ROSÂNGELA SILVA

**BANCOS DE DADOS GEOGRÁFICOS:
UMA ANÁLISE DAS ARQUITETURAS DUAL (SPRING) E
INTEGRADA (ORACLE SPATIAL)**

Dissertação apresentada à
Escola Politécnica da
Universidade de São Paulo para
obtenção do título de Mestre em
Engenharia.

São Paulo

2002

ROSÂNGELA SILVA

**BANCOS DE DADOS GEOGRÁFICOS:
UMA ANÁLISE DAS ARQUITETURAS DUAL (SPRING) E
INTEGRADA (ORACLE SPATIAL)**

Dissertação apresentada à
Escola Politécnica da
Universidade de São Paulo para
obtenção do título de Mestre em
Engenharia.

Área de Concentração:
Engenharia de Transportes

Sub-área:
Informações Espaciais

Orientador:
José Alberto Quintanilha

São Paulo

2002

A todos aqueles que se interessam pelas Arquiteturas
de Sistemas Gerenciadores de Bancos de Dados Geográficos.

“Só estou onde estou porque acredito
em todas as possibilidades”

“O segredo dos que triunfam é saber
sempre recomeçar”

AGRADECIMENTOS

A Deus, pela oportunidade, pela saúde, inteligência, perseverança e pelas pessoas maravilhosas que colocou no meu caminho.

Aos meus queridos pais, que certamente sem a ajuda deles eu não estaria hoje aqui defendendo esta dissertação.

Ao meu orientador e amigo Prof. Dr. José Alberto Quintanilha a quem muito admiro, por me aceitar como orientanda, pelo respeito às minhas idéias e por me conduzir até o fim deste trabalho.

Ao Prof. Dr. Marcos Rodrigues, por ter me ensinado os primeiros passos e por me mostrar que “o importante neste processo é o caminhar”.

Ao Prof. Dr. João Argemiro C. Paiva, do Instituto Nacional de Pesquisas Espaciais (INPE), pelo imprescindível acompanhamento do trabalho e pelas orientações.

Ao Departamento de Engenharia de Transportes e à Diretoria da Escola Politécnica da USP pela compreensão e por terem viabilizado o desenvolvimento do trabalho.

Aos amigos Marco Augusto Alves Weber, Silvio Luís Rafaeli Neto, João Gilberto Ponciano Pereira, Tarcísio Neves da Cunha, Diana Sarita Hamburger, Rosângela Leal dos Santos, Arlete Ohata, Alejandro Diaz Cordero, Franz H. Pereyra Zamora, Solly Exman Kleingesinds, Eric Aubert, Mara Estela de Souza, Silvana Susko Marcellini, Deivenson Carvalho, Marcelo Antonio Nero e Olga Kazuko Iwai, pelas contribuições, correções, críticas e sugestões.

Ao amigo Jorge Augusto Carvalhinho Lopes, pela disponibilidade e pelo apoio no ORACLE SPATIAL.

Aos amigos Marcelo Marcos Barbosa e Bruno Franklin pela ajuda com a Linguagem PL/SQL.

Às amigas Silvana Amaral Kampel e Rosângela Leal dos Santos pela ajuda no SPRING.

Ao Prof. Dr. Witold Zmitrowicz e aos amigos Aristeu Zansaburo Nakamura, Clara Chuen Kai Ohtsuki, Neide Lima Farran e Clélia Mendonça de Moraes pela ajuda nos assuntos relativos ao Planejamento Urbano.

A Silvia Regina Saran Della Torre pelo apoio na geração da ficha catalográfica e a Maria de Fátima da Silva pelas correções nas Referências Bibliográficas.

Ao Prof. Edvaldo Simões da Fonseca Júnior, coordenador da informática do Departamento de Engenharia de Transportes, e aos colegas de trabalho Renato Alvarenga e Adalberto Moreira Mariano por me substituírem nos horários de suporte.

Ao Nilton Sérgio Capuano pelos muitos momentos em que esteve ao meu lado.

A todos os meus amigos pela força em continuar o trabalho, que por várias vezes me fizeram acreditar que era possível.

A todos os alunos de pós-graduação e usuários do Laboratório de Geoprocessamento pela compreensão e colaboração durante a minha ausência como funcionária.

A toda a minha família pelos conselhos e incentivos.

RESUMO

As características particulares dos dados geográficos constituem a razão pela qual se faz necessário estruturar novos tipos de dados e arquitetar novas formas de armazenamento e acesso aos dados. Este trabalho apresenta uma análise considerando as Arquiteturas Dual e Integrada em relação à forma de gerenciamento e recuperação da informação espacial, em conjunto com as informações não espaciais.

Este trabalho aborda os conceitos fundamentais acerca dos Sistemas Gerenciadores de Banco de Dados Geográficos. Para demonstrar como estes conceitos são importantes e influenciam diretamente na eficiência dos mesmos, conclui-se o trabalho com o desenvolvimento de alguns testes de funcionalidade sob duas ferramentas com arquiteturas distintas, são elas: o SPRING, de Arquitetura Dual, e o ORACLE SPATIAL, de Arquitetura Integrada. Os testes de funcionalidade objetivaram verificar se e como as ferramentas em estudo, suportam determinados tipos de consultas espaciais. Para tanto foi escolhido o cenário de Planejamento Urbano e selecionados alguns tipos de consultas envolvendo componentes espaciais, que normalmente são implementadas neste tipo de aplicação.

Os resultados obtidos permitem concluir, principalmente, que as ferramentas analisadas suportaram as consultas espaciais utilizadas nos testes - algumas envolvendo o objeto espacial e o atributo ao mesmo tempo - porém, com algumas restrições. Além disso, foi possível fazer algumas considerações em relação à utilização ou não de índices espaciais para otimização das consultas e algumas constatações sobre as arquiteturas de banco de dados geográficos analisadas, em relação à integração dos dados espaciais com os dados não espaciais.

ABSTRACT

The spatial data complexity justifies the need to develop new spatial data types and to design new structures to store, to query and to handle spatially referenced data inside a database management system (DBMS).

This work presents an analysis of these issues considering the different architectures of geographic databases. The Dual and Integrated architectures are considered in relation to spatial data and attribute handling.

The principal concepts concerning the spatial DBMS are presented. To demonstrate how these concepts are important and influence directly in the efficiency of Geographic Information Systems (GIS) tools, these work concludes with the development of some tests of functionality. Two GIS programs with distinct architectures were tested, and they are: the SPRING, of Dual Architecture, and ORACLE SPATIAL, of Integrated Architecture. The functionality tests aimed to verify if the tools in study support some kind of spatial queries, describing the necessary steps to perform these queries. In order to perform the tests, an Urban Planning application was chosen and some spatial queries were defined and executed.

SUMÁRIO

AGRADECIMENTOS

RESUMO

ABSTRACT

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO	15
1.1 Objetivo.....	16
2 CONCEITOS BÁSICOS	17
2.1 Conceitos de Banco de Dados	18
2.2 Modelos de Dados para SGBD	22
2.3 Arquiteturas de Bancos de Dados Geográficos	26
3 CARACTERÍSTICAS DE DADOS GEOGRÁFICOS	35
3.1 Modelos de Dados	35
3.2 Representação de Dados	38
3.3 Indexação Espacial.....	41
4 LINGUAGEM DE CONSULTA ESPACIAL	51
4.1 Característica do SQL	52
5 SISTEMAS PARA INFORMAÇÃO GEOGRÁFICA	64
5.1 SPRING.....	64
5.2 ORACLE SPATIAL	77
6 APLICAÇÃO PRÁTICA E TESTES EFETUADOS.....	91
6.1 Descrição dos Dados	91

6.2 Testes de Funcionalidade	92
7 CONCLUSÕES E CONSTATAÇÕES	125
REFERÊNCIAS BIBLIOGRÁFICAS	129
APÊNDICE A - DADOS DISPONÍVEIS	135

LISTA DE FIGURAS

Fig. 1 – Principais Componentes de um SGBD	21
Fig. 2 – Arquitetura Puramente Matricial e “Flat-files”	29
Fig. 3 – Arquitetura Dual: Mapa de municípios de São Paulo associado a uma tabela relacional com dados descritivos dos municípios	30
Fig. 4 – SGBD relacional acoplado à ferramenta SIG	31
Fig. 5 – SGBD relacional externo à ferramenta SIG, cuja conexão é feita utilizando-se um <i>software</i> que faz a interface entre os bancos de dados.	32
Fig. 6 – Tabela de endereçamento dos clientes, onde o campo “endereço” é a fonte para o georreferenciamento e o campo localização é do tipo “ponto”, ou seja, um tipo abstrato de dados (uma classe) pré-definida.	33
Fig. 7 – Representação do Modelo de Campos: Fotografia Aérea Vista do Município de São Paulo – SP	36
Fig. 8 – Representação do Modelo de Campos: Mapa Temático – Mapa de Uso do Solo obtido através de Classificação de uma Imagem do Satélite Landsat 7-Sensor TM (Vista do Município de São Bernardo do Campo - SP).....	37
Fig. 9 – Representação do Modelo de Objetos: Mapa de Municípios do Estado de São Paulo.....	37
Fig. 10 – Representação matricial: grade regular (a) e grade irregular (b)	38
Fig. 11 – Elementos da Estrutura Vetorial.....	39
Fig. 12 - Representação vetorial e matricial de um mapa temático	40
Fig. 13 – Decomposição do espaço em <i>Quad-tree</i>	44
Fig. 14 – Exemplo de uma referência de objetos no índice <i>Quad-tree</i>	45
Fig. 15 – Caminhamento na <i>Quad-tree</i> : Ordem proposta por Morton.....	45
Fig. 16 – Representação da <i>Kd-tree</i>	48
Fig. 17 – Decomposição <i>R-tree</i> (Árvore Balanceada)	49
Fig. 18 – Sintaxe do Comando <i>Select</i> em apenas uma tabela relacional: <i>‘Encontrar todos os estudantes com 18 anos’</i>	53
Fig. 19 – Sintaxe do Comando <i>Select</i> em duas tabelas relacionais correlacionadas: <i>‘Encontrar os nomes dos marinheiros que reservaram o barco 103’</i>	53
Fig. 20 – Geometrias suportadas pelo DB2.....	54

