQRLabel;
    QRDBText6: TQRDBText;
    QRBand2: TQRBand;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
    QRLabel5: TQRLabel;
    QRDBText2: TQRDBText;
    QRLabel2: TQRLabel;
    QRLabel3: TQRLabel;
    QRLabel4: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRDBText3: TQRDBText;
    ADOQuery1: TADOQuery;
    ADOQuery1ESCOLANOME: TWideStringField;
    ADOQuery1ALUNONOME: TWideStringField;
```

```
    ADOQuery1SERIE: TWideStringField;
    ADOQuery1COD_ESC: TWideStringField;
    ADOQuery1TURMANOME: TWideStringField;
    ADOQuery1COD_TURM: TIntegerField;
    QRLabel8: TQRLabel;
    ADOQuery1COD_ALUN: TIntegerField;
    QRDBText4: TQRDBText;
    QRDBText5: TQRDBText;
    QRDBText7: TQRDBText;
    QRDBText8: TQRDBText;
    ADOQuery1DISTANCIA: TIntegerField;
    ADOQuery1AREA_INFLU: TBooleanField;
    QRExpr1: TQRExpr;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormTurProc1: TQRMDFormTurProc1;

implementation

{$R *.DFM}

end.
```

A.1.30 *Unit* UnitRelTurProc2

```
unit UnitRelTurProc2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Qrctrls, Db, ADODB, QuickRpt, ExtCtrls;
type
  TQRMDFormTurProc2 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape1: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
    QRDBText2: TQRDBText;
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRGroup1: TQRGroup;
    QRLabel1: TQRLabel;
    QRDBText1: TQRDBText;
    QRLabel5: TQRLabel;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel9: TQRLabel;
    QRDBText6: TQRDBText;
    QRBand2: TQRBand;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
    ADOQuery1: TADOQuery;
    ADOQuery1NOME: TWideStringField;
    ADOQuery1COD_ESC: TWideStringField;
    ADOQuery1BAIRRO: TWideStringField;
    ADOQuery1SERIE: TWideStringField;
    ADOQuery1QUANT: TIntegerField;
    QRLabel2: TQRLabel;
    QRExpr1: TQRExpr;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormTurProc2: TQRMDFormTurProc2;

implementation

{$R *.DFM}

end.
```

A.1.31 *Unit* UnitRelTurProc3

```
unit UnitRelTurProc3;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  Db, ADODB, Qrctrls, QuickRpt, ExtCtrls;

type
  TQRMDFormTurProc3 = class(TForm)
    QuickRep1: TQuickRep;
    TitleBand1: TQRBand;
    QRSysData1: TQRSysData;
    QRShape1: TQRShape;
    ColumnHeaderBand1: TQRBand;
    QRBand1: TQRBand;
```

```
    QRDBText3: TQRDBText;
    QRDBText4: TQRDBText;
    QRGroup1: TQRGroup;
    QRLabel1: TQRLabel;
    QRDBText1: TQRDBText;
    QRLabel6: TQRLabel;
    QRLabel7: TQRLabel;
    QRLabel9: TQRLabel;
    QRDBText6: TQRDBText;
    QRBand2: TQRBand;
    QRLabel2: TQRLabel;
    QRBand3: TQRBand;
    QRSysData2: TQRSysData;
    QRShape5: TQRShape;
    ADOQuery1: TADOQuery;
    ADOQuery1NOME: TWideStringField;
    ADOQuery1COD_ESC: TWideStringField;
    ADOQuery1SERIE: TWideStringField;
    QRLabel3: TQRLabel;
    ADOQuery1QUANT: TIntegerField;
    QRLabel4: TQRLabel;
    ADOQuery2: TADOQuery;
    WideStringField1: TWideStringField;
    WideStringField2: TWideStringField;
    WideStringField3: TWideStringField;
    IntegerField1: TIntegerField;
    QRDBText2: TQRDBText;
    ADOQuery3: TADOQuery;
    WideStringField4: TWideStringField;
    WideStringField5: TWideStringField;
    WideStringField6: TWideStringField;
    IntegerField2: TIntegerField;
    QRDBText5: TQRDBText;
    QRExpr1: TQRExpr;
    QRExpr2: TQRExpr;
    QRExpr3: TQRExpr;
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  QRMDFormTurProc3: TQRMDFormTurProc3;

implementation

{$R *.DFM}

end.
```

ANEXO B

Formulários: Escola e Professor

B.1 Introdução

As informações que foram cadastradas nas tabelas professor e escola do banco de dados Matrigeo.mdb foram coletadas a partir dos formulários abaixo. Este foi um dos primeiros passos, antes mesmo de se iniciar a elaboração do sistema.

Os dois formulários foram idealizados levando-se em conta o modelo conceitual apresentado no capítulo 2 (figura 2). Os mesmos foram enviados aos diretores das escolas que ficaram responsáveis pelo preenchimento, seja direta ou indiretamente em relação aos de escolas e pela "cobrança" de preenchimento por parte dos professores do formulário dos mesmos.

B.1.1 Formulário Escola

Este formulário faz parte da pesquisa realizada por Janaina Eliza Fadel, mestranda do Curso de Pós Graduação em Engenharia de Computação, Área de concentração Geomática da Universidade do Estado do Rio de Janeiro (UERJ) e suas informações serão utilizadas como fonte de dados em sua dissertação de mestrado.

Dados Gerais sobre as Escolas

Favor preencher em letra de forma

Nome da Escola: Escola Municipal Althair Pereira de Moraes			
Endereço: Rua Cachoeira Grande – Três Fronteiras			
Número: 368	Complemento: –	Bairro: Antero	
Cidade: Nova Iguaçu		UF: RJ	Cep: 26.395 - 400
Tel:	E-mail:	Diretor (a): Adalice Freitas de Carvalho	
Ano Construção: 1996	Nº Prédio: –	Nº de salas: 09	Nº médio carteiras/sala (vagas):
Rede de Esgoto: (X) sim ( ) não	Rede de Água: (X) sim ( ) não		Energia Elétrica: (X) sim ( ) não
Avaliação: (X) Bimestral ( ) Trimestral ( ) Semestral	Total Alunos na escola: 598		Total Professores na escola: 22
Corrente Pedagógica: (X) Tradicional (X) Construtivismo ( ) Outros - Informar qual tipo de corrente:			
Nível Educação: (X) Educação Infantil (X) Ensino Fundamental ( ) Ensino Médio			

Informação sobre as séries, total de turmas e total de alunos por turno

Séries	Total de Turmas	Turmas - Manhã	Total Alunos	Turmas - Tarde	Total Alunos	Turmas - Noite	Total Alunos
Educ. Infantil	01	01	25	–	–		
1º Etapa	02	02	65	–	–		
2º Etapa	01	–	–	01	38		
3º Etapa	03	03	99	–	–		
3º Série	03	03	107	–	–		
4º Série	02	–	–	02	70		
5º Série	02	–	–	02	81		
6º Série	02	–	–	02	59		
7º Série	01	–	–	01	37		
8º Série	01	–	–	01	47		

B.1.2 Formulário Professor

Caro Professor da Rede Municipal de Ensino da Cidade de Nova Iguaçu, este formulário faz parte da pesquisa realizada por Janaina Eliza Fadel, mestranda do Curso de Pós Graduação em Engenharia de Computação, Área de concentração Geomática da Universidade do Estado do Rio de Janeiro (UERJ) e suas informações serão utilizadas como fonte de dados em sua dissertação de mestrado.

Dados Gerais sobre os Professores

Favor preencher em letra de forma

Nome: SANDRA DE SOUZA PEREIRA			
Rua: FLUMINENSE			No.: 88
Complemento: ----	Bairro: AUSTIN		Cidade: NOVA IGUAÇU
UF: RJ	Cep:	Telefone: 3765-4967	
Celular:	E-mail:		
Nome da (s) Escola(s) que leciona *: ESCOLA MUNICIPAL BETHIA PIMENTA DO MORAIS			
1a. Seguinte (1º A - 4a Série): ( ) sim ( ) não	2a. Seguinte (5a Série - 8a Série): (X) sim ( ) não		Ensino Médio: ( ) sim ( ) não

OBS: As informações são de suma importância para a realização da pesquisa. Somente as informações referentes aos campos telefone, celular e e-mail são facultativos, apesar de fazerem parte do banco de dados. Os demais campos deverão ser preenchidos.

* Quando o professor lecionar em mais de uma escola (somente escolas municipais da cidade de Nova Iguaçu), identificá-las com um número após o nome. Ex: E. M. Argentina (1) e E. M. Paraguai (2). Estes números irão identificar as escolas em que os professores lecionam na tabela a seguir.

Dados sobre Matérias Lecionadas

ESCOLA **	MATÉRIA	SÉRIES	TURMAS
	LÍNGUA PORTUGUESA	5º a 8º SÉRIES	501 502 601

** Esta coluna só deverá ser preenchida se o professor lecionar em mais de uma escola. Nesta coluna a escola será identificada pelo número associado ao nome da escola. Ex. Escola - 1; Matéria – Português; Séries - 5º. e 6º. Séries; Turmas - 501, 502, 601, 603.

ANEXO C

Relatórios dos Testes

C.1 Introdução

Neste anexo serão apresentados todos os resultados referente aos testes efetuados no capítulo 5, no formato de relatórios. Cada seção tem em seu título, a qual teste está associada, uma vez que um mesmo teste poderá apresentar mais de um resultado relevante.

Os relatórios gerados no **MatriGeo**, possuem numeração própria em suas páginas, mas para efeito de visualização dos resultados, esta numeração será suprimida.