Fig. 21 – O Resultado de uma interseção: $A \cap B$ (a) pode ser dois polígonos (b); dois polígonos e uma linha (c); ou dois polígonos, uma linha e um ponto (d).....	58
Fig. 22 – Matriz de Interseção entre as propriedades Interior (I), Exterior (E) e Fronteira (F) de A e B.....	60
Fig. 23 – Exemplos de relacionamentos topológicos	61
Fig. 24 – Modelo Conceitual do SPRING.....	65
Fig. 25 – Arquitetura de Tabelas do SPRING.....	67
Fig. 26 – Exemplo de Associação de um objeto a mais de um PI.....	69
Fig. 27 – Exemplo de Associação de um PI a mais de um Objeto.....	69
Fig. 28 – Hierarquia de classes para representação vetorial.....	71
Fig. 29 – Interface de Consulta Espacial do SPRING.....	76
Fig. 30 – Modelo de Dados do ORACLE SPATIAL. O Estado do Hawaii é um exemplo de geometria que consiste de um conjunto de elementos (ilhas). ..	78
Fig. 31 – Tipos de Dados Espaciais Primitivos do ORACLE SPATIAL	79
Fig. 32 – Estrutura de uma tabela espacial no ORACLE SPATIAL	81
Fig. 33 – Instância de um polígono na tabela Distrito. Exemplo de consulta à coluna espacial (Resultado Parcial).....	81
Fig. 34 – Interface de Consulta Espacial do ORACLE SPATIAL. Exemplo de Consulta utilizando o relacionamento ANYINTERACT	90
Fig. 35 – Município de São Paulo: Distritos e Unidades de Saúde.....	93
Fig. 36 – Casos de Interseção entre ‘buffers’	102
Fig. 37 – Casos de Extrapolação dos ‘buffers’ em relação ao Limite do Município (Zona Leste).....	102
Fig. 38 – Município de São Paulo: Linhas do Metrô e Distritos	108
Fig. 39 – Seleção dos Distritos com população acima de 50.000	111
Fig. 40 – Seleção dos Distritos com população acima de 50.000 habitantes e que se interceptam com as linhas do Metrô.....	111
Fig. 41 – Município de São Paulo: Linhas do Metrô e Unidades de Saúde.....	113
Fig. 42 – “Buffer” de 2 Km gerado ao redor da Linha Vermelha do Metrô.....	114
Fig. 43 – Grade de distâncias (200x200m) gerada a partir da Linha Vermelha do Metrô.....	117
Fig. 44 – Imagem correspondente à grade de distâncias (200x200m)	117

Fig. 45 – Fatiamento da grade de distâncias (200x200m) do PI Numérico.....	118
Fig. 46 – Unidades de Saúde do Distrito 53.....	119
Fig. 47 – Unidades de Saúde do Distrito 53.....	122
Fig. 48 – Exemplo do Operador Direcional “Norte”	124

LISTA DE TABELAS

Tabela 1 – Característica das Arquiteturas Dual e Integrada.....	33
Tabela 2 - Operadores Espaciais e Funções Espaciais do ORACLE SPATIAL.....	82

1 INTRODUÇÃO

O presente trabalho é um estudo de dois tipos de Arquiteturas de Banco de Dados Geográficos, sobretudo no tocante ao armazenamento e manipulação de dados espaciais.

As primeiras ferramentas SIG (Sistemas de Informação Geográfica) desenvolvidos eram apenas uma informatização dos procedimentos para confecção de mapas. Segundo Reeve (2001), não tinham a capacidade de armazenar dados alfanuméricos, foram desenvolvidos para prover análises espaciais, visualização cartográfica e tecnologia de interface gráfica, não dando tanta importância em oferecer recursos de bancos de dados. Segundo Goodchild e Kemp (1990), muitos SIGs foram desenvolvidos para um ambiente de pesquisa pequenas bases de dados. Com o passar dos tempos, surgiu a necessidade cada vez maior de acrescentar características (atributos) aos dados espaciais. Nesta época, já existiam os Sistemas Gerenciadores de Banco de Dados (SGBD) convencionais. Segundo Zhou (2001), os sistemas de banco de dados tiveram origem no final da década de 50 e início da década de 60, em grande parte pelas pesquisas e desenvolvimentos da IBM Corporation. Muitos destes desenvolvimentos visavam atender às necessidades de negócios, dos militares, do governo e de instituições educacionais, que eram organizações complexas com dados complexos e com necessidades de informações. Mas as estruturas de dados utilizadas para armazenar dados espaciais não foram as mesmas adotadas pelos SGBDs convencionais. As distinções existentes entre dados espaciais e dados não espaciais, que, até então, eram considerados complexos para as estruturas de dados convencionais vigentes, implicaram na necessidade de concepções diferentes para processamento e armazenamento dos mesmos.

Houve um desenvolvimento significativo nas Arquiteturas das ferramentas SIG, que evoluíram de programas essencialmente de aplicações para verdadeiros sistemas gerenciadores de bancos de dados espaciais, incorporando capacidades de gerenciamento e recuperação da informação, e integração com bancos de dados convencionais.

Por outro lado, houve também uma evolução nos SGBDs convencionais (que antes se restringiam na geração de relatórios em forma de tabelas e gráficos), permitindo o armazenamento e a recuperação de dados espaciais, através de uma extensão

espacial.

Este trabalho apresenta uma análise considerando as Arquiteturas Dual e Integrada em relação à forma de gerenciamento e recuperação da informação espacial, em conjunto com as informações não espaciais. O conhecimento das arquiteturas das ferramentas SIG é essencial para se usufruir de todo o potencial das mesmas.

Dada a diversidade de assuntos envolvidos na questão do armazenamento e manipulação de dados espaciais, os Capítulos 2, 3 e 4 são introdutórios e apresentam os conceitos fundamentais em Bancos de Dados Geográficos. O quinto capítulo descreve o interior de duas ferramentas de mercado que suportam aplicações espaciais, são elas: o SPRING (do Instituto Nacional de Pesquisas Espaciais - INPE) e o ORACLE SPATIAL (da Oracle Corporation). Conclui-se o trabalho com o desenvolvimento de testes que demonstram como esses conceitos são importantes e influenciam diretamente na eficiência dos Sistemas Gerenciadores de Banco de Dados Geográficos.

1.1 Objetivo

O objetivo do trabalho é apresentar os conceitos fundamentais acerca dos Sistemas Gerenciadores de Banco de Dados Geográficos, que ainda são alvo de pesquisa e exploração, e verificar como esses conceitos estão implementados, na prática, em dois tipos de ferramentas com Arquiteturas distintas, uma Dual e outra Integrada.

A avaliação consiste basicamente em analisar os tipos de dados e as operações espaciais que cada uma oferece, os índices que são utilizados na otimização das consultas e as diferenças em relação às Linguagens de Consultas Espaciais. Este objetivo se materializa através da conversão de uma base de dados espacial para ambas as ferramentas, e da percepção das variantes nessas duas situações. As ferramentas propostas para esta análise são o SPRING, que possui Arquitetura Dual, e o ORACLE SPATIAL, que possui Arquitetura Integrada.

O estudo dessas ferramentas se justifica pelas diferenças entre as arquiteturas e porque estas são mais acessíveis por estarem disponíveis na Internet (O SPRING é totalmente gratuito, o ORACLE, está disponível uma versão de uso temporário).

2 CONCEITOS BÁSICOS

Neste capítulo serão apresentados os componentes-chave dos Bancos de Dados Geográficos, as Arquiteturas e Modelos de Dados sob os quais esses bancos são baseados.

Considerando-se que o objetivo de um Sistema Gerenciador de Banco de Dados Geográficos é armazenar, acessar e analisar rápida e eficientemente dados espaciais, entende-se por componentes-chave as estruturas que, de uma forma integrada, permitem o alcance desse objetivo. São componentes-chave: as Estruturas de Representação do(s) Dado(s) Espacial(is), a Linguagem de Consulta Espacial, as Técnicas de Processamento de Consulta Espacial e os Métodos de Acesso Espacial.

Os Bancos de Dados Geográficos precisam de estruturas para representar os dados espaciais, bem como as operações para manipular esses dados. Dada a natureza complexa¹ e multidimensional² dos dados espaciais, aliada ao fato dos Sistemas Gerenciadores de Banco de Dados convencionais - a grande maioria baseada no Modelo Relacional - não contemplarem os requisitos espaciais, novos tipos de dados (Tipos Abstratos de Dados³) e operadores⁴ foram construídos. Conseqüentemente, as Linguagens de Consulta, também sofreram e vêm sofrendo alterações para englobar esses novos tipos de dados e operadores. Do mesmo modo, novos métodos de acesso, também chamados de mecanismos de indexação espacial, foram implementados, através de estruturas para dados multidimensionais, tais como *Kd-trees*, *Quad-trees* e *R-trees*. Os Métodos de Acesso são o componente-chave para garantir a eficiência na recuperação dos dados no banco de dados, principalmente no caso dos bancos de dados espaciais que geralmente armazenam grandes volumes de

---

¹ Objetos complexos são agregações de objetos simples. Uma rede elétrica é um exemplo de objeto complexo constituído de pelo menos dois objetos simples como os postes e os cabos de transmissão.

² O termo "multidimensional", segundo InterSystems (2002), significa que os dados podem ser indexados por tantos parâmetros quantos forem necessários: eles não estão limitados a linhas e colunas. Enquanto um banco de dados relacional pode ser considerado bidimensional, uma base multidimensional considera cada atributo dos dados (como produtos, região geográfica e período) como uma dimensão separada. Foram feitos alguns estudos nos últimos anos e ferramentas foram desenvolvidas, tais como a OLAP (OnLine Analytical Processing), para incluir a dimensão "tempo" nas aplicações convencionais. Em relação às aplicações SIG, o foco é incluir a dimensão "espaço".

³ Tipos Abstratos de Dados (TAD) combinam dados com operações, onde a implementação dessas operações é encapsulada. Segundo Egenhofer (1995), TAD's são estruturas que definem por completo o comportamento dos objetos, ou seja, a semântica dos objetos e de suas operações.

⁴ Os operadores representam os relacionamentos entre os objetos espaciais, que podem ser, basicamente, topológicos ou métricos.

dados. O Processamento da Consulta Espacial consiste na interpretação (sintática e semântica) de uma consulta escrita em linguagem de alto nível para instruções de baixo nível, gerando-se um plano estratégico de execução e diretrizes para o processamento.

Além das considerações anteriores, os Sistemas de Bancos de Dados Geográficos adotam tipos específicos de Arquitetura e Modelos de Dados. Arquiteturas podem ser vistas como mecanismos de estruturação da informação (espacial e não espacial), e são baseadas em Modelos de Dados, os quais implementam a estrutura lógica das Arquiteturas. As Arquiteturas podem ser de dois tipos: Dual e Integrada, que diferem no modo como os dados espaciais são associados aos atributos descritivos que os caracterizam, imprescindíveis na maioria das aplicações. Os Modelos de Dados abordados são os Modelos Relacional, Orientado a Objeto e Objeto-Relacional.

Os conceitos citados acima se inter-relacionam, de modo que Arquiteturas implementam Modelos de Dados que, por sua vez, especificam o tipo de Linguagem de Consulta a ser utilizado. Por exemplo, a Arquitetura Integrada implementa o Modelo Objeto-Relacional; e a SQL (Structure Query Language) é específica para o Modelo Relacional.

2.1 Conceitos de Banco de Dados

O termo “banco de dados” significa um conjunto de dados organizados de modo a atender uma determinada finalidade, ou um conjunto de finalidades integradas. O termo “banco de dados espaciais” é utilizado quando os dados a serem armazenados possuem características espaciais, ou seja, possuem propriedades que descrevem a sua localização no espaço e a sua forma de representação. Seja, por exemplo, o caso de uma companhia de energia elétrica que necessita manter um cadastro de toda a rede elétrica, com seus vários elementos constituintes, tais como: transformadores, capacitores, cabos, postes, etc. Além disso, é importante para a concessionária saber quem são seus clientes, como se comporta o consumo desses clientes e a que parte da rede eles estão conectados. A concepção de um banco de dados que suporte aplicações de gestão da rede elétrica, envolvendo questões do tipo: “avisar os clientes que ficarão sem energia durante a troca do transformador T” e “qual a necessidade de ampliação da rede nos próximos 5 anos” demandam conhecimentos de quais clientes

estão conectados ao transformador, onde estão esses clientes e onde está o transformador a ser trocado; e informações sobre consumos de energia, respectivamente.

A organização do banco de dados envolve, basicamente, um mecanismo eficiente de armazenamento e manipulação de dados, cujo gerenciamento é controlado pelo Sistema Gerenciador de Banco de Dados (SGBD). Um SGBD é um conjunto de softwares que gerencia a estrutura do banco de dados e controla o acesso aos dados armazenados no mesmo. A meta básica de um SGBD é proporcionar um ambiente conveniente e eficiente para armazenamento e recuperação da informação.

Entende-se por armazenamento a definição de estruturas de dados para organização e representação dos dados. Normalmente os bancos de dados exigem muita memória e por isso são armazenados em memória secundária, usualmente discos, e transferidos para a memória principal quando necessário. A velocidade de acesso ao disco é bem mais baixa quando comparado à memória principal, logo, os dados são estruturados de modo a minimizar a necessidade de movimentação entre disco e memória. Desse modo, um SGBD geralmente mantém índices para os dados, que são estruturas de métodos de acesso que aceleram (otimizam) a recuperação dos dados, diminuindo os acessos ao disco.

No caso particular dos bancos de dados espaciais, questões referentes ao armazenamento da representação (geometria) dos dados espaciais, bem como dos relacionamentos entre eles (especialmente os topológicos) são relevantes e carecem de estruturas de dados específicas. A geometria diz respeito à forma geométrica utilizada para representar o dado espacial e está intimamente ligada à escala de trabalho e ao objetivo da aplicação. Relacionamentos topológicos são aqueles que se mantêm inalterados após distorções⁵ no mapa, tais como a conectividade e a adjacência. No exemplo sobre a concessionária de energia elétrica, citado anteriormente, suponha que a escala de trabalho seja 1:2000 e que o objetivo seja a gestão da rede elétrica. Os postes, capacitores e transformadores podem ser, simplificadaamente, representados como pontos, enquanto os cabos podem ser representados como linhas. Os elementos da rede (postes, transformadores e capacitores) são topologicamente conectados aos cabos, constituindo assim a rede

---

⁵ As distorções são do tipo: mudanças de escala, rotação e translação.

elétrica.

Entende-se por manipulação do banco de dados, operações do tipo: definição do esquema do banco de dados (projeto geral do banco), recuperação da informação, atualização, inserção e exclusão de dados.

A Figura 1 mostra o esquema típico de um Sistema Gerenciador de Banco de Dados. Na parte inferior, o elemento cilíndrico representa o banco de dados estático, tal como ele é fisicamente armazenado no disco. Observa-se que juntamente com os dados estão os metadados⁶, isto é, um conjunto de dados que contém informação sobre a estruturas dos dados, também chamado de dicionário de dados ou catálogo do sistema. O dicionário de dados é gerado quando da especificação do esquema do banco de dados, através da Linguagem de Definição de Dados (DDL). A definição do esquema do banco de dados envolve especificar os tipos de dados (o tipo e o formato de armazenamento de cada item), a estrutura de cada arquivo, os relacionamentos e as restrições impostas aos dados a serem armazenados no banco de dados⁷. O catálogo é usado pelo SGBD e também pelos usuários do banco que precisam de informação sobre a estrutura do banco de dados. Em bancos de dados orientados a objetos e objeto-relacionais, operações sobre os dados podem ser definidas como parte da definição do banco de dados.

O Gerenciador de Armazenamento faz a interface entre os dados armazenados e o Processador de Consultas, capturando do disco o(s) bloco(s) de dados necessários para atender aos requisitos de uma consulta (por exemplo) e gerenciando esses dados nos *buffers* de memória principal.

O Processador de Consultas, também chamado de Gerenciador de Consultas, interpreta a Linguagem de Manipulação de Dados (DML) utilizada pelo banco de dados, gerando um plano estratégico de execução com instruções de baixo nível para que o Gerenciador de Armazenamento possa efetuar as operações necessárias. A DML é uma linguagem que viabiliza o acesso ou a manipulação (recuperação,

---

⁶ No caso de Bancos de Dados Geográficos, o termo “metadados” tem um sentido mais amplo, uma vez que possui informações também a respeito da base cartográfica, tais como fonte de origem, data de aquisição, finalidade para a qual foi gerada, dentre outras. Guidara Jr. (2000) define metadados como a descrição dos dados em termos de conteúdo, de qualidade, de condição e de outras características próprias dos dados, que podem também ser simplesmente definidos como informações sobre dados.

inserção, remoção ou modificação) do banco de dados. O acesso ao banco de dados necessita de informações de quais dados são necessários e como obtê-los. Algumas DMLs são mais amigáveis ao usuário, exigindo apenas que se especifique quais são os dados necessários, deixando o “como obtê-los” sob responsabilidade dos SGBDs. Desse modo, esses tipos de SGBDs demandam técnicas de otimização para recuperar eficientemente as informações.