C.1.1 Alunos Matriculados nas Escolas (teste1)

Alunos Matriculados nas Escolas

Escola CRECHE MUNICIPAL DE AUSTIN — **Código da Escola** C2

Nome do Aluno Gabriel Carlos Soares — **Código do Aluno** 5
Nome da Turma c10 — **Série** Creche — **Código da Turma** 5
Distância para a Escola (m) 3534 — **Área de Influência da Escola** Não

Escola ESCOLA MUNICIPAL ALTHAIR PIMENTA DE MORAES — **Código da Escola** 005

Nome do Aluno Catarina de Abreu — **Código do Aluno** 4
Nome da Turma 301 — **Série** Terceira Série — **Código da Turma** 4
Distância para a Escola (m) 196 — **Área de Influência da Escola** Sim

Escola ESCOLA MUNICIPAL DR. ODIR ARAUJO — **Código da Escola** 023

Nome do Aluno Antônio Pereira — **Código do Aluno** 2
Nome da Turma A — **Série** Terceira Etapa — **Código da Turma** 2
Distância para a Escola (m) 428 — **Área de Influência da Escola** Sim

Nome do Aluno Cássia Ferreira — **Código do Aluno** 1
Nome da Turma A — **Série** Segunda Etapa — **Código da Turma** 1
Distância para a Escola (m) 534 — **Área de Influência da Escola** Sim

Nome do Aluno Fábio Azevedo — **Código do Aluno** 3
Nome da Turma A — **Série** Primeira Etapa — **Código da Turma** 3
Distância para a Escola (m) 435 — **Área de Influência da Escola** Sim

Escola ESCOLA MUNICIPAL JOSE LUIZ DA SILVA — **Código da Escola** 043

Nome do Aluno Márcio Alexandre Santos — **Código do Aluno** 6
Nome da Turma 301 — **Série** Terceira Série — **Código da Turma** 6
Distância para a Escola (m) 29 — **Área de Influência da Escola** Sim

C.1.2 Alunos Matriculados nas Escolas (teste2)

Alunos Matriculados nas Escolas

Escola CRECHE MUNICIPAL DE AUSTIN **Código da Escola** C2

Nome do Aluno Gabriel Carlos Soares **Código do Aluno** 5
Nome da Turma c10 **Série** Creche **Código da Turma** 5
Distância para a Escola (m) 3534 **Área de Influência da Escola** Não

Escola ESCOLA MUNICIPAL ALTHAIR PIMENTA DE MORAES **Código da Escola** 005

Nome do Aluno Catarina de Abreu **Código do Aluno** 4
Nome da Turma 301 **Série** Terceira Série **Código da Turma** 4
Distância para a Escola (m) 196 **Área de Influência da Escola** Sim

Escola ESCOLA MUNICIPAL DR. ODIR ARAUJO **Código da Escola** 023

Nome do Aluno Anônio Pereira **Código do Aluno** 2
Nome da Turma A **Série** Terceira Etapa **Código da Turma** 2
Distância para a Escola (m) 428 **Área de Influência da Escola** Sim

Nome do Aluno Cássia Ferreira **Código do Aluno** 1
Nome da Turma A **Série** Segunda Etapa **Código da Turma** 1
Distância para a Escola (m) 534 **Área de Influência da Escola** Sim

Nome do Aluno Fábio Azevedo **Código do Aluno** 3
Nome da Turma A **Série** Primeira Etapa **Código da Turma** 3
Distância para a Escola (m) 435 **Área de Influência da Escola** Sim

Nome do Aluno Gabriela Carlos Soares **Código do Aluno** 7
Nome da Turma 301 **Série** Terceira Série **Código da Turma** 7
Distância para a Escola (m) 1210 **Área de Influência da Escola** Não

Escola ESCOLA MUNICIPAL JOSE LUIZ DA SILVA **Código da Escola** 043

Nome do Aluno Márcio Alexandre Santos **Código do Aluno** 6
Nome da Turma 301 **Série** Terceira Série **Código da Turma** 6
Distância para a Escola (m) 29 **Área de Influência da Escola** Sim

C.1.3 Quantidade de Professores nas Escolas (teste2)

Quantidade de Professores nas Escolas

Escola CRECHE MUNICIPAL DE AUSTIN
Quantidade de Salas 3 **Quantidade de Turnos** 1 **Código** C2

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Primeiro Segmento	3	3

Total de Turmas Possíveis de Primeiro Segmento 3
Total de Turmas Possíveis de Segundo Segmento 0

Escola CRECHE MUNICIPAL VILA SÃO MIGUEL
Quantidade de Salas 2 **Quantidade de Turnos** 1 **Código** C11

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Primeiro Segmento	2	2

Total de Turmas Possíveis de Primeiro Segmento 2
Total de Turmas Possíveis de Segundo Segmento 0

Escola ESCOLA MUNICIPAL ALTHAIR PIMENTA DE MORAES
Quantidade de Salas 9 **Quantidade de Turnos** 2 **Código** 005

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Artes	1	6
Ciências	2	6
Educação Física	1	6
Geografia	2	6
História	2	6
Inglês	1	6
Matemática	2	9
Português	2	6
Primeiro Segmento	9	9

Total de Turmas Possíveis de Primeiro Segmento 9
Total de Turmas Possíveis de Segundo Segmento 6

Escola ESCOLA MUNICIPAL DR. ODIR ARAUJO
Quantidade de Salas 4 **Quantidade de Turnos** 2 **Código** 023

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Primeiro Segmento	7	7

Total de Turmas Possíveis de Primeiro Segmento 7
Total de Turmas Possíveis de Segundo Segmento 0

Escola ESCOLA MUNICIPAL JOSÉ LUIZ DA SILVA
Quantidade de Salas 9 **Quantidade de Turnos** 2 **Código** 043

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Primeiro Segmento	20	20

Total de Turmas Possíveis de Primeiro Segmento 18
Total de Turmas Possíveis de Segundo Segmento 0

Escola ESCOLA MUNICIPAL NENA RODRIGUES
Quantidade de Salas 5 **Quantidade de Turnos** 2 **Código** 056

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Primeiro Segmento	10	10

Total de Turmas Possíveis de Primeiro Segmento 10
Total de Turmas Possíveis de Segundo Segmento 0

Escola ESCOLA MUNICIPAL PROF. ENILZA BARROS DOS SANTOS CHICONELLI
Quantidade de Salas 16 **Quantidade de Turnos** 2 **Código** 074

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Artes	2	11
Educação Física	2	11
Geografia	3	11
História	4	11
Inglês	2	11
Matemática	5	17
Português	4	12
Primeiro Segmento	21	21

Total de Turmas Possíveis de Primeiro Segmento 21
Total de Turmas Possíveis de Segundo Segmento 0

Escola ESCOLA MUNICIPAL PROF. MARCIO CAULINO SOARES
Quantidade de Salas 13 **Quantidade de Turnos** 2 **Código** 066

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Artes	3	15
Ciências	5	15
Educação Física	4	15
Geografia	4	15
História	5	15
Inglês	1	6
Matemática	5	18
Português	5	15
Primeiro Segmento	10	10

Total de Turmas Possíveis de Primeiro Segmento 10
Total de Turmas Possíveis de Segundo Segmento 6

Escola ESCOLA MUNICIPAL RUY BARBOSA
Quantidade de Salas 3 **Quantidade de Turnos** 2 **Código** 084

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Primeiro Segmento	21	21

Total de Turmas Possíveis de Primeiro Segmento 6
Total de Turmas Possíveis de Segundo Segmento 0

Escola ESCOLA MUNICIPAL SOUZA E MELLO
Quantidade de Salas 6 **Quantidade de Turnos** 2 **Código** 088

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Artes	1	5
Ciências	2	5
Educação Física	1	5
Geografia	1	5
História	1	5
Inglês	1	5
Matemática	2	5
Português	2	5
Primeiro Segmento	12	12

Total de Turmas Possíveis de Primeiro Segmento 12
Total de Turmas Possíveis de Segundo Segmento 5

Escola ESCOLA MUNICIPAL WALFREDO DA SILVA LESSA
Quantidade de Salas 11 **Quantidade de Turnos** 2 **Código** 094

Disciplina	Quantidade de Professores	Quantidade Máxima de Turmas
Artes	1	6
Ciências	2	6
Educação Física	1	6
Geografia	2	6
História	2	6
Inglês	1	6
Matemática	2	6
Português	2	6
Primeiro Segmento	16	16

Total de Turmas Possíveis de Primeiro Segmento 16
Total de Turmas Possíveis de Segundo Segmento 6

C.1.4 Cadastro de Aluno via Gera Teste para MatriGeo (teste3)

Alunos Matriculados nas Escolas

Escola	ESCOLA MUNICIPAL DR. ODIR ARAUJO	**Código da Escola**	023
Nome do Aluno	Aluno 01	**Código do Aluno**	1
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 02	**Código do Aluno**	2
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 03	**Código do Aluno**	3
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 04	**Código do Aluno**	4
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 05	**Código do Aluno**	5
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 06	**Código do Aluno**	6
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 07	**Código do Aluno**	7
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 08	**Código do Aluno**	8
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 09	**Código do Aluno**	9
Nome da Turma	301	**Série** Terceira Série	**Código da Turma** 1
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 10	**Código do Aluno**	10
Nome da Turma	302	**Série** Terceira Série	**Código da Turma** 2
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 100	**Código do Aluno**	100
Nome da Turma	303	**Série** Terceira Série	**Código da Turma** 3
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 101	**Código do Aluno**	101
Nome da Turma	303	**Série** Terceira Série	**Código da Turma** 3
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim
Nome do Aluno	Aluno 102	**Código do Aluno**	102
Nome da Turma	303	**Série** Terceira Série	**Código da Turma** 3
Distância para a Escola (m)	11	**Área de Influência da Escola**	Sim

C.1.5 Total de Alunos Matriculados nas Escolas (teste4)

Total de Alunos Matriculados nas Escolas em (Bairros X Séries)

Escola ESCOLA MUNICIPAL DR. ODIR ARAUJO		**Código** 023
Bairro	**Série**	**Quantidade de Alunos**
Bairro da escola	Terceira Série	245
Total de Alunos na Escola 245		
Escola ESCOLA MUNICIPAL SOUZA E MELLO		**Código** 088
Bairro	**Série**	**Quantidade de Alunos**
Bairro da escola	Terceira Série	255
Total de Alunos na Escola 255		

C.1.6 Alunos Matriculados nas Escolas (teste5)

Escola CRECHE MUNICIPAL VILA SAO MIGUEL		**Código da Escola** C11
Nome do Aluno Aluno 01		**Código do Aluno** 1
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 02		**Código do Aluno** 2
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 03		**Código do Aluno** 3
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 04		**Código do Aluno** 4
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 05		**Código do Aluno** 5
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 06		**Código do Aluno** 6
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 07		**Código do Aluno** 7
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 08		**Código do Aluno** 8
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 09		**Código do Aluno** 9
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim
Nome do Aluno Aluno 10		**Código do Aluno** 10
Nome da Turma c10	**Série** Creche	**Código da Turma** 1
Distância para a Escola (m) 463	**Área de Influência da Escola**	Sim

Nome do Aluno	Aluno 11		Código do Aluno 11
Nome da Turma	c10	Série Creche	Código da Turma 1
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 12		Código do Aluno 12
Nome da Turma	c10	Série Creche	Código da Turma 1
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 13		Código do Aluno 13
Nome da Turma	c10	Série Creche	Código da Turma 1
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 14		Código do Aluno 14
Nome da Turma	c10	Série Creche	Código da Turma 1
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 15		Código do Aluno 15
Nome da Turma	c10	Série Creche	Código da Turma 1
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 16		Código do Aluno 16
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 17		Código do Aluno 17
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 18		Código do Aluno 18
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 19		Código do Aluno 19
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 20		Código do Aluno 20
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 21		Código do Aluno 21
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 22		Código do Aluno 22
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 23		Código do Aluno 23
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 24		Código do Aluno 24
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 25		Código do Aluno 25
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 26		Código do Aluno 26
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 27		Código do Aluno 27
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim
Nome do Aluno	Aluno 28		Código do Aluno 28
Nome da Turma	c11	Série Creche	Código da Turma 2
Distância para a Escola (m)	463	Área de Influência da Escola	Sim

Nome do Aluno	Aluno 29		**Código do Aluno** 29
Nome da Turma	c11	**Série** Creche	**Código da Turma** 2
Distância para a Escola (m)	463	**Área de Influência da Escola**	Sim

Nome do Aluno	Aluno 30		**Código do Aluno** 30
Nome da Turma	c11	**Série** Creche	**Código da Turma** 2
Distância para a Escola (m)	463	**Área de Influência da Escola**	Sim

Escola CRECHE MUNICIPAL DE AUSTIN **Código da Escola** C2

Nome do Aluno	Aluno 31		**Código do Aluno** 31
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 32		**Código do Aluno** 32
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 33		**Código do Aluno** 33
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 34		**Código do Aluno** 34
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 35		**Código do Aluno** 35
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 36		**Código do Aluno** 36
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 37		**Código do Aluno** 37
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 38		**Código do Aluno** 38
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 39		**Código do Aluno** 39
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 40		**Código do Aluno** 40
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 41		**Código do Aluno** 41
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 42		**Código do Aluno** 42
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 43		**Código do Aluno** 43
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 44		**Código do Aluno** 44
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 45		**Código do Aluno** 45
Nome da Turma	c10	**Série** Creche	**Código da Turma** 3
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 46		**Código do Aluno** 46
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 47		**Código do Aluno** 47
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 48		**Código do Aluno** 48
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 49		**Código do Aluno** 49
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 50		**Código do Aluno** 50
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 51		**Código do Aluno** 51
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 52		**Código do Aluno** 52
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 53		**Código do Aluno** 53
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 54		**Código do Aluno** 54
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 55		**Código do Aluno** 55
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 56		**Código do Aluno** 56
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 57		**Código do Aluno** 57
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 58		**Código do Aluno** 58
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 59		**Código do Aluno** 59
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 60		**Código do Aluno** 60
Nome da Turma	c11	**Série** Creche	**Código da Turma** 4
Distância para a Escola (m)	1714	**Área de Influência da Escola**	Não

Nome do Aluno	Aluno 61		**Código do Aluno** 61
Nome da Turma	c12	**Série** Creche	**Código da Turma** 5
Distância para a Escola (m)	129	**Área de Influência da Escola**	Sim

Nome do Aluno	Aluno 62		**Código do Aluno** 62
Nome da Turma	c12	**Série** Creche	**Código da Turma** 5
Distância para a Escola (m)	129	**Área de Influência da Escola**	Sim

Nome do Aluno	Aluno 63		**Código do Aluno** 63
Nome da Turma	c12	**Série** Creche	**Código da Turma** 5
Distância para a Escola (m)	129	**Área de Influência da Escola**	Sim

Nome do Aluno	Aluno 64		**Código do Aluno** 64
Nome da Turma	c12	**Série** Creche	**Código da Turma** 5
Distância para a Escola (m)	129	**Área de Influência da Escola**	Sim

Nome do Aluno Aluno 65			**Código do Aluno** 65
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 66			**Código do Aluno** 66
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 67			**Código do Aluno** 67
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 68			**Código do Aluno** 68
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 69			**Código do Aluno** 69
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 70			**Código do Aluno** 70
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 71			**Código do Aluno** 71
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 72			**Código do Aluno** 72
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 73			**Código do Aluno** 73
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 74			**Código do Aluno** 74
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	
Nome do Aluno Aluno 75			**Código do Aluno** 75
Nome da Turma c12		**Série** Creche	**Código da Turma** 5
Distância para a Escola (m) 129		**Área de Influência da Escola** Sim	

C.1.7 Total de Alunos Matriculados nas Escolas (teste6)

Total de Alunos Matriculados nas Escolas em (Bairros X Séries)

Escola ESCOLA MUNICIPAL ALTHAIR PIMENTA DE MORAES		**Código** 005
Bairro	**Série**	**Quantidade de Alunos**
Bairro da escola	Quinta Série	210
Total de Alunos na Escola 210		
Escola ESCOLA MUNICIPAL WALFREDO DA SILVA LESSA		**Código** 094
Bairro	**Série**	**Quantidade de Alunos**
Bairro da escola	Quinta Série	35
Total de Alunos na Escola 35		

C.1.8 Alunos não Matriculados (teste7)

Alunos Não Matriculados

Nome	Aluno 76	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 76	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 76	**Telefone 1**	( ) - **Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número**	s/n **CEP**
Complemento	sem complemento	**Bairro**	Bairro da escola
Cidade	Nova Iguaçu	**Estado**	Rio de Janeiro

Nome	Aluno 77	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 77	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 77	**Telefone 1**	( ) - **Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número**	s/n **CEP**
Complemento	sem complemento	**Bairro**	Bairro da escola
Cidade	Nova Iguaçu	**Estado**	Rio de Janeiro

Nome	Aluno 78	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 78	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 78	**Telefone 1**	( ) - **Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número**	s/n **CEP**
Complemento	sem complemento	**Bairro**	Bairro da escola
Cidade	Nova Iguaçu	**Estado**	Rio de Janeiro

Nome	Aluno 79	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 79	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 79	**Telefone 1**	( ) - **Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número**	s/n **CEP**
Complemento	sem complemento	**Bairro**	Bairro da escola
Cidade	Nova Iguaçu	**Estado**	Rio de Janeiro

Nome	Aluno 80	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 80	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 80	**Telefone 1**	( ) - **Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número**	s/n **CEP**
Complemento	sem complemento	**Bairro**	Bairro da escola
Cidade	Nova Iguaçu	**Estado**	Rio de Janeiro

Nome	Aluno 81	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 81	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 81	**Telefone 1**	( ) - **Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número**	s/n **CEP**
Complemento	sem complemento	**Bairro**	Bairro da escola
Cidade	Nova Iguaçu	**Estado**	Rio de Janeiro

Nome	Aluno 82	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 82	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 82	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 83	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 83	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 83	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 84	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 84	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 84	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 85	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 85	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 85	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 86	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 86	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 86	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 87	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 87	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 87	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 88	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 88	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 88	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 89	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 89	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 89	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 90	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 90	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 90	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 91	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 91	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 91	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 92	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 92	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 92	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 93	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 93	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 93	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 94	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 94	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 94	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

Nome	Aluno 95	**Endereço Eletrônico**	usuario@dominio
Pai	Pai do aluno 95	**Data de Nascimento**	10/06/00 **Série** Creche
Mãe	Mãe do aluno 95	**Telefone 1** ( ) -	**Telefone 2** ( ) -
		ENDEREÇO	
Logradouro	Rua da escola	**Número** s/n	**CEP**
Complemento	sem complemento	**Bairro** Bairro da escola	
Cidade	Nova Iguaçu	**Estado** Rio de Janeiro	

C.1.9 Total de Alunos Matriculados nas Escolas (teste7)

Total de Alunos Matriculados nas Escolas em (Área de Influência X Séries)

Escola CRECHE MUNICIPAL DE AUSTIN			**Código** C2
Série	**Alunos**	**Alunos na Área de Influência**	**Alunos Fora da Área de Influência**
Creche	45	5	40
Totais de Alunos	45	5	40
Escola CRECHE MUNICIPAL VILA SAO MIGUEL			**Código** C11
Série	**Alunos**	**Alunos na Área de Influência**	**Alunos Fora da Área de Influência**
Creche	30	30	0
Totais de Alunos	30	30	0

ANEXO D

Ferramentas Utilizadas

D.1 Introdução

Neste anexo tem-se um pequeno tutorial para cada uma das ferramentas utilizadas na elaboração da dissertação. São elas: Microsoft Access 2000 [21,22], ArcView 3.2 [23], Delphi 5.0 [24, 25, 26, 27] e MapObjects 2.1 [28, 29].

O objetivo deste anexo é deixar o mais claro possível para um usuário (que muitas vezes é totalmente leigo em conhecimentos específicos em computação) a forma pela qual este projeto foi elaborado e como ele deverá proceder na utilização dos *softwares* mencionados acima.

D.2 Microsoft Access 2000

Com o Microsoft Accesss 2000 você poderá criar e editar tabelas, consultas, formulários, relatórios, macros e módulos. Não é necessário ter um conhecimento prévio sobre como usar uma aplicação de banco de dados, apesar deste pequeno tutorial assumir conhecimento de conceitos básicos de banco de dados.

D.2.1 Criando Tabelas

Neste tutorial, você criará primeiramente duas tabelas relacionadas, uma para Escola e outra para Professor.

a) Para criar um banco de dados novo, deve-se primeiramente abrir o Microsoft Access 2000.

b) No quadro *Create a new database using*, selecionar a opção *Blank Database* como mostrado na Figura 35.

c) Especifique um nome e localização para o banco de dados.

d) Ao clicar em criar o banco de dados, uma janela se abrirá com o nome de seu banco de dados, como mostrado na Figura 36.

e) Criar uma tabela chamada Escolas.

f) Caso ainda não esteja selecionado, marcar a opção *Tables* (na esquerda).

g) Clique duas vezes em *create table in design view*. A nova janela que abrirá é mostrada na Figura 37.

h) Ainda na Figura 37, deve-se: indicar uma chave primária para a tabela (em destaque), digitar o nome de cada campo na coluna *Field Name*, indicar que tipo de dado será composto tal campo (texto, booleano, número, etc.) e, se necessário fazer uma pequena descrição do campo na coluna *Description*.

i) No menu *File* selecione a opção *Save* e nomeie a tabela como Escola.

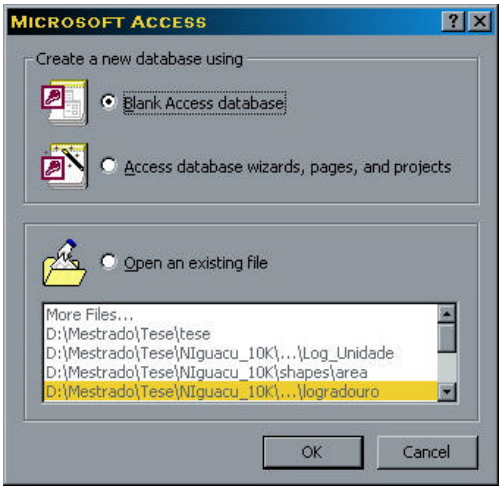


Figura 35 - Janela de Novo Banco de Dados

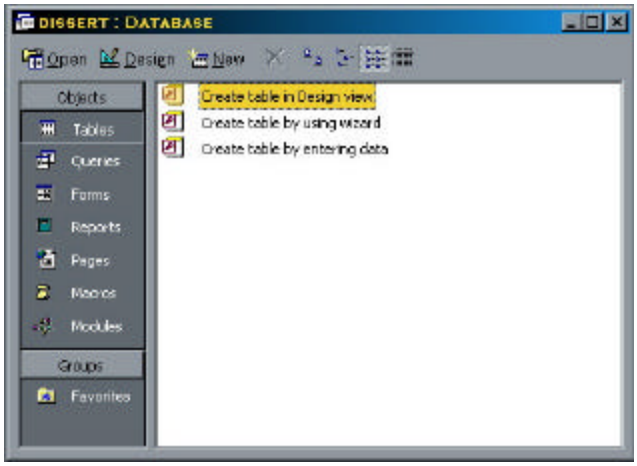
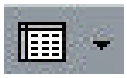


Figura 36 - Janela de Banco de Dados

j) Criar uma tabela chamada Professor e repita os mesmos passos descritos para a tabela Escolas.

k) Para entrar com dados nas tabelas Escola e Professor, deve-se após criar a estrutura da tabela, clicar no ícone *Datasheet View*  e começar a preenchê-la com seus dados, como na figura 38.

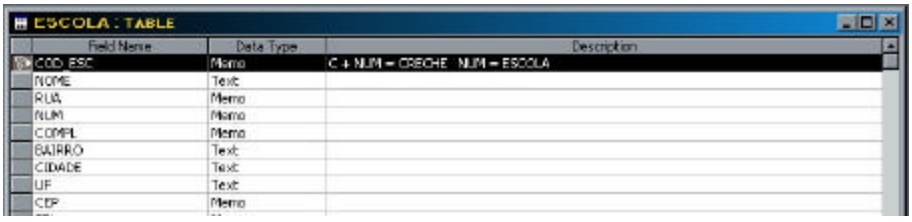


Figura 37 - Janela da Estrutura da Tabela Escola

Figura 38 – Tabela Escolas

D.2.2 Criando Relacionamentos

Depois de criar diferentes tabelas para cada assunto em seu banco de dados, será necessário arrumar um modo para juntar novamente aquela informação. Para realizar isto, serão criados relacionamentos (vínculos) entre as tabelas.

a) Para criar um relacionamento, primeiro deve-se selecionar no menu *Tools* a opção *Relationships*. Em seguida o menu *Relationships* será ativado e deve-se escolher a opção *ShowTable*. Após esse passo a janela apresentada na Figura 39 irá se abrir e deve-se selecionar as tabelas que irão fazer parte do relacionamento.

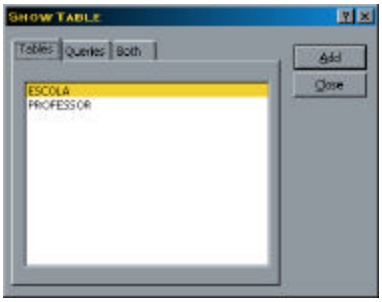


Figura 39 - Janela Mostrar Tabelas

b) Para criar o relacionamento, arraste o campo COD_ESC da tabela escola para a tabela professor. Ao fazer este passo a janela *Edit Relationships* (figura 40) se abrirá e deve-se clicar na checkbox *Enforce Referencial Integrity*.

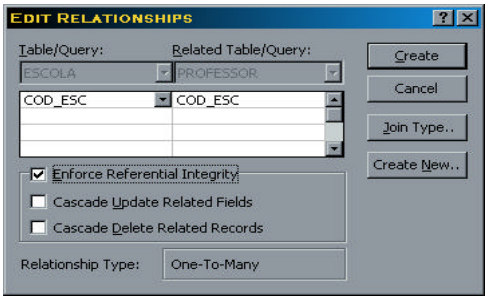


Figura 40 - Janela Edição de Relacionamentos

c) Clique no botão *Create* para criar o relacionamento.