Fig. 1 – Principais Componentes de um SGBD

Fonte: Adaptado de Ullman e Widom (1997)

Uma consulta é um caso especial de manipulação do banco de dados e corresponde a uma solicitação de recuperação de informações. A parte da DML responsável pela recuperação de informações é chamada Linguagem de Consulta. Tradicionalmente a Linguagem SQL (Structured Query Language) foi adotada como Linguagem de Consulta padrão para os sistemas de banco de dados baseados no Modelo Relacional, mas posteriormente, com o surgimento dos Modelos Orientado a Objetos e Relacional Extendido, a SQL vem recebendo alguns incrementos para tratar os novos

⁷ Por exemplo, no Modelo Entidade Relacionamento, são definidas as entidades, os atributos, os relacionamentos e as restrições ou limitações (como, por exemplo, as regras de cardinalidade) que o

tipos de dados, tais como os espaciais e os de multimídia.

O Processador de Consultas, é responsável por processar e otimizar consultas em linguagem de alto-nível. Uma consulta expressa em linguagem de alto-nível (tal como a SQL) é interpretada em três passos pelo Processador de Consultas. No primeiro passo, as palavras-chaves, nomes de atributos e nome de relacionamentos são identificados no texto da consulta. No segundo passo, verifica-se a sintaxe da consulta para determinar se ela foi formulada de acordo com as regras de sintaxe (regras da gramática) da linguagem de consulta. Finalmente, no terceiro passo, a consulta deve ser validada, verificando-se se todos os nomes de atributos e relacionamentos e se os nomes têm significado semântico no esquema do banco de dados em particular sendo consultado. Uma representação interna para a consulta deve então ser criada. O SGBD deve definir um plano estratégico de execução para recuperar o resultado da consulta e passar o controle e as devidas instruções ao Gerenciador de Armazenamento. Uma consulta geralmente apresenta várias possibilidades estratégicas de execução e o processo de se escolher a mais adequada para executar uma consulta é conhecido como otimização de consulta, tarefa esta, considerada mais importante do Processador de Consultas.

Apesar do nome, o Processador de Consultas é responsável tanto por processar as consultas como as requisições de modificação nos dados e nos metadados.

Além de armazenamento e manipulação da informação, o SGBD precisa proporcionar segurança ao armazenamento de informações, diante das falhas do sistema ou acesso não autorizado. O Gerenciador de Transação⁸ é responsável por garantir a integridade do sistema. Ele assegura que várias consultas sendo executadas simultaneamente não interfiram umas com as outras e que o sistema não perderá dados mesmo em caso de falhas.

2.2 Modelos de Dados para SGBD

Apesar dos SGBDs convencionais compartilharem a mesma arquitetura proposta pelo ANSI/SPARC Study Group on Data Base Management Systems, cujos detalhes podem ser encontrados em Elmasri e Navathe (2000), há um número distinto de

---

banco de dados deve garantir.

famílias de banco de dados, com abordagens diferentes para estruturação dos dados, baseadas nos seguintes modelos⁹: Modelo Relacional e os modelos chamados pós-relacionais; tais como o Relacional Extendido e o Orientado a Objetos.

A seguir estes modelos são brevemente apresentados, incluindo considerações quanto a sua aplicação a dados espaciais.

2.2.1 Modelo Relacional

O Modelo Relacional usa um conjunto de tabelas para representar tanto os dados como as relações entre eles. Uma linha em uma tabela representa um relacionamento entre um conjunto de valores. Uma vez que essa tabela é um conjunto de relacionamentos, há uma estreita correspondência entre o conceito de tabela e o conceito matemático de relação. Foi desta correspondência que surgiu o nome do Modelo Relacional. A descrição completa do Modelo Relacional pode ser encontrada em Codd (1990).

Em linguagem comum, sentenças como “x é irmão de y” e “x é primo de y” são denominadas relações entre x e y. Em linguagem matemática, as sentenças anteriores são sentenças abertas de duas variáveis, ou seja, são afirmações que não sabemos se são verdadeiras ou falsas; elas se tornam verdadeiras ou falsas quando atribuímos valores a x e a y. Define-se relação de um conjunto A num conjunto B como todo subconjunto não vazio de $A \times B$. R é relação de A em B $\Leftrightarrow R \subset A \times B$, $R \neq \emptyset$. Por exemplo, Consideremos $A = \{2;3;5\}$, $B = \{1;4;6\}$ $R = \{ (x;y) \in A \times B \mid x > y \}$, que é subconjunto de $A \times B$.

Em relação a dados espaciais, a tecnologia relacional não é muito eficiente para manipular dados espaciais. Segundo Reeve (2001), devido à diferença existente entre realidade e representação oriunda deste modelo, os objetos espaciais, para serem representados, precisam ser quebrados em várias tabelas (por exemplo, um polígono é um conjunto de segmentos, que por sua vez é um conjunto de pontos – para se representar um polígono são necessárias três tabelas no modelo relacional: polígono, ponto e segmento). Além disso, dada a existência de relacionamentos topológicos,

---

⁸ Uma transação é uma seqüência de operações de manipulação no banco de dados. Através deste conceito é possível implementar a proteção contra falha e o acesso concorrente.

⁹ Neste trabalho, não serão consideradas as Arquiteturas baseados nos Modelos Hierárquico e Em Rede.

uma simples alteração em uma feição espacial requer alteração não apenas nos registros que descrevem esta feição, mas também de todas as feições adjacentes a ele (efeito em cascata), o que requer atualização simultânea em um grande número de registros em múltiplas tabelas. Mesmo assim, como observado por Woodsford (2001), a tecnologia relacional, apesar de tudo, foi a pioneira na implementação da topologia.

Outro problema apresentado pelo Modelo Relacional está associado ao tamanho fixo e à ordem dos registros. Segundo Reeve (2001), o Modelo Relacional requer que todos os registros em uma tabela sejam de tamanho fixo e não há nenhuma restrição quanto à ordem em que as linhas são armazenadas em uma tabela. Por outro lado, os elementos geográficos em um mapa são de diferentes tamanhos e a ordem das coordenadas que descrevem esses elementos é significativa. Por exemplo, a forma de uma linha é determinada pela ordem das coordenadas que a descrevem.

2.2.2 Modelo Relacional Estendido

O Modelo Relacional Estendido¹⁰ é uma mistura do Modelo Relacional com o Modelo Orientado a Objetos (apresentado no item 2.2.3), e é também chamado de Objeto-Relacional. Esta abordagem teve origem no conceito de “servidores universais”. Segundo Filgueiras (2000), *“a idéia básica de servidores universais é a busca de uma solução única para problemas de gerenciamento de banco de dados, estendendo o modelo relacional para o objeto-relacional. A intenção é permitir que os desenvolvedores continuem em seu já familiar ambiente de desenvolvimento, baseado em estruturas tabulares e com linguagens de definição (DDL) e manipulação (DML) de dados bem definidas, porém sendo capazes de explorar as potencialidades das tecnologias orientadas a objeto.”*

O Modelo Relacional Estendido veio atender, basicamente, à necessidade de criação de novos tipos de dados, até então não contemplada no Modelo Relacional. Assim, o Modelo Relacional Estendido permite a integração de dados convencionais com outros tipos de dados. O modelo puramente Relacional só permitia tipos de dados simples, tais como: inteiros, reais, *strings*, datas, entre outros. A abordagem objeto-

---

¹⁰ É interessante observar que são uma evolução dos SGBD's relacionais líderes de mercado, tais como o Oracle e DB2.

relacional, da mesma forma que os SGBDs Orientados a Objeto, suporta objetos complexos, além dos simples.

A primeira abordagem foi permitir campos de tamanho variável, através da criação de um tipo adicional de dados chamado “bulk field” (campo longo), provendo um armazenamento dinâmico ao invés de uma estrutura fixa. Posteriormente, através do conceito de sistemas abertos¹¹ e da orientação a objetos, foi possível suportar objetos complexos, como é o caso dos dados espaciais e multimídia.

2.2.3 Modelo Orientado a Objetos

O Modelo Orientado a Objetos utiliza todos os conceitos oriundos da Orientação a Objetos - por exemplo classes, objetos, herança¹² e polimorfismo - que podem ser encontrados detalhadamente em Rumbaugh et al. (1994).

Sucintamente, um objeto é constituído por um conjunto de atributos e um conjunto de métodos. Os atributos definem as características do objeto e os métodos definem o comportamento do objeto. No exemplo da concessionária de energia elétrica, por exemplo, para um objeto do tipo “poste”, uma característica pode ser “Localização” (par de coordenadas X,Y) e um método pode ser “DeterminarPosição”, uma função que retorna a localização do poste, ou seja, os valores assumidos para X e Y. Observe que as características dos objetos somente são acessadas através dos métodos, desse modo, encapsulando os detalhes de implementação do objeto. Objetos com características e comportamento semelhantes são agrupados em classes¹³.

Além do forte esquema de tipificação apresentado anteriormente, o foco fundamental dos SGBDs Orientados a Objetos, segundo Reeve (2001), é arquivar um alto nível de abstração, ou seja, o modo como as entidades e eventos são

---

¹¹ Sistemas abertos permitem aos desenvolvedores adaptar o software adicionando novos tipos de dados (Tipos Abstratos de Dados - ADT). ADT's combinam dados com operações, onde a implementação dessas operações é encapsulada. Seja, por exemplo, um tipo “polígono” com a operação “PolígonoArea”, que determina a área desse polígono. Definições de ADT's são dirigidas para aplicações, ou seja, os ADT's são definidos de acordo com as necessidades das aplicações, a partir de outros ADT's considerados básicos e que fazem parte da ferramenta utilizada.