d) O relacionamento entre as duas tabelas pode ser observado na figura 41.

Figura 41 - Janela de Relacionamentos

D.2.3 Criando Consultas

Na figura 36, na coluna da esquerda observa-se várias opções. Na seção D.2.1, *tables* encontrava-se em destaque e foi possível criar novas tabelas. Nesta seção, estará sendo abordada a construção de consultas e para tanto, o primeiro passo é destacar a palavra *Queries*. Pode-se elaborar as consultas de duas formas: utilizando um assistente, ou usando o modo de estrutura. E será esta última maneira que será abordada a seguir.

a) Com a opção *queries* destacada, basta dar um duplo clique em *Create query in design view*. A janela *Show Table* abrirá e deve-se selecionar as tabelas que irão fazer parte da consulta e clicar em *Add*, para adicioná-las.

b) Na janela *Select Query* (figura 42) deve-se: selecionar o campo que irá fazer parte da consulta (*field*), a tabela correspondente a tal campo (*table*), como serão dispostos os dados (*sort*), se o campo selecionado irá ficar ou não visível na consulta (*show*), e o critério de seleção, se for o caso (*criteria*).

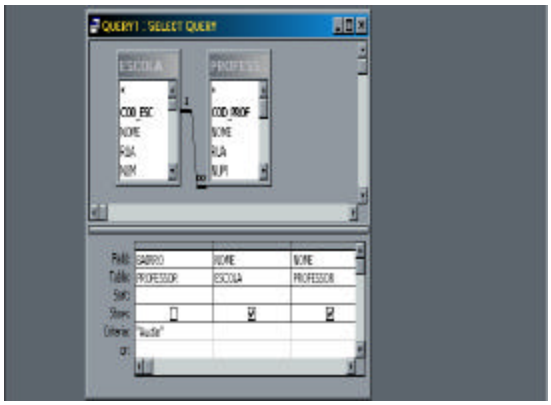


Figura 42 – Select Query

c) Pode-se arrastar a informação que fará parte da consulta diretamente da tabela, ou clicar nos campos. Nesse caso, abrirá as opções no próprio campo.

d) Exemplo: Selecionar todos os professores que morem no bairro "Austin", na tabela professores e mostrar o nome dos mesmos, como também das escolas na qual estes professores lecionam. O Resultado pode ser observado na figura 43.

Figura 43 – Resultado da Consulta

D.2.4 Criando Formulários

Nesta seção serão descritos os passos necessários para a elaboração de um formulário, através de um assistente. Como visto nas seções anteriores, na coluna da esquerda da figura 36, observa-se algumas opções.

Neste tutorial, já foram descritas as *Tables* e as *Queries*, e na presente seção é a vez dos *Forms*. As demais opções não serão abordadas.

a) Com a opção *Forms* destacada, basta dar um duplo clique em *Create form by using wizard*. A janela *Form Wizard* irá se abrir (figura 44). Deve-se selecionar o nome da tabela a qual o formulário corresponde no campo *Tables/Queries*. Pode-se escolher quais campos da tabela irão fazer parte do formulário. Em *Available Fields*, seleciona-se um ou todos os campos da tabela. Para adicionar um a um, ou apenas alguns campos utilize o botão >, e para incluir todos os campos da tabela no formulário, basta clicar uma única vez no botão >>. Clicar em *Next*.

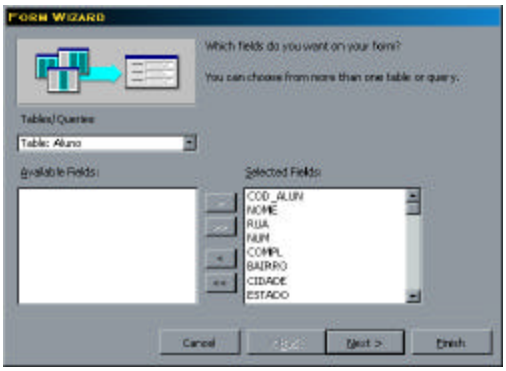


Figura 44 – *Form Wizard*: Escolha de Tabela

b) O próximo passo corresponde a escolher o formato do formulário, como pode ser observado na figura 45. Após selecionar a estrutura, clicar em *Next*.

c) Agora se deve escolher o estilo do formulário e clicar em *Next*. (figura 46)

d) Finalmente deve-se nomear o formulário e clicar no botão *Finish*. (figura 47)

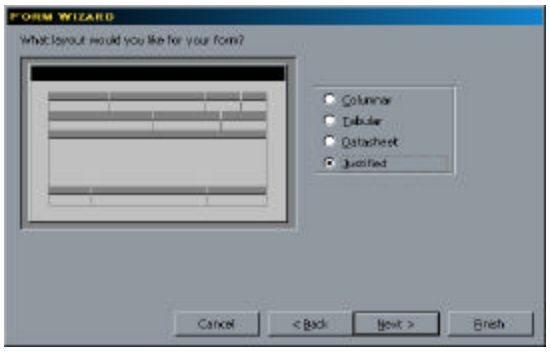


Figura 45 – *Form Wizard*: Estrutura do Formulário

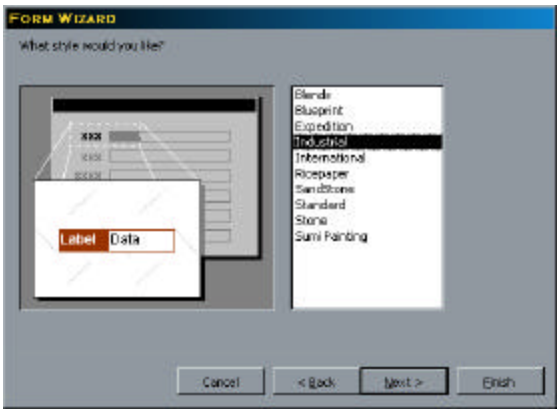


Figura 46 – *Form Wizard*: Estilo do Formulário

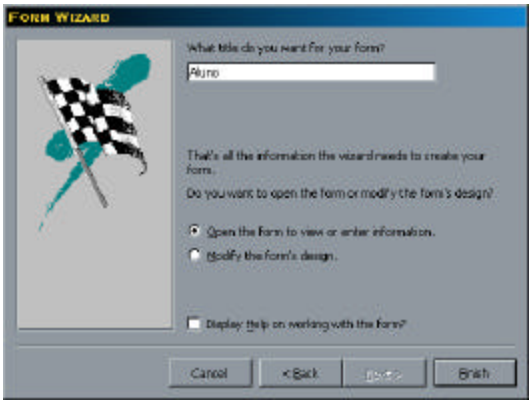


Figura 47 – *Form Wizard*: Nome do Formulário

Após os passos acima, será criado um formulário com as informações selecionadas no primeiro.(figura 48)

Figura 48 – Formulário Aluno

D.3 ArcView 3.2

Nos tópicos a seguir serão vistos alguns pontos relevantes para se entender como utilizar o ArcView 3.2. Estes são: como iniciar um projeto, tipos de temas, como adicionar um tema, tipos de *zoom*, cartografia temática, como criar novos temas, como utilizar arquivos CAD e como usar a opção GeoProcessing.

D.3.1 Iniciando um Projeto

Quando você inicia o ArcView, você está no projeto chamado *Untitled* (figura 49). Um projeto é todo o trabalho que você está fazendo em um problema particular, e pode consistir em vários mapas, tabelas, gráficos, *layouts*, e *Avenue*, que são *scripts* da linguagem de programação.

Atenção: Quando você salva um projeto, você não está salvando os dados, mas caminhos para os mesmos e um registro de qualquer mudança que você faz neles. Isto significa que você não pode levar o projeto a outro computador, a menos que os dados com os quais você esteja trabalhando também estejam disponíveis naquele computador. Se o ArcView não achar os dados no diretório especificado como parte do projeto salvo, serão indicados os caminhos corretos para os dados, assim, reconstruindo seu projeto.

D.3.2 Tipos de Temas

Você começa abrindo uma nova *View*, que você pode pensar como um mapa. Como mapas de papel, que podem ser constituídos de vários temas, uma *View* pode ser composta de vários temas. Cada tema representa um tipo de característica no mapa.

Os temas podem ser representados na forma de pontos, linhas e polígonos. Assim sendo um mapa poderá ter as três representações.

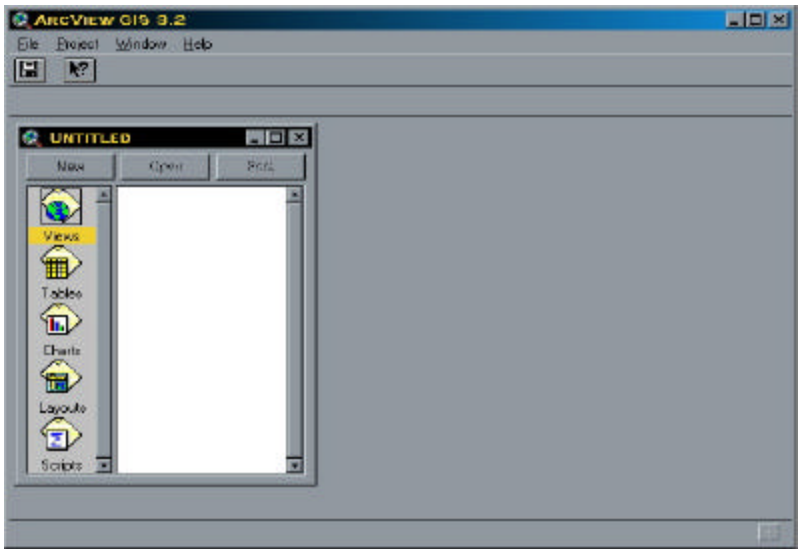


Figura 49 - Janela Principal

Os temas aparecem no mapa na ordem na qual eles aparecem à esquerda na barra de índice da janela *View* (figura 50). Se um tema é coberto por outro, pode-se torná-lo visível clicando e arrastando seu símbolo para cima, no índice.

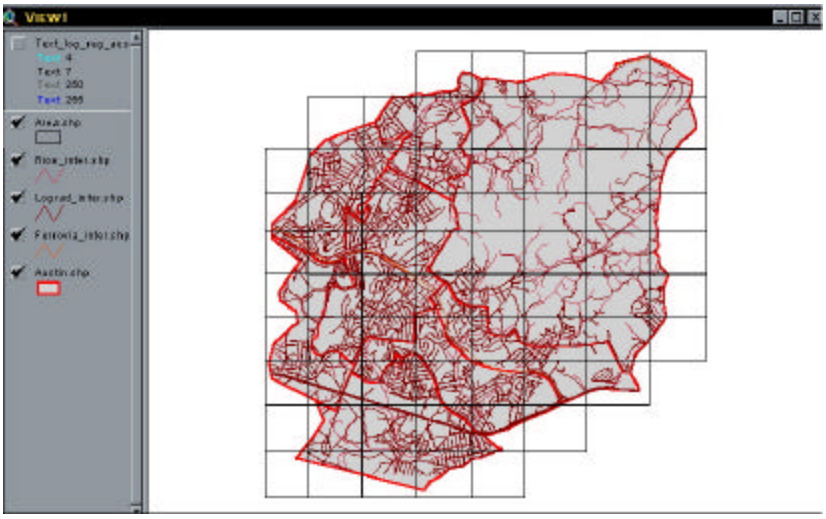


Figura 50 – *View* e Temas

D.3.3 Adicionando Temas a *View*

Criar uma nova *view*, clique duas vezes no ícone 

no menu do projeto. Isto abre uma nova *View*, vazia para adicionar novos temas.

Você adiciona qualquer tema na *View* apenas clicando no botão *Add Theme*  ou selecionando no menu *View*, a opção *Add Theme* .

Qualquer um destes abrirá uma ferramenta de navegação pela qual você pode selecionar o tema que você quer abrir (figura 51).