¹² Segundo Woodsford (1995), a abordagem orientada a objetos permite modelar os objetos complexos e hierárquicos através do conceito de herança.

¹³ Segundo Egenhofer (1995), a palavra objeto é utilizada para uma simples ocorrência (instância) de dados descrevendo alguma coisa que tenha individualidade e algum comportamento observável. Classificação é o mapeamento de vários objetos (instâncias) em uma classe comum.

representados no banco de dados, está mais próximo de como as coisas são e de como as pessoas pensam sobre elas no mundo real. Assim, os SGBDs Orientados a Objetos são um modelo de representação mais próximo da realidade.

As seguintes considerações diferem os SGBDs Orientados a Objetos das gerações anteriores:

- a abordagem orientada a objetos, diferentemente da abordagem relacional que utiliza várias tabelas para representar uma entidade do mundo real, agrupa todos os dados e operações que descrevem esta entidade;
- nos SGBDs Orientados a Objeto, cada instância de um objeto tem uma identificação diferente, permitindo que objetos diferentes tenham os mesmos dados e as mesmas funções, o que não ocorria nos relacionais;
- os processos das empresas saem da aplicação e passam a ser armazenados também no banco de dados, através dos métodos que representam o comportamento dos objetos (encapsulamento de regras de negócio). Isto facilita a manutenção do sistema, uma vez que os processos estão mais organizados e as alterações ocorrem nos objetos e não nas aplicações, e também facilita a reutilização de código; e
- o mapeamento do Modelo Lógico para o Modelo Físico passa a ser automático. Antes, a modelagem de aplicações era orientada a objetos mas a representação era relacional.

Segundo Filgueiras (2000), os SGBDs Orientados a Objetos apresentam um problema de escalabilidade, pois *“quando são submetidos a condições em que grupos de usuários vão além da ordem de dezenas, ou em que dados ultrapassam a ordem de 10 GB, as vantagens no tratamento dos dados complexos tendem a desaparecer. Os anos de investimento em pesquisas pelos vendedores de SGBD Relacionais também os têm deixado à frente no tocante à segurança, integridade e disponibilidade”*. Segundo Araújo Jr (2000), a abordagem puramente orientada a objetos traz problemas de controle de concorrência e transações.

2.3 Arquiteturas de Bancos de Dados Geográficos

A existência de diferentes arquiteturas de banco de dados geográficos, se deve à

necessidade de integração entre dados convencionais e dados espaciais¹⁴. Essa integração é importante, senão fundamental, porque permite a análise conjunta de vários tipos de informações e onde elas ocorrem no espaço.

A integração de dados espaciais com dados convencionais foi uma preocupação posterior ao desenvolvimento das ferramentas SIG e dos SGBDs convencionais. As ferramentas SIG surgiram inicialmente com a finalidade de processar mapas¹⁵. Segundo Reeve (2001), não tinham a capacidade de armazenar dados alfanuméricos, foram desenvolvidos para prover análises espaciais, visualização cartográfica e tecnologia de interface gráfica, não dando tanta importância em oferecer recursos de bancos de dados. Por outro lado, as aplicações envolvendo dados convencionais, se restringiam a relatórios em forma de tabelas e gráficos. Logo, a necessidade de integração impulsionou o desenvolvimento de novos mecanismos computacionais de conexão entre esses dois tipos de dados.

O estudo de alternativas para integração foi necessário devido à natureza dos dados espaciais, que, até então, eram considerados complexos¹⁶ para as estruturas de dados convencionais vigentes, baseadas no Modelo Relacional. Como primeira alternativa, as ferramentas SIG, que inicialmente só armazenavam dados espaciais, passaram a incorporar SGBDs relacionais em suas arquiteturas, através de uma Arquitetura Dual, na qual o SGBD relacional é utilizado somente para armazenar atributos, enquanto que os dados geográficos são armazenados utilizando-se as estruturas internas da ferramenta SIG. Esta abordagem apresentou resultados satisfatórios, sobretudo com a melhora do *hardware* e do *software* e para aplicações *stand-alone*. Mas, à medida que foram surgindo as aplicações distribuídas e a necessidade por segurança (proteção contra falhas), integridade, consistência, capacidade para armazenar grandes bases de dados e controle de acesso aos dados, questões até então não atendidas pelas ferramentas SIG, mas consideradas em estado avançado em SGBDs convencionais, mais uma vez foi necessário repensar as arquiteturas

---

¹⁴ Dada uma entidade do mundo real, dados convencionais, também chamados de atributos, dados descritivos ou alfanuméricos, são aqueles que descrevem as características dessa entidade, independentes de quaisquer propriedades espaciais. Uma vez que essa entidade é colocada no espaço, os dados espaciais tais como forma, localização e relacionamentos com outras entidades, descrevem as propriedades desta entidade neste espaço.

¹⁵ Mais detalhes sobre a evolução das ferramentas SIG podem ser encontradas em Tomlinson (1988), UW-Pathbreakers (1996) e Laurini e Thompson (1992).

existentes. A segunda opção foi levar os dados espaciais para os SGBDs convencionais, o que ficou conhecido como Arquitetura Integrada, na qual ambos os tipos de dados (espacial e alfanumérico) são armazenados no mesmo banco de dados.

A seguir, essas abordagens e suas implicações são apresentadas em mais detalhes. É importante lembrar que a evolução das arquiteturas vai de encontro às necessidades do mercado e este exige flexibilidade de acordo com os requisitos das aplicações.

2.3.1 Arquitetura das primeiras ferramentas SIG

As primeiras ferramentas SIG desenvolvidas eram puramente matriciais¹⁷, logo não permitiam uma separação entre dados locacionais e atributos. O modelo adotado dividia o espaço em uma grade regular em que cada célula armazenava o valor do atributo correspondente à sua localização. Entretanto, essa abordagem não propôs uma solução adequada para muitas aplicações que precisavam armazenar grandes volumes de dados (atributos) ou manipular atributos independentemente dos dados espaciais. Além disso, para cada variável existia um mapa correspondente, gerando redundância (de mapas).

A abordagem *flat-file* foi adotada em seguida como uma forma de armazenar atributos pelas ferramentas SIG. A vantagem desta abordagem, comparada com a abordagem puramente matricial, é a eliminação da redundância, ou seja, o mapa (a descrição geográfica do mapa) passou a ser armazenado apenas uma vez. *Flat-files* podem ser imaginados como tabelas individuais do Modelo Relacional, sem a propriedade de relacionamento entre as mesmas.

A Figura 2 ilustra ambas as estruturas citadas anteriormente.

---

¹⁶ Mais detalhes sobre a complexidade dos dados espaciais podem ser encontrados em Laurini e Thompson (1992) e Egenhofer (1995).

¹⁷ O termo “matricial” se refere a um tipo de decomposição regular do espaço, através de um modelo que utiliza uma grade regular (também chamado de *raster*) onde as células constituintes (também chamadas de *pixels*) são definidas por polígonos de tamanho fixo.

Fig. 2 – Arquitetura Puramente Matricial e “Flat-files”

Fonte: Adaptado de Reeve (2001)

O *software* Idrisi (Eastman, 1999), na versão para Windows, desenvolvido pela Clark University, é um exemplo de um *software* que utiliza o conceito “flat-file”, para armazenamento de atributos, utilizando um subconjunto limitado do SGBD relacional Access (da Microsoft) para suportar manipulações SQL em arquivos com atributos. Entretanto, o Idrisi para Windows somente pode lidar com uma tabela simples de atributos por vez, de modo que não deve ser utilizado o termo “banco de dados”. Além disso, com este sistema, há uma descontinuidade entre dados espaciais e bancos de dados externos, os quais têm de ser conectados, em repetitivos processos de importação e exportação. As tabelas têm de ser extraídas do banco de dados externo, convertidas para o formato do Idrisi e depois importadas para dentro do Idrisi. Isso pode gerar inconsistências, uma vez que as tabelas são duplicadas.

2.3.2 Arquitetura Dual

Os sistemas duais podem ser de dois tipos: sistemas no qual o SGBD relacional é parte da ferramenta SIG, ou são externos, e a comunicação se dá através de *links* ou vínculos. Em ambos os casos, o SGBD relacional é utilizado somente para armazenar atributos, enquanto que os dados espaciais são armazenados utilizando-se as estruturas internas da ferramenta SIG, daí o nome “Arquitetura Dual”. Como observado em Câmara (1995), um identificador comum (geo-id) liga os componentes geométrico e convencional do objeto geográfico, como mostra a Figura 3.

A Figura 4 ilustra o caso em que o SGBD relacional está embutido na ferramenta SIG. Nesta abordagem, a integridade pode ser comprometida, caso ocorra algum

problema com a conexão entre a parte espacial e a parte não espacial - como por exemplo uma falha no sistema quando efetuando uma atualização em ambos os bancos de dados - a menos que a ferramenta SIG tenha recursos de segurança contra falhas e volte o sistema à situação imediatamente anterior à falha. Além disso, uma vez que os dados espaciais estão separados dos convencionais, para responder a uma simples consulta, dois processos devem ser disparados (o que demanda conjuntos de comandos separados para dados espaciais e dados não espaciais) e a resposta então é composta.