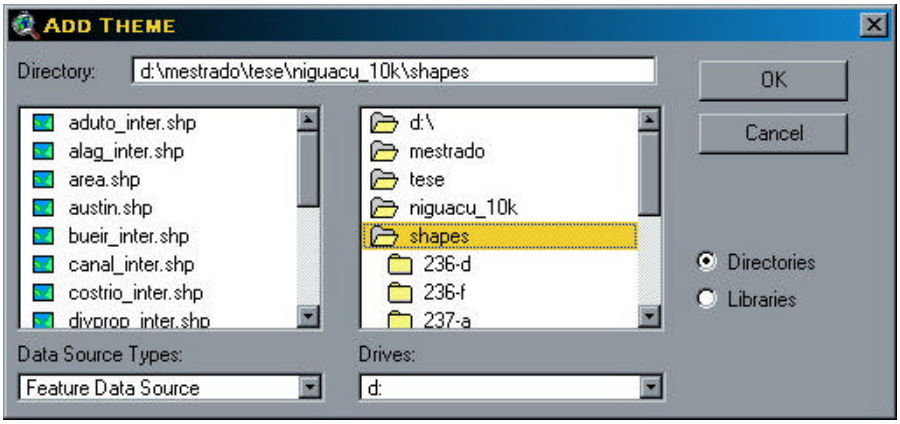


Figura 51 – Adicionar Temas

No espaço denominado de *Data Source Types*, escolhe-se que tipo de dado será adicionado a *view*: *Feature Data Source* refere-se a dados vetoriais (pontos, linhas, polígonos e textos), enquanto que *Image Data Source* refere-se a dados raster (imagem de satélite, figuras .jpg, .tiff etc). Ao adicionar outras extensões, será possível a inclusão de outras opções de formato de dados.

Depois de selecionar um tema, seu nome é exibido à esquerda na barra de índice da janela da *View*. Porém, o próprio tema não exibe na *View* até que você clique o *checkbox* à esquerda do nome do tema.

O formato nativo para dados vetor do ArcView é o *shapefile* que tem uma extensão *.shp*. O ArcInfo, que exporta arquivos de formato (.e00), também pode ser importado prontamente no ArcView. Outros formatos, como Cad's podem ser importados.

D.3.4 Tipos de *Zoom*

Você pode mudar o mapa através de vários tipos de *Zoom*, como descritos a seguir:

Tabela 1 – Descrição de Ferramentas

Nome Ferramenta	Ícone	Significado
Zoom In Tool		Aumenta a visualização, através de um *click*, ou por uma janela no mapa.
Zoom Out Tool		Diminui a visualização, através de um *click*, ou por uma janela no mapa.
Pan Tool		Arrasta o mapa.
Zoom In to Center Tool		Aumenta a visualização do mapa de forma centralizada.
Zoom Out from Center Tool		Diminui a visualização do mapa de forma centralizada.
Zoom to Full Extent of View Tool		*Zoom* total para a área de trabalho.
Zoom to Active Theme Tool		*Zoom* no tema ativo.
Zoom to Selected Feature Tool		Ao selecionar um tema, ao acionar este *zoom*, será localizado o tema em questão.
Zoom to Previous Extent Tool		Volta ao *zoom* anterior.

D.3.5 Mudanças de Unidades do Mapa

Para mudar a unidade do mapa, vá em *View* na barra de menu, e selecione a palavra *Properties*.

As propriedades da *View* incluem: o nome dela (mapa), sua projeção, quais unidades são usadas para medir as distâncias etc (figura 52).

Neste momento, você pode querer fixar o mapa e unidades de distância. Isto lhe permite medir distâncias no mapa, e também lhe permite criar uma barra de escala precisa para o *layout* final do mapa.

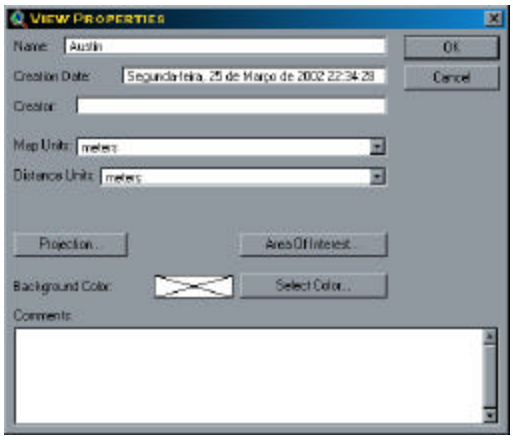


Figura 52 – Propriedades da *View*

D.3.6 Cartografia Temática

O tema “austin.shp” apresenta apenas uma coloração (cinza, com bordas em vermelho). Isto significa que todos os bairros da área aparecem no mapa com a mesma coloração. Para mudar isto, clique duas vezes na caixa cinza. Isto abre o Editor de Legenda (figura 53). Através do Editor de Legenda, pode-se mudar o modo de exibição de um tema, mudando a cor para todos as ocorrências, ou usando os valores de uma das colunas da tabela associada ao tema.

Considerando que "austin.shp" é um tema de polígono, uma caixa colorida aparece no Editor de Legenda. Se fosse um tema de linha, haveria uma linha colorida. Se fosse um tema de ponto, haveria um ponto. Para mudar a cor ou estilo do símbolo para as características clique duas vezes no desenho. Isto expõe uma paleta da qual você pode escolher cores, estilos, larguras de borda, e outros elementos de formato. Quando você clica em *Apply*, as mudanças aparecem no mapa.

Para deixar cada bairro com uma coloração diferente, deve-se abrir a linha correspondente a *Legend Type* e mude de *Single Symbol* para *Unique Value* (figura 54). No campo *Values Field*, selecione o campo da tabela que queira classificar o tema. No exemplo, o campo escolhido foi bairro. Assim sendo, cada bairro será classificado por uma cor diferente e, após clicar em *Apply*, o mapa passará a ter para cada bairro um tom diferente, não mais aparecerão todos com a tonalidade cinza.

Abrindo *Color Schemes*, você terá uma gama de combinações de cores, tais como tonalidades de azul, tons pastéis etc. Caso nenhuma das opções lhe agradar, você pode editar cada um individualmente, clicando duas vezes sobre o polígono. Irá abrir a palheta de estilos e você poderá selecionar a cor desejada para cada bairro.

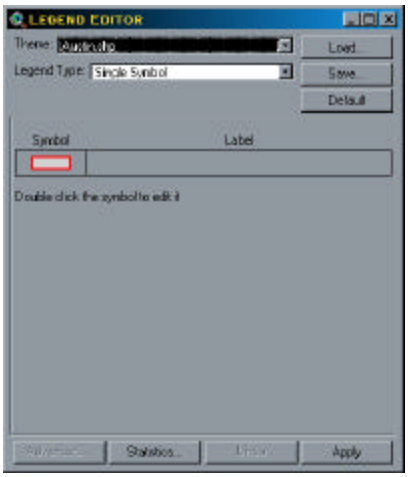


Figura 53 – Editor de Legenda

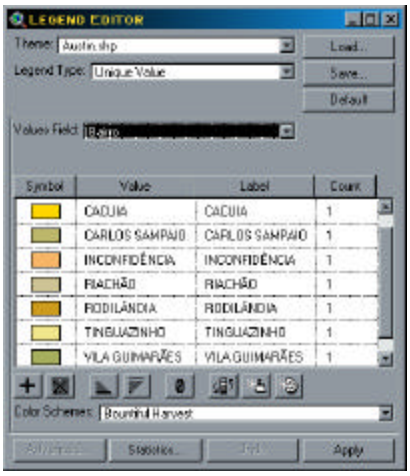


Figura 54 – Editor de Legendas

D.3.7 Adicionando seus Próprios Temas

Caso você queira incluir seus próprios temas, criá-los, basta ir ao menu *View* e clicar em *New Theme*. Após esse primeiro passo, você deverá selecionar que tipo de dado será (ponto, linha, ou polígono). Na figura abaixo está a demonstração da inclusão de um novo tema tipo ponto, denominado "escolas.shp". Após indicar o diretório no qual será salvo, você deverá selecionar o ícone em destaque (um ponto) e incluir o tema no seu mapa, na área desejada.

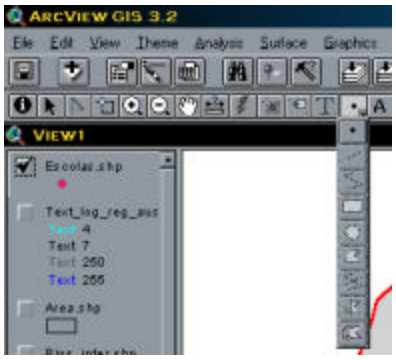


Figura 55 – Novo Tema Criado pelo Usuário

Ao se criar um novo tema, uma tabela automaticamente é gerada. Nela haverá um primeiro campo onde é indicado que tipo de tema está sendo criado (ponto, linha, polígono). E automaticamente, um segundo campo, denominado de ID. Você deverá entrar com a numeração desejada para indicar cada novo registro (figura 56).

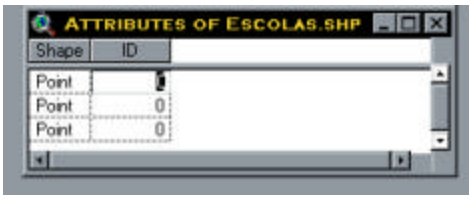


Figura 56 – Inclusão de Dados no Novo Tema

Você pode utilizar o botão  para deixar em edição a tabela e alterar os dados. No exemplo acima, caso você queira incluir o nome da escola, basta ir ao menu *Edit*, selecionar a opção *Add Field*, indicar que tipo de informação será incluída na tabela (booleano, numérico, string, ou data), definir o nome do campo e o tamanho do mesmo (figura 57).

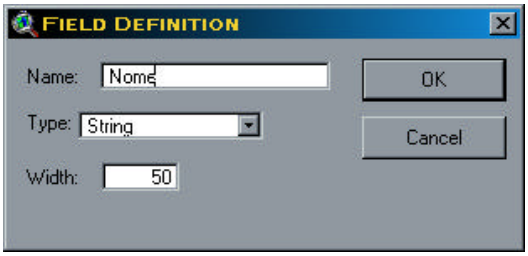


Figura 57 – Definição do Campo da Tabela

D.3.8 Utilizando Arquivos CAD

Arquivos de CAD são arquivos de desenho eletrônicos, criados em *software* como o AutoCAD da Autodesk (arquivos de extensão .dwg ou .dxf) ou o Microstation da Bentley (arquivos de extensão .dgn).

Você pode trabalhar com arquivos de CAD no ArcView, assim como trabalha com os arquivos *shapefile*. Ou seja, pode-se manipular e examinar um arquivo CAD como qualquer outro tema. Porém, você não pode editar ou modificar um arquivo CAD no ArcView. Para essa finalidade, deve-se converter os *layers* dos temas CAD em *shapfiles*. Entretanto, apenas as linhas, polígonos e pontos contidos em um arquivo de CAD podem ser convertidos em *shapefile*. O mesmo não é possível em relação aos textos (*annotation*).

Para poder ler estes arquivos, você deverá carregar a extensão *Cad Reader*. No menu *File*, selecionar a opção *extensions* (figura 58).

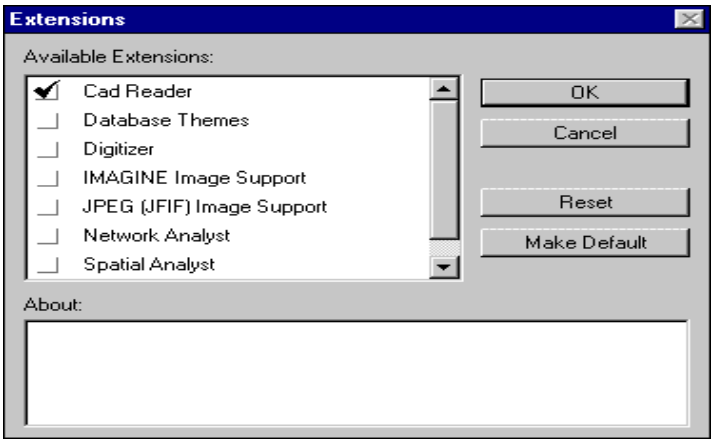


Figura 58 – Inclusão de Extensão

Agora, todos arquivos CAD poderão ser lidos no ArcView. O procedimento para adicionar um tema já foi mencionado anteriormente.