Fig. 3 – Arquitetura Dual: Mapa de municípios de São Paulo associado a uma tabela relacional com dados descritivos dos municípios

Fonte: Programa de Transferência de Tecnologia: Base de Dados (1998)

Fig. 4 – SGBD relacional acoplado à ferramenta SIG

Fonte: Adaptado de Reeve (2001)

A segunda opção, ou seja, quando os SGBDs relacionais são externos, garante maior flexibilidade ao usuário, uma vez que é permitido escolher o SGBD relacional a ser utilizado. A Figura 5 ilustra essa abordagem.

O *software* Arc-Info (ESRI, 1994), desenvolvido pela ESRI (Environmental Systems Research Institute), é um exemplo de arquitetura dual que possui tanto um SGBD relacional embutido, como fornece a possibilidade de conexão com bancos de dados externos. No primeiro caso, Info é um banco de dados relacional (pertencente à ferramenta SIG) e Arc é um *software* escrito pela ESRI para armazenar e manipular dados espaciais.

O *software* SPRING (INPE, 2002), desenvolvido pelo INPE (Instituto Nacional de Pesquisas Espaciais), também é outro exemplo de software que implementa a Arquitetura Dual.

Fig. 5 – SGBD relacional externo à ferramenta SIG, cuja conexão é feita utilizando-se um *software* que faz a interface entre os bancos de dados.

Fonte: Adaptado de Reeve (2001)

2.3.3 Arquitetura Integrada

A Arquitetura Integrada utiliza o Modelo Relacional Extendido. Inicialmente, através dos campos longos, foi possível armazenar campos contendo um número variável de pares de coordenadas necessários para descrever cada segmento. Essa abordagem eliminou as desvantagens da abordagem puramente relacional, apresentadas no item anterior. Posteriormente, com o desenvolvimento dos SGBDs Objeto-Relacionais, com a abertura dos sistemas de banco de dados para incorporação de novos tipos de dados, objetos complexos, como os dados espaciais, puderam ser incorporados aos bancos de dados convencionais.

A abordagem integrada assegura uma forte ligação entre dados espaciais e dados não espaciais (ver Figura 6), reduzindo as dificuldades em manter a integridade. Além disso, os dados espaciais podem se beneficiar das facilidades de segurança, *backup* e recuperação que os SGBDs relacionais oferecem. A integração também ocorre no âmbito das estruturas de comando e processamento, isto é, a consulta à base de dados integrada envolve a recuperação de ambos os tipos de dados.

No entanto, esta abordagem continua utilizando as ferramentas SIG na visualização dos dados espaciais e dos resultados das análises espaciais.

Segundo Filgueiras (2000), a IBM e a Oracle, que desenvolveram o Módulo Espacial para as versões mais atuais para os SGBDs DB2 e Oracle, respectivamente, apostam

no conceito de “banco de dados universal”.

Tabela: Clientes

Fig. 6 – Tabela de endereçamento dos clientes, onde o campo “endereço” é a fonte para o georreferenciamento e o campo localização é do tipo “ponto”, ou seja, um tipo abstrato de dados (uma classe) pré-definida.

2.3.4 Arquitetura Dual x Arquitetura Integrada

A Tabela 1 apresenta um resumo das principais características das Arquiteturas Dual e Integrada.

Tabela 1 – Característica das Arquiteturas Dual e Integrada

ARQUITETURA DUAL (SPRING)	ARQUITETURA INTEGRADA (ORACLE SPATIAL)
– Comunicação dos dados espaciais com os dados não espaciais através de vínculos (identificadores comuns); – Implementa o Modelo Relacional para dados não espaciais e o Modelo Orientado a Objetos para dados geográficos; – SQL (Structured Query Language) interno para acesso a dados não espaciais; – LEGAL (Linguagem Espacial para Geoprocessamento Algébrico) para acesso a dados espaciais. LEGAL é uma linguagem orientada a objetos, baseada no padrão ODMG (Object Data Management Group), onde uma expressão de busca é construída a partir da linguagem OQL (Object Query Language); – Devido à dualidade, podem ocorrer problemas de integridade do banco de dados, em caso de falhas no processamento do dado espacial e não atualização do dado não espacial, e vice-versa. Além disso, no processamento das consultas envolvendo ambos os tipos de dados, dois processos são disparados e o resultado então é composto.	– Forte ligação entre dados espaciais e não espaciais; – Implementa o Modelo Objeto-relacional; – SQL Espacial, uma extensão da Linguagem SQL; – Herda todos os recursos de bancos de dados, tais como segurança no acesso ao dado, <i>backup</i>, recuperação da informação, proteção contra falhas, dentre outras; – A consulta à base de dados envolve recuperação de ambos os tipos de dados e um único processo é disparado, aumentando a eficiência da consulta e evitando problemas como a perda de integridade do banco de dados.

3 CARACTERÍSTICAS DE DADOS GEOGRÁFICOS

As características particulares dos dados geográficos é a razão pela qual se faz necessário estruturar novos tipos de dados e arquitetar novas formas de armazenamento e acesso aos dados.

Como observado em Tutorial do SPRING (INPE, 2000), “os fenômenos do mundo real podem ser representados por diversos tipos de dados, que são: dados temáticos (por exemplo: tipos de solo), dados cadastrais (por exemplo: cadastro urbano e rural), dados de redes (por exemplo: rede de esgoto e logradouros), dados de modelos numéricos (por exemplo: dados geofísicos e topográficos) e dados do tipo imagens (por exemplo: foto aérea e de satélite).”

Neste capítulo serão apresentados os conceitos relacionados na modelagem dessa diversidade de tipos de dados para um ambiente SIG.

3.1 Modelos de Dados

Gomes e Velho (1995) distinguem quatro universos a serem considerados no processo de Modelagem de Dados: universo do mundo real, universo conceitual, universo de representação e universo de implementação. O universo do mundo real inclui as entidades da realidade que serão modeladas no sistema. O universo conceitual inclui uma definição matemática (formal) das entidades que serão incluídas no modelo. No universo de representação as diversas entidades formais são mapeadas para as representações gráficas (geométricas) utilizadas no modelo. O universo de implementação é onde as estruturas de dados e algoritmos para as operações sobre os dados (geográficos) são escolhidas e o sistema é codificado. Mais detalhes sobre o paradigma dos quatro universos pode ser encontrado também em Câmara (1995).

No nível conceitual, de acordo com Frank e Goodchild (1990), a realidade geográfica pode ser percebida segundo duas visões: a visão de campos e a visão de objetos. Na visão de campos, segundo o Manual do SPRING (INPE, 2002), o espaço geográfico é visto como uma superfície contínua onde a cada ponto é associado um ou vários valores de atributos, definido como uma função contínua de x em y . Por exemplo (INPE, 2002), um mapa de vegetação descreve uma distribuição que associa a cada

ponto do mapa, um tipo específico de cobertura vegetal, enquanto um mapa geoquímico associa o teor de um mineral a cada ponto. Outros exemplos de variáveis que assumem esse comportamento são: altitude (cada ponto da superfície terrestre tem um valor para a altitude), precipitação, temperatura, poluição, etc. Na visão de objetos, a realidade é vista como uma superfície ocupada por uma coleção de entidades¹⁸ distintas e identificáveis e cada posição (X,Y) do espaço poderá estar ou não ocupada. Por exemplo, no cadastro espacial da rede elétrica de um município identifica-se cada elemento da rede como um dado individual, com atributos que o distinguem dos demais.

A Figura 7 e 8 ilustram exemplos de representações no Modelo de Campos e a Figura 9 ilustra um exemplo de representação no Modelo de Objetos. Na Figura 8, uma classe temática é representada por um conjunto finito de *pixels* que juntos caracterizam uma determinada região da imagem. Já na Figura 9, cada município é representado por apenas um objeto espacial, distinto e identificável.

Fig. 7 – Representação do Modelo de Campos: Fotografia Aérea Vista do Município de São Paulo – SP

Fonte: Programa de Transferência de Tecnologia: Material do Curso de Arcview (1998)

¹⁸ Rigaux, Scholl e Voisard (2002) definem entidade (ou feição) como sinônimo para objeto geográfico. Um objeto geográfico é composto de identidade (identificação), objeto espacial (forma e localização do objeto) e descrições. Ao conjunto de objetos geográficos, com descrições e forma comuns, denomina-se tema. Por exemplo, o conjunto de postes da rede elétrica formam o tema “postes”, onde os objetos possuem a mesma forma (ponto) e o mesmo conjunto de atributos descritivos, porém identidades diferentes.

Fig. 8 – Representação do Modelo de Campos: Mapa Temático – Mapa de Uso do Solo obtido através de Classificação de uma Imagem do Satélite Landsat 7- Sensor TM (Vista do Município de São Bernardo do Campo - SP)

Fonte: Quintanilha et alii. (2000)

Fig. 9 – Representação do Modelo de Objetos: Mapa de Municípios do Estado de São Paulo.

Fonte: Programa de Transferência de Tecnologia: Carta-imagem (1998)

3.2 Representação de Dados

No nível de representação, define-se a representação geométrica para os dados (objetos) espaciais. Existem basicamente dois modos de representação¹⁹: matricial (ou tesselação) e vetorial. A representação matricial é uma decomposição (regular ou irregular) finita do plano em células disjuntas e discretas. A Figura 10 ilustra a representação matricial decomposta em uma grade regular (a) e em uma grade irregular (b). Quando a decomposição é regular, o modelo utiliza uma grade regular (também chamado de *raster*) onde as células constituintes (também chamadas de *pixels*) são definidas por polígonos de tamanho fixo. As imagens de satélite oriundas do Sensoriamento Remoto Orbital, bem como as fotografias aéreas digitais oriundas de aerolevantamentos são um exemplo típico de representação por grade regular. Ao contrário, na decomposição irregular, as células são de tamanho variável. Um mapa de zonas de uso do solo é um exemplo para o caso de grade irregular, conforme Figura 10 (b).

Fig. 10 – Representação matricial: grade regular (a) e grade irregular (b)

Fonte: Adaptado de Rigaux, Scholl e Voisard (2002)

Na estrutura vetorial, os objetos espaciais podem ser representados por:

- Pontos: são objetos de dimensão zero, representados por um par de coordenadas (x,y). São utilizados para representar entidades cuja forma não é significativa, tais como os clientes da rede elétrica, ou os postes.
- Objetos lineares: são objetos de uma dimensão, representados por uma *polyline*²⁰. São usualmente utilizados para representar redes, tais como:

¹⁹ Rigaux, Scholl e Voisard (2002) apresentam uma terceira opção de representação denominada “half-plane”, mas que não será abordada neste trabalho.

²⁰ Uma *polyline* pode ser definida como um conjunto de segmentos de linhas conectadas por vértices, onde cada vértice é compartilhado por exatamente dois segmentos de linha, exceto nos extremos da *polyline* (*end-points*). Existem *polyline* fechadas (quando os dois pontos extremos se encontram) que são chamadas de anel.

rodovias, hidrografia, rede elétrica (cabos), etc.

- Objetos com superfície: são objetos de duas dimensões, representados por polígonos²¹. São utilizados para representar entidades cujas áreas são importantes, tais como regiões administrativas, parcelas em mapas de uso do solo, represas, etc. No entanto, objetos de duas dimensões não são necessariamente conectados, definindo assim uma região (conjunto de polígonos), como por exemplo a representação do conjunto de ilhas que formam o Japão.

A Figura 11 ilustra os elementos da estrutura vetorial.

Fig. 11 – Elementos da Estrutura Vetorial

Fonte: Extraído de Tutorial do SPRING (INPE, 2000)

Observa-se que os conceitos de campos e objetos podem ser ambos mapeados para a representação matricial e vetorial. Seja por exemplo a representação de um mapa temático (modelo de campos). Como apresentado na Figura 12, este mapa pode ser tanto representado na estrutura matricial, resultante de uma classificação de uma imagem de satélite por exemplo, ou na estrutura vetorial, através de classes representadas por polígonos. Do mesmo modo, um ponto na representação matricial corresponde a um simples *pixel* e um polígono é representado por um conjunto finito de *pixels*. O *software* Idrisi (da Clark University) é um exemplo de ferramenta SIG que utiliza a representação matricial tanto para mapear campos quanto objetos.

²¹ Um polígono é uma região do plano delimitada por um anel, que constitui a sua borda ou fronteira.

Fig. 12 - Representação vetorial e matricial de um mapa temático

Fonte: Extraído de Tutorial do SPRING (INPE, 2000)

Ainda no universo de representação, é importante tratar os relacionamentos espaciais (relacionamentos entre os objetos espaciais)²² que podem ser armazenados explicitamente ou não no banco de dados. Segundo Rigaux, Scholl e Voisard (2002), existem três tipos de representação que são utilizadas para uma coleção de objetos espaciais: Modelo *Spaghetti*, Modelo de Redes e Modelo Topológico. Esses modelos de representação diferem, principalmente, no modo como representam a topologia.

No Modelo *Spaghetti*, a geometria dos objetos espaciais é representada independentemente e nenhuma topologia é armazenada, de modo que há redundância de representação quando os elementos são adjacentes. Por exemplo, num mapa de regiões administrativas, os segmentos de linha que são comuns a duas regiões são representados duas vezes no banco de dados. Os relacionamentos topológicos são computados sob demanda. Neste modelo, estão presentes os seguintes objetos básicos: ponto, *polyline*, polígono (definido como uma lista de pontos) e região (definida como um conjunto de polígonos).

O Modelo de Redes²³ é normalmente utilizado para representar redes, tal como a rede elétrica do exemplo utilizado no início deste capítulo. Os relacionamentos topológicos (conectividade) entre pontos e *polylines* são armazenados através do conceito de “arcos” e “nós” (topologia Arco-nó). Um arco é uma *polyline* que começa em um nó e termina em outro nó. Um nó é um ponto que conecta uma lista de arcos. As redes podem ser classificadas em planares ou não planares. Nas redes planares, cada intersecção de *polylines* é armazenada como um nó, mesmo que esse

²² Estamos considerando que as entidades a serem representadas pertencem ao modelo de objetos e a representação é vetorial.

²³ Baseado no conceito de Grafos.

nó não corresponda a um objeto geográfico. Em uma rede não planar, arcos podem cruzar sem produzir uma intersecção. Neste modelo, não há representação topológica entre objetos 2D.

O Modelo Topológico é semelhante ao Modelo de Redes, com exceção que a rede é planar. Este modelo representa a topologia Arco-nó (conectividade) e Arco-polígono (adjacência, vizinhança, etc.), onde um arco comum a dois polígonos é representado apenas uma vez (reduzindo o espaço em disco utilizado para armazenamento). Neste modelo, estão presentes os seguintes objetos básicos: ponto, nó, arco, polígono (representado por uma lista de arcos) e região (definida como um ou mais polígonos adjacentes). Comparado ao Modelo *Spaghetti*, perde na simplicidade das estruturas de armazenamento e na facilidade de inserção de novos objetos (necessita um rearranjo da estrutura dos dados). Por outro lado, facilita as consultas topológicas e a atualização no banco de dados, uma vez que os arcos adjacentes são armazenados apenas uma vez, minimizando problemas de inconsistência.

3.3 Indexação Espacial

Índices são estruturas de dados desenvolvidas para otimizar o acesso aos bancos de dados que, normalmente, estão armazenados em memória secundária. Como visto no Capítulo 1, o acesso à memória secundária é bem mais lento comparado à memória principal. No caso dos bancos de dados convencionais, os índices aceleram o processamento das consultas, através da utilização de estruturas de dados dinâmicas, tais como *Tabelas Hash* e *B-trees* (Árvores Balanceadas)²⁴, tendo como chave um ou vários campos²⁵. No caso de bancos de dados espaciais, esses índices não se aplicam, devido à natureza multidimensional dos dados espaciais. Os índices espaciais levam em consideração a localização dos objetos no espaço. Os índices espaciais processam consultas espaciais eficientemente, através da redução do conjunto de dados a ser verificado no processamento.

As consultas espaciais, como observado em Oracle Spatial User's Guide and Reference (2002), envolvem questões do tipo “encontrar objetos que interagem com

²⁴ Mais detalhes sobre *Tabelas Hash* e *B-tree* pode ser encontrado em Elmasri e Navathe (2000).

²⁵ Segundo Rigaux, Scholl e Voisard (2002), todas as propriedades do índice *B-tree* que permitem eficiência são devidas à existência de uma ordem total dos valores das chaves do índice e que não é facilmente adaptável aos dados espaciais.

um dado ponto ou com uma área de interesse” (chamadas *query window*); e “encontrar pares de objetos que interagem espacialmente entre si” (*spatial join*). Nas consultas do tipo *query window*, o usuário especifica uma área ou janela de interesse e o SGBD espacial deve recuperar todos os dados que interceptam ou estão contidos nesta área específica. *Spatial joins* são consultas do tipo “mostre todos os clientes (da rede elétrica) do bairro B”. Normalmente, as consultas espaciais envolvem um teste espacial, no qual verifica-se um predicado espacial, do tipo “dois objetos se interceptam?” ou “dois objetos têm algum tipo de relacionamento espacial específico?”.

No caso de aplicações SIG, processar uma consulta espacial lida com operações geométricas computacionalmente custosas e complexas, e o volume dos dados armazenados no disco geralmente é grande. O processamento de uma consulta envolvendo um predicado espacial é realizado em dois passos, conforme descrito em Rigaux, Scholl e Voisard (2002). No primeiro passo, chamado “filtragem”, um índice espacial é utilizado para selecionar os objetos que satisfazem o predicado espacial. No segundo passo, chamado “refinamento”, somente o conjunto resultante da filtragem é analisado sequencialmente e o teste espacial (que é o mais custoso) é feito nas geometrias reais dos objetos, diminuindo assim, o custo de complexidade computacional para avaliar os predicados espaciais.

Segundo Egenhofer et al. (1999), uma grande variedade de métodos de acesso para dados espaciais foi desenvolvida, cada um buscando ocupar menos espaço em disco ou ser mais eficiente que o outro, sendo difícil analisar qual método é o melhor. As diferenças entre eles são pequenas, mas alguns fatores tais como a variação na distribuição e densidade dos objetos no espaço²⁶, o tipo de geometria do objeto no *layer*²⁷, dados fortemente correlacionados e o número de atualizações no banco de dados, certamente influenciam no desempenho do índice. Ainda segundo Egenhofer et al. (1999), as ferramentas SIG ou os SGBDs com extensões espaciais, têm optado por métodos de acesso que são mais fáceis de compreender e implementar,

²⁶ Rigaux, Scholl e Voisard (2002) classificam os índices como *space-driven structures* e *data-driven structures*. O primeiro caso, ao contrário do segundo, não leva em consideração a distribuição dos objetos no espaço. *Quad-tree* e *Grid-file* são exemplos de *space-driven structures* e *R-tree* é um exemplo de *data-driven structure*.