Cada arquivo CAD apresenta objetos linha, ponto, polígono e texto. Para adicionar cada um separadamente, deve-se clicar na pasta para abrir a opção de tipo de objeto (figura 59). Selecione as linhas. E por fim, clique em OK.

Para identificar cada *Layer*, ir ao menu *Theme* e selecionar a opção *Properties*. Na caixa de diálogo que se abre, clicar no botão *Drawing* (figura 60).

Quando se trabalha com *shapefiles*, na *View* basta clicar no *checkbox* para deixá-lo visível. Com os temas no formato CAD este procedimento é diferente. Como cada tema é composto de vários *Layers*, para deixá-los ativos é necessário selecionar os *layers* desejados na caixa de diálogo *Theme Properties*.

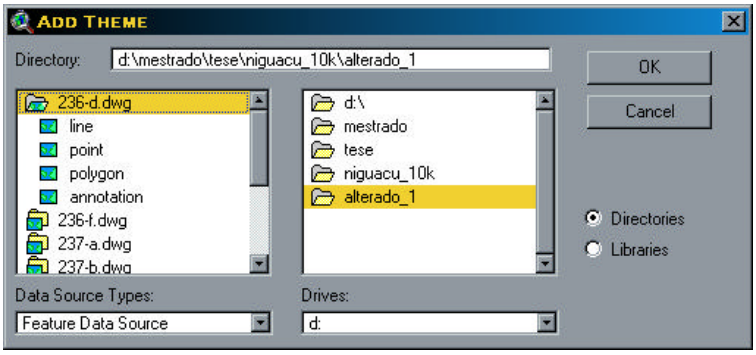


Figura 59 – Adicionar Tema

O próximo passo refere-se a selecionar no menu *Theme* a opção *Convert to Shapefile*. Após indicar o diretório onde será salvo o novo arquivo (agora com extensão .shp) aparece uma caixa de diálogo perguntando se você quer adicionar o *shapefile*

como tema na *View*. Clicar em *Yes* e assim, um novo tema será adicionado a *View* (figura 61).

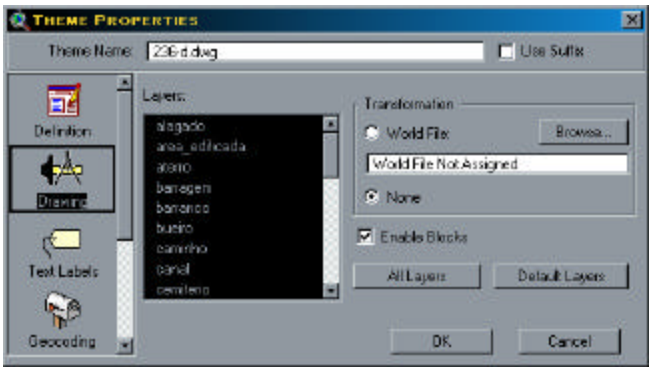


Figura 60 – Propriedades do Tema

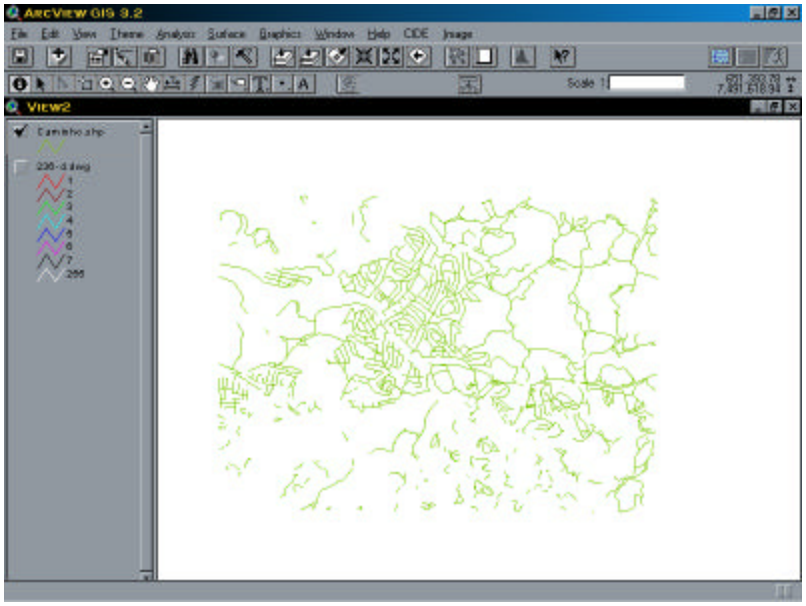


Figura 61 – Inclusão de Tema Cad convertido para Shapefile

D.3.9 GeoProcessing

O *GeoProcessing Wizard* irá auxiliar nas seguintes funções:

a) *Merge themes together*: lhe permite combinar as características de dois ou mais temas do mesmo tipo geométrico (ex.: polígono), e cria um tema novo com os atributos do primeiro tema.

b) *Clip one theme based on another*: lhe permite criar um tema novo usando um tema de polígono (ou seleciona polígonos naquele tema).

c) *Dissolve features based on an attribute*: lhe permite agregar features que têm o mesmo valor de atributo, e remove limites entre features iguais.

d) *Intersect two themes*: lhe permite criar um novo tema com sobreposição de dois temas distintos e preservando apenas as features que estejam dentro da extensão de espaço comum a ambos os temas.

e) *Union two themes*: lhe permite criar um tema novo sobrepondo dois temas e incluindo todas as características de ambos os temas, inclusive as features que não se sobrepõe.

f) *Assign data by location*: lhe permite executar uma junção espacial entre dois temas selecionados, usando um ou três relacionamentos, mais próximo, dentro de ou parte de, para unir a tabela de um atributo de um tema com a tabela de atributo de outro tema.

Neste tutorial, será apenas demonstrado como proceder ao *Intersect* entre dois temas.

Primeiramente você deve ir ao menu *View* e selecionar a opção *GeoProcessing Wizard*. Quando aparecer a caixa de diálogo representada pela figura 62, selecionar a opção *Intersect* two *Themes* e clicar *Next*.

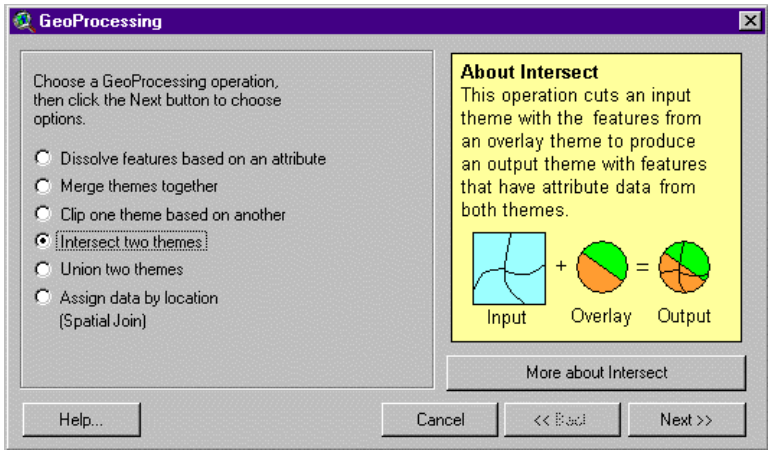


Figura 62 – Seleção de Operação

Na janela que se segue (figura 63), você irá selecionar os dois temas que serão sobrepostos e escolher o diretório no qual será salvo o novo *shapefile*.

Finalmente você irá clicar em *Finish*, esperar o arquivo ser processado e automaticamente o novo shapefile será adicionado a *View*.

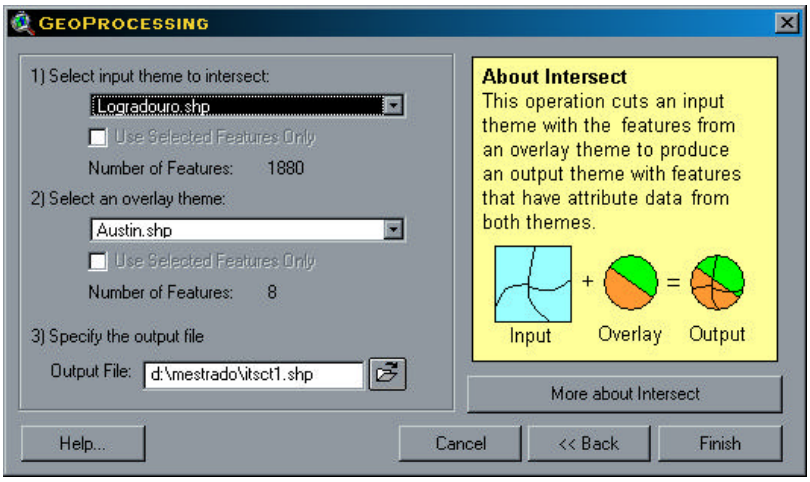


Figura 63 – Seleção de Temas e Diretório

Agora você poderá conferir as duas situações. Na figura 64 antes de usar o intersect e na figura 65, após o processamento do intersect entre os dois temas.

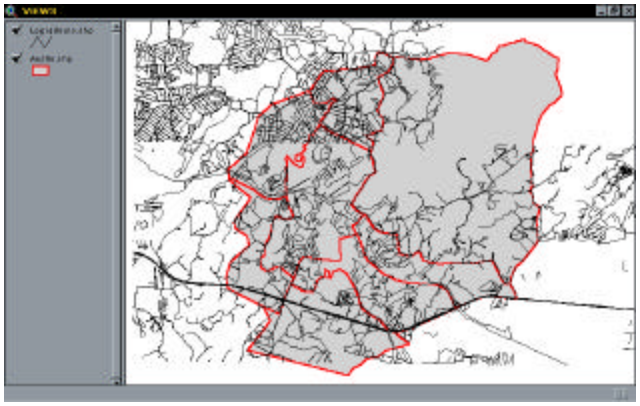


Figura 64 – Mapa antes do Intersect

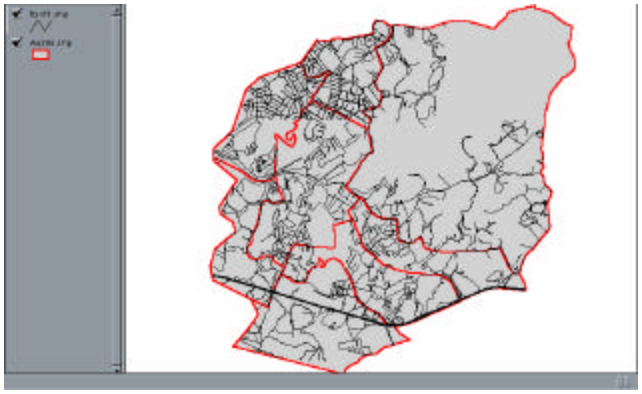


Figura 65 – Mapa após Intersect

D.4 Delphi 5.0

O Sistema Delphi é uma ferramenta de desenvolvimento que oferece facilidades para programação visual e que possui uma biblioteca de objetos classificados de acordo com suas características (Classe), e que possui também métodos e propriedades em sua estrutura. Permite construir aplicações orientadas a objetos com interfaces gráficas.

O código fonte das aplicações construídas nessa ferramenta, tanto a parte referente aos componentes visuais (Classes de Interface do usuário), quanto à parte que resolve o problema da aplicação (Classes de Negócio) é organizado em *units* (Unidades de Bibliotecas em Pascal).

A linguagem Pascal Objeto Delphi oferece três formas para controlar o acesso a membros de uma classe definida com o construtor "class", como segue:

a) Especificador "private" os membros "private" são visíveis somente na unidade (arquivo fonte "unit") onde estão declarados.

b) Especificador "protected": os membros "protected" podem ser utilizados somente pelos serviços da própria classe e de suas classes derivadas.

c) Especificador "public": os membros "public" podem ser utilizados por serviços da sua própria classe, serviços de suas classes derivadas, quaisquer serviços de outras classes ou funções de uma aplicação.

D.4.1 Alguns Componentes da Ferramenta Usados no Sistema

a) TADOQuery 

Use TADOQuery, da Palheta *ADO*, para ter acesso a uma ou mais tabelas em um banco de dados usando declarações em SQL.

b) TADOTable 

TADOTable, da palheta *ADO*, é usado para acessar dados de uma tabela em um banco de dados simples usando ADO. TADOTable provê acesso direto para todo registro e campo em uma tabela de banco de dados subjacente. Um componente ADO também pode trabalhar com um subconjunto de registros dentro de uma tabela de banco de dados que usa classes e filtros.

c) TButton 

TButton, da palheta *Standard*, é usado para colocar um botão em um formulário. TButton introduz várias propriedades para controlá-los através de uma caixa de diálogos. Os usuários escolhem o controle de botão para iniciar ações.

Para inserir um botão que exibe um bitmap em vez de um *label*, use TBitBtn.

Nota: Considerando que o *caption* de TButton está sempre centrado, a mudança do alinhamento do *BiDi* não tem nenhum efeito.

d) TCheckBox 

TCheckBox, da palheta *Standard*, é um componente que apresenta uma opção para o usuário. O usuário pode marcar a caixa para selecionar a opção, ou desmarcar para deselecionar a opção.

e) TComboBox 

TComboBox, da palheta *Standard*, é um componente que possui uma caixa de edição com uma seta que ao ser clicada abre um leque de opções, ou seja, possui uma lista associada a esse componente. Os usuários podem selecionar um item da lista ou podem digitar diretamente o item na caixa de edição.

f) TDataSource 

TDataSource, da palheta *Data Access*, é utilizado para fazer uma ligação entre um *dataset* e controles de *data-aware* em um formulário para habilitar a exibição, navegação, e edição dos dados que estão subordinados ao *dataset*.

Todos os *datasets* devem ser associados a um componente DataSource caso seus dados venham a ser exibidos e manipulados em controles *data-aware*. Semelhantemente, cada controle de *data-aware* precisa ser associado com um componente DataSource para que o controle receba e manipule dados.

g) TDBCheckBox 

Este componente TDBCheckBox, da palheta *Data Controls*, é muito semelhante ao TCheckBox, da palheta *Standard*. Os dois possuem a mesma aparência, uma caixa seguida de um *label*, na qual estarão sendo tratadas as informações booleanas de um certo dado (sim/não) através da seleção ou não de tal caixa. A diferença entre os dois

componentes está no fato de que neste, a caixa aparece selecionada ou não através dos dados vindos de uma tabela já existente (está associado a um *dataset*).

h) TDBEdit 

TDBEdit, da palheta *Data Controls*, é usado para permitir aos usuários a editar um campo do banco de dados. TDBEdit usa a propriedade de Texto para representar os conteúdos do campo.

TDBEdit permite somente uma única linha de texto. Caso o campo contenha dados longos que requeiram linhas múltiplas, use um objeto TDBMemo.

i) TDBNavigator 

O navegador de banco de dados é usado em formulários que contêm controles de *data-aware*, como TDBGrid ou TDBEdit. TDBNavigator (palheta *Data Controls*) proporciona para o usuário controle sobre o *dataset* quando editando ou vendo os dados.

Quando o usuário escolhe um dos botões do navegador, a ação apropriada acontece no *dataset* ao qual o navegador é ligado. Por exemplo, se o usuário clica o botão de Inserção, um registro em branco é inserido no *dataset*.

O TDBNavigator poderá mostrar todos os botões ou apenas alguns dos listados a seguir: Primeiro, Anterior, Próximo, Último, Inserir, Deletar, Post, Cancelar e Atualizar.

j) TEdit 

Um objeto de TEdit, da palheta *Standard*, é usado para pôr um controle de edição em um formulário. Controles de edição são usados para recuperar o texto que os usuários digitam. Controles de edição também podem exibir texto pré – definidos ao usuário.

TEdit implementa o comportamento genérico introduzido em TCustomEdit. TEdit publica muitas das propriedades herdadas de TCustomEdit, mas não introduz um novo procedimento. Para especializar os controles de edição, use classes descendentes ou derivadas de TCustomEdit.

k) TForm

O controle TForm é o principal componente de uma aplicação desenvolvida em Delphi, pois é nele que são inseridos os demais controles. Um controle do tipo TForm

pode ser usado como uma janela, uma caixa de diálogo ou qualquer tipo de formulário para entrada de dados.

Um formulário pode conter outros objetos, como TButton, TCheckBox, e objetos de TComboBox.

l) TLabel 

O componente TLabel, da palheta *Standard*, mostra um texto em uma área determinada. É utilizada normalmente pra nomear um outro componente.

O texto contido neste componente é determinado pela propriedade *Caption*.

m) TListBox 

TlistBox, da palheta *Standard*, implementa o procedimento genérico introduzido em TCustomListbox. TListBox publica muitas das propriedades herdadas de TCustomListBox, mas não introduz um novo procedimento.

Use TListBox para exibir uma lista de itens que os usuários podem selecionar, adicionar ou apagar.

n) TMainMenu 

TmainMenu, da palheta *Standard*, é usado para prover um menu principal para um formulário. Para começar a projetar um menu, adicione um menu principal ao formulário, e clique duas vezes no componente.

Não é necessário executar o programa para ver os resultados, uma vez que o menu está sempre visível no formulário com a mesma aparência que vai ter ao ser executado o programa.

o) TMap 

TMap, da palheta *Active X*, é o componente responsável pela entrada de um objeto do tipo mapa em uma aplicação. Algumas funcionalidades deste componente podem ser observadas na seção D.5.

p) TMaskEdit 

Um objeto TMaskEdit, da palheta *Additional*, é usado para pôr um controle de máscara de edição em um formulário. Estes controles de máscara irão validar o texto

que o usuário entrará na máscara. Um exemplo é usar um campo para entrada de data de aniversário, onde existe um controle de máscara no qual o usuário entra com caracteres numéricos, num espaçamento pré-definido. A máscara também pode formatar o texto que é exibido ao usuário.

TMaskEdit implementa um comportamento genérico introduzido em TCustomMaskEdit. TMaskEdit publica muitas das propriedades e métodos herdados de TCustomMaskEdit, mas não introduz um novo procedimento.

q) TPanel 

TPanel, da palheta *Standard*, é usado para pôr um painel vazio em um formulário.

TPanel implementa o procedimento genérico introduzido em TCustomPanel. TPanel publica muitas das propriedades herdadas de TCustomPanel, mas não introduz um novo procedimento.

r) TProgressBar 

Use *TprogressBar*, da palheta *Win 32*, para adicionar uma barra de progresso em um formulário. Barras de progresso proporcionam para os usuários uma visualização sobre o progresso de um procedimento dentro de uma aplicação. Como os progressos dos procedimentos, a barra de progresso retangular vai sendo preenchida gradativamente com uma cor em destaque, que no caso do sistema é azul.

s) TStatusBar 

O componente TStatusBar, da palheta *Win 32*, é uma série de painéis, normalmente alinhada na parte inferior do formulário, que mostra as informações sobre a aplicação quando está sendo executada.

t) TToolBar 

TToolBar, da palheta *Win 32*, é um componente que cria barras de ferramentas, onde serão inseridos botões e controles. Estes irão ser arranjados em linhas e automaticamente ajustados em seus tamanhos e posições.

D.5 MapObjects

MapObjects é um conjunto de componentes de *software* cartográficos que permite adicionar mapas em suas aplicações. Pode ser desenvolvido em várias linguagens de programação, tais como Delphi, Visual Basic, FoxPro, C++ e outras.

MapObjects inclui um controle ActiveX (OCX) chamado de controle de Mapa e um conjunto de mais de quarenta e cinco objetos de automatização ActiveX.

Programas desenvolvidos com MapObjects podem ser rodados em Windows 95, 98 e Windows NT 4 ou superiores.

A seguir, serão listadas algumas das funções que poderão ser implementadas em programas desenvolvidos com MapObjects:

- Exibir mapas com múltiplas camadas, como estradas, fluxos e limites.
- Funções de *zoom* e *pan* (deslocamento).
- Desenhar feições gráficas como pontos, linhas, elipses, retângulos e polígonos.
- Desenhar textos descritivos.
- Identificar as feições em um mapa ao apontá-los com o *mouse*.
- Selecionar feições ao longo de linhas e dentro de caixas, áreas, polígonos, e círculos.
- Selecionar feições num raio ou distância a partir de outra feição.
- Selecionar feições usando uma expressão de SQL.
- Calcular estatística básica em feições selecionadas.
- Consultas e atualização de atributos dos dados associados com as feições selecionadas.
- *Label* com textos de um determinado campo.
- Criar um novo *shapefile*.
- Arrastar imagens de fotografias aéreas ou imagens de satélite.
- Digitar um endereço e achar uma localização em um mapa.
- Projetar seus dados em sistemas de coordenada diferentes.
- Exibir dinamicamente dados em tempo real ou dados de série temporal.

Os dados geográficos que podem ser usados com MapObjects são: ESRI Shapfiles; ARC/INFO Coverages; SDE layers; CAD files; VPF files e StreetMap.

MapObjects proporciona para sua aplicação um caminho para acessar dados provenientes de uma variedade de fontes de arquivo e de outros bancos de dados. Os dados estão dentro de um objeto *Table* do MapObjects, que podem ser usados de várias formas. O uso mais comum de um novo objeto *Table* é associá-lo com dados de feições geográficas, fazendo um relacionamento.

Os dados podem ser acessados pela arquitetura de fonte de dados do Microsoft's OLE DB usando *Active X Data Objects*, ou através do Microsoft Jet database (.MDB), usando Data Access Objects (DAO) 3.5. MapObjects também faz acesso a banco de dados ArcSDE, ODBC, ISAM, e INFO.

A figura 66 representa um diagrama com as possíveis rotas conceituais usadas para ter acesso aos dados por um objeto *Table* do MapObjects.

Agora encontre a fonte de dados que deseja ter acesso e, então, siga o diagrama da figura 66 para ver quais as rotas que podem levar ao acesso de tais dados. Algumas fontes de dados podem ser acessadas por várias rotas. Por exemplo, bancos de dados Access podem ser acessados por DAO, ADO ou por ODBC. A escolha final de qual a melhor rota seguir vai depender de qual tecnologia de dados será instalada em seu computador, e qual a tecnologia você vai desejar utilizar na sua aplicação.

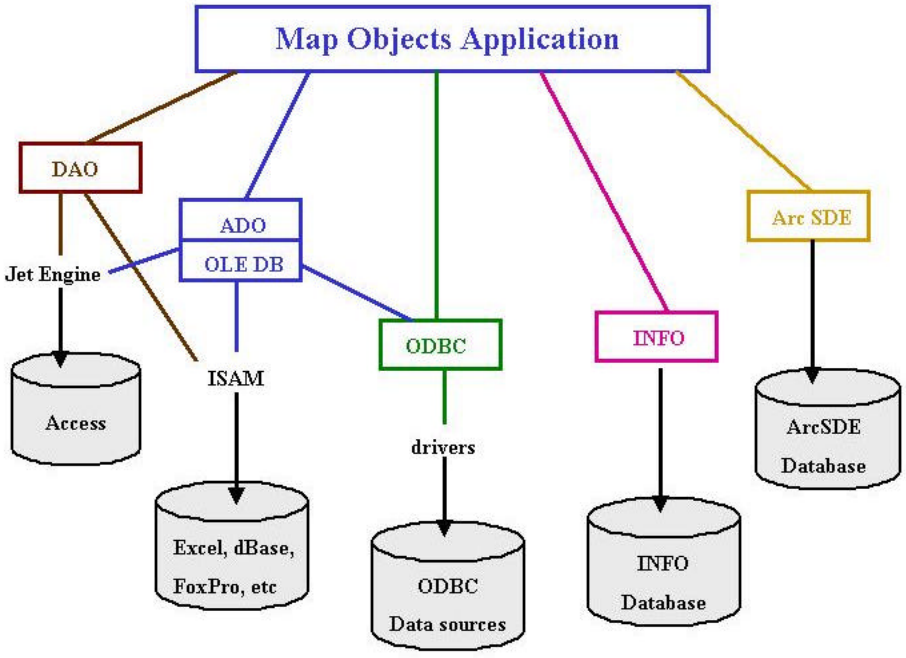


Figura 66 – Acesso a Banco de Dados

D.5.1 Carregando MapObjects

Inicie Delphi e escolha *Import ActiveX Control* no menu *Component*. Encontre na lista de controles disponíveis a referência ESRI MapObjects 2.1 (figura 67). Caso ele não estiver listado, clique em *Add* para incluir a referência do MapObjects 2.1 na lista. Finalmente, clique *Install* para reconstruir a biblioteca do componente.