²⁷ *Layer*, segundo Oracle Spatial User's Guide and Reference (2001), consiste de geometrias que compartilham um conjunto de atributos comuns.

parecendo que a preocupação com o desempenho fica em segundo plano, devido à pequena diferença entre os métodos. Algumas ferramentas, tal como o ORACLE SPATIAL, oferecem mais de uma opção de índice (*Quad-tree* e *R-tree*), de modo que o projetista do banco de dados possa escolher qual índice utilizar em uma determinada consulta, inclusive proporcionando a opção de se criar índices diferentes para a mesma geometria. Outras ferramentas, como o SPRING por exemplo, pré-determinam o tipo de índice a ser utilizado de acordo com a representação da geometria utilizada. No SPRING, conforme Manual do SPRING (INPE, 2002) e observado em Câmara (1995), uma imagem temática (como um mapa de vegetação por exemplo) utiliza um índice *Quad-tree*; e um grafo orientado (por exemplo representando uma rede elétrica) utiliza um índice *R-tree*.

Os índices são constituídos, basicamente, das seguintes estruturas:

- Estrutura para representar o espaço: é uma estrutura conceitual de decomposição do espaço, buscando associar os objetos com as suas localizações neste espaço. Cada índice utiliza um método diferente de decomposição do espaço, por exemplo no *Grid-file*, o espaço é decomposto em um *grid* de células retangulares; o *Quad-tree* é baseado na decomposição sucessiva do espaço em *tiles* (seções); e o *R-tree* é baseado numa representação aproximada dos objetos através do mínimo retângulo envolvente (MRE).
- Estrutura de implementação: o modelo conceitual do espaço apresentado acima é implementado através de uma estrutura de dados conhecida - normalmente são utilizados *arrays* ou árvores - que relaciona o dado espacial (armazenado no disco) com a sua localização neste modelo do espaço.

De uma forma geral, no processamento de uma consulta nestes índices, por exemplo numa *query window*, primeiro verifica-se que parte do espaço a *window* se sobrepõe. O espaço, neste caso depende da estrutura utilizada para representar o espaço. Por exemplo, no *Grid-file*, verifica-se quais células do espaço são interceptadas pela *window*; no *Quad-tree*, verifica-se quais seções interceptam a *window*; e no *R-tree*, quais MREs estão envolvidos na consulta. Num segundo passo, a estrutura de dados que implementa o índice é então percorrida, de modo a encontrar os objetos espaciais que estão relacionados a essa porção do espaço. O índice retorna, então, o conjunto

de identificadores desses objetos espaciais.

A seguir serão apresentados alguns exemplos de métodos de acesso para dados espaciais. Dada a diversidade de métodos, buscou-se aqueles que são os mais utilizados pelas ferramentas SIG e também pelos SGBDs convencionais com extensão espacial. São eles: *Quad-tree*, *Kd-tree* e *R-tree*. Esses métodos serão analisados em relação ao Modelo Conceitual do Espaço adotado, ou seja, como o espaço é estruturado (dividido), qual o critério de parada para a divisão do espaço, quais as vantagens e desvantagens de cada método e para que tipo de geometria (ponto, linha ou polígono) é mais indicado.

3.3.1 Quad-tree

a) Modelo Conceitual do Espaço

Uma *Quad-tree*, segundo Oracle Spatial Training Guide (ORACLE, 2001), é um mecanismo de codificação para determinar múltiplas chaves, ou seções, para cada geometria indexada, como mostra a Figura 13. Seções podem ser imaginados como retângulos que definem uma cobertura exclusiva (nenhuma seção se sobrepõe a outra) e exaustiva (as seções cobrem completamente o objeto) de cada geometria armazenada no *layer*.

Fig. 13 – Decomposição do espaço em *Quad-tree*

Fonte: Extraído de Oracle Spatial Training Guide (ORACLE, 2001)

Na estrutura *Quad-tree*, o processo de decomposição do espaço é conhecido como tesselação, o qual determina que seções cobrem uma dada geometria. Na tesselação, o espaço de coordenadas é decomposto de uma maneira regular e hierárquica. No primeiro nível de decomposição, o espaço é dividido em quatro seções. Cada seção que interage com a geometria que está sendo decomposta, é também sub-dividida em

outras quatro seções e assim sucessivamente até que se alcance o limite estipulado de divisão. Os resultados da tesselação, são então armazenados no índice, uma árvore quaternária cujos nodos, segundo McGill Project 32 (1999), ou são folhas (nodos sem filhos) ou têm quatro filhos ordenados (ver Figura 14) e a cada folha pode ser associada uma página do disco. As seções podem ser rapidamente examinadas para verificar alguma interação entre os *layers*.

Fig. 14 – Exemplo de uma referência de objetos no índice *Quad-tree*

Fonte: Adaptado de Rigaux, Scholl e Voisard (2002)

As seções são linearmente ordenadas de modo a preservar a proximidade espacial. Existem alguns algoritmos para ordenação tais como as curvas de Peano, Hilbert e Morton. A Figura 15 mostra a curva de Morton que é utilizada na ordenação dos *Quad-trees* no ORACLE SPATIAL, e que também direcionam o caminharmento na árvore.

Fig. 15 – Caminhamento na *Quad-tree*: Ordem proposta por Morton

Fonte: Extraído de Oracle Spatial User's Guide and Reference (ORACLE, 2001)

b) Critérios de Parada para a Construção do Índice

Segundo McGill Project 32 (1999), geralmente o limite imposto de subdivisão da *Quad-tree* está ligado com a capacidade de armazenamento disponível e o tempo de processamento requerido. Segundo Rigaux, Scholl e Voisard (2002), a divisão é feita até que o número de seções de cada quadrante seja menor que a capacidade de uma página de memória. Segundo Oracle Spatial User's Guide and Reference (2001), o limite pode estar relacionado com a resolução (da seção), ou seja, pode-se determinar um tamanho fixo de resolução da seção como único fator de controle, assim, a tesselação termina quando o espaço de coordenadas pode ser decomposto em um número específico de vezes, tendo cada seção o mesmo tamanho. Ainda segundo Oracle Spatial User's Guide and Reference (2001), índices espaciais utilizando seções de tamanho fixo facilitam a comparação entre duas geometrias, que interagem quando compartilham uma ou mais seções.

c) Vantagens e Desvantagens

A eficiência deste método, segundo Oracle Spatial User's Guide and Reference (2001), depende do número de níveis determinado e da variação dos tamanhos das geometrias dos *layers*. Quando os mesmos tamanhos de seções são utilizados para cobrir geometrias de tamanhos pequeno e grande, um grande número de seções serão necessários no segundo caso. Entretanto, se o tamanho da seção for muito grande, essas seções não representarão geometrias pequenas muito bem. A escolha do tamanho da seção traz consequências diretas nos resultados das consultas espaciais.

Segundo Daniel Abugov em Oracle Spatial Discussion Forum (Oracle, 2002), o uso do índice *Quad-tree* é indicado para casos em que ocorrem atualizações constantes no banco de dados, pois o índice continua utilizável por bastante tempo, sem a necessidade de reconstrução, mesmo havendo atualizações.

Uma desvantagem desse método é que, para geometrias do tipo polígono ou linha, a duplicação das referências dos objetos ocorre em vários níveis, uma vez que esses objetos podem ocupar mais de uma seção no espaço.

d) Tipo de Geometria Associado

A princípio este índice pode ser utilizado para indexar qualquer tipo de geometria,

considerando a desvantagem apontada no item anterior para geometrias do tipo linha e polígono. Este índice é também utilizado para indexar imagens (Ver Samet (1989)).

Segundo Daniel Abugov em Oracle Spatial Discussion Forum (ORACLE, 2002), para dados pontuais, se a velocidade de atualização do índice for absolutamente crítica para o sucesso da aplicação, deve-se utilizar o índice *Quad-tree*.

3.3.2 *Kd-tree*

a) Modelo Conceitual do Espaço

No *Kd-tree*, o espaço é dividido em setores cujas formas dependem da ordem em que os pontos são adicionados. A Figura 16 mostra um exemplo de *Kd-tree*, onde as divisões foram geradas a partir da seguinte sequência de dados: Chicago, Mobile, Toronto, Buffalo, Denver, Omaha, Atlanta e Miami. A Figura 16 mostra a criação de setores a partir dessa sequência, considerando as características do *Kd-tree*, ou seja, a partir de cada dado inserido, é gerada uma divisão dos setores. Iniciando-se com as coordenadas de Chicago, é feita uma divisão vertical a partir da sua coordenada X (gerando-se os lados direito e esquerdo). Seguindo a sequência de cidades, considera-se Mobile (lado direito de Chicago) como ponto de referência para a nova divisão. Essa divisão será com uma linha horizontal (gerando lados superior e inferior) que possa “tocar” a divisão anterior feita por Chicago. A continuação seria Toronto, segundo a sequência, mas devido a que já foi utilizado o lado direito com Mobile, a regra deverá ser utilizar o lado esquerdo de Chicago antes de usar alguma outra divisão