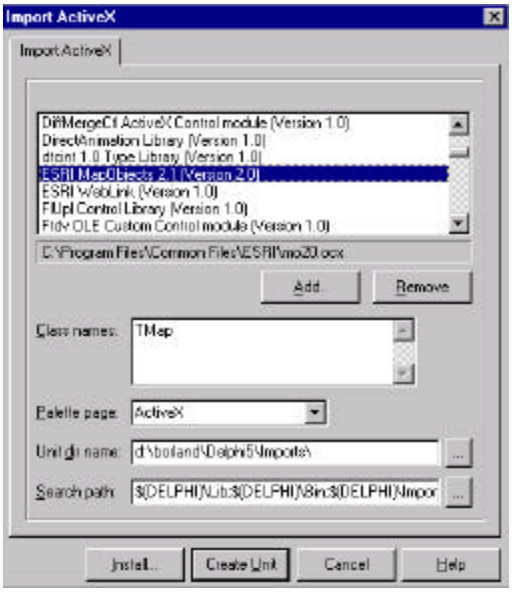


Figura 67 – Janela Import ActiveX

Uma nova ferramenta é incluída na lista de ícones na barra de ferramentas de componentes ActiveX (figura 68). Esta nova ferramenta é o *MapObjects 2.1 map control*.

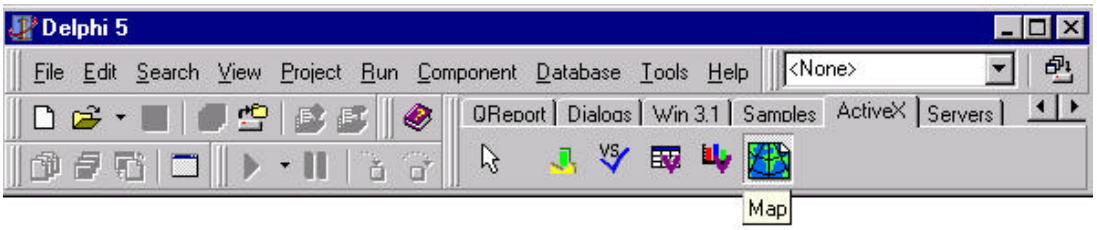


Figura 68 – Ferramenta ActiveX

D.5.2 Ajuda do MapObjects

O controle de mapa é um dos 48 objetos que compõem o MapObjects 2.1. Para obter informações sobre os vários objetos, abra a ajuda *online* do MapObjects e selecione o Objeto no índice.

O sistema de ajuda provê ajuda para todo objeto, propriedade, método, evento, e constante em MapObjects 2.1.

D.5.3 Adicionando um Mapa

Serão listados os passos necessários para que o usuário insira um mapa na sua aplicação.

D.5.3.1 Adicione o *Map Control* no *Form*

1. Clique duas vezes no *map control* na paleta de ActiveX para adicionar um novo mapa no *form*. (figura 69)
2. Reconfigure a extensão do mapa para preencher o *form*.

Figura 69 – Adicionar Mapa no *Form*

D.5.3.2 Adicione Dados no Mapa

Você pode especificar os dados que serão exibidos no mapa fixando propriedades na janela de propriedade do controle de mapa (figura 70).

1. Clique com o botão direito do *mouse* no mapa para exibir a propriedade de controle do mapa.

2. Escolha a opção *Properties* para exibir a janela de propriedade.

3. Clique *Add* e localize o diretório onde o dados que serão adicionados no mapa estão armazenados.

4. Escolha o arquivo e clique em *Apply*.

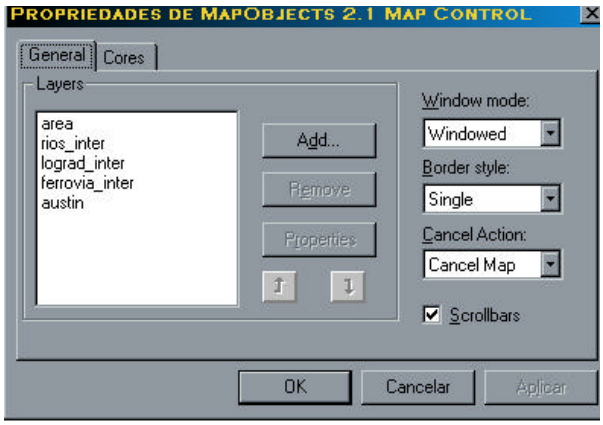


Figura 70 – Janela Propriedades do Controle de Mapa

D.5.3.3 Propriedades Fixas para as Camadas

1. Com a janela de propriedades do controle de mapa do MapObjects ainda aberta, selecione uma das camadas e clique em *Properties*.

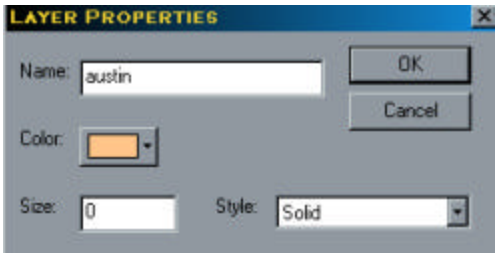


Figura 71 – Propriedades das Camadas

2. Clique no botão *Color* para selecionar uma cor para uma camada. (figura 71)

3. Clique no botão *OK* para fechar a caixa de diálogo.

D.5.3.4 Teste sua Aplicação

1. Clique no botão *Run* na barra de ferramentas do Delphi.
2. Deixar rodar sua aplicação.
3. O resultado da aplicação ao rodar é visualizado na figura 72.

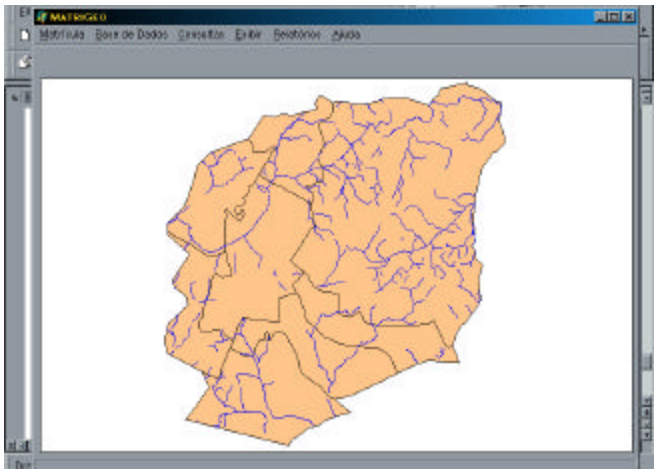


Figura 72 – Aplicação em uso

BIBLIOGRAFIA

[1] INSTITUTO MUNICIPAL DE URBANISMO PEREIRA PASSOS, Diretoria de Informações Geográficas, "Rio Atlas 98", Rio de Janeiro, 1998, Mídia Digital.

[2] FADEL, J.E. "Giz ou Gis na sala de aula? Utilização de Sistemas de Informação Geográfica no Ensino Médio". In: *XII Encontro Nacional de Geógrafos*, p. 249, Florianópolis, Julho. 2000.

[3] http://www.geomatica.eng.uerj.br/debates/geomatica.html

[4] http://www.library.wisc.edu/libraries/Steenbock/bipage/arcview/arcview.htm

[5] PIZZOLATO, N.D., BASSIL, K W., SOARES, T.S. "Avaliação da Localização de Escolas Públicas com Uso do SIG". In: *XXXI SBPO*, pp. 606-619, Juiz de Fora, 1999.

[6] PIZZOLATO, N.D., SILVA, H. B. F. "The Location of Public Schools: Evaluation of Practical Experiences", International Transactions in Operational Research, v.4, n.1, pp.13-22, 1997.

[7] PIZZOLATO, N.D. "A Heuristic for Large-Size P-Median Location Problems With Application to School Location". In: *Annals of Operations Research*, pp. 473-485, Estados Unidos, 1994.

[8] PIZZOLATO, N.D., MIZUBUTI, S., SILVA, G.G. "Avaliação da Oferta de Ensino Fundamental pela Rede Pública e da sua Distribuição Espacial: Aplicação ao Município de Niteroi, RJ", *Revista Brasileira de Estudos Pedagógicos* , v. 80, n.195, pp. 327-341, Maio. 2001.

[9] SCUCATO, J.B.S., RIBAS, L.G.S. "Rede Fisica: O Mapeamento das Escolas do Paraná". In: *Gisbrasil 2001*, mídia digital, Curitiba, 2001.

[10] http://www.cieg.ufpr.br/Producao/artigos/simiart.htm

[11] CARVALHO, F. "Planejando o Atendimento dos Alunos para a Rede Urbana de Ensino Fundamental". In: *Gisbrasil 2001*, mídia digital, Curitiba, 2001.

[12] FADEL, J.E., BERNARDO FILHO, O. "O Uso de Novas Tecnologias no Auxílio à Matrícula em Escolas da Rede Pública". In: *I Simpósio Ibero Americano de Cartografia para Criança*, pp. 66-67, Rio de Janeiro, Agosto. 2002.

[13] http://www.educacao.pe.gov.br/matricula/matricula.index

[14] FADEL, J.E., *Retratação e Análise da Distribuição Espacial de Alunos e Professores das Escolas do Bairro do Grajaú/RJ: Um Exemplo de Utilização de Técnicas de Geomática*. Bacharel em Geografia. Trabalho de Conclusão de Curso, Universidade Federal Fluminense, Niterói, RJ, Brasil, 2001.

[15] G. B. Korte, *The GIS Book*, Fourth Edition, OnWord Press, 1997.

[16] P. A. Burrough, *Principles of Geographical Information Systems for Land Resources Assessment*, Oxford University Press., 1986.

[17] D. Maguire, M. Goodchild e D. Rhind, *Geographical Information Systems: Principles and Applications*, New York, John Wiley and Sons, 1991.

[18] ELMASRI, R., NAVATHE, S. B., *Fundamentals of Database Systems*. Redwood City: Benjamin/Cummings Publishing Company, Inc., 1994.

[19] http://www.pmni.com.br

[20] AUTODESK, "AutoCad R14", USA, Mídia Digital.

[21] http://www.santuarioaccess.hpg.ig.com.br/index.html

[22] MICROSOFT, "Microsoft Access 2000", USA, Midia Digital.

[23] ENVIRONMENTAL SYSTEMS RESEARCH INSTITUTE INC., "ArcView®3.1", USA, 1996, Mídia Digital.

[24] SONNINO, B., *Desenvolvendo Aplicações com Delphi 5.0*. São Paulo, Makron Books, 2000.

[25] OLIVEIRA, A., MEDEIROS, M., *Delphi 5.0 – Conceitos Básicos*. Florianópolis, Advanced Books, 2000.

[26] BORLAND, "Delphi 5.0", USA, Midia Digital.

[27] http://www.geocities.com/SiliconValley/Foothills/9467/delphi.htm

[28] ENVIRONMENTAL SYSTEMS RESEARCH INSTITUTE INC., Map Objects 2.1, USA, 1999, Help On Line.

[29] http://www.esri.com/software/mapobjects/index.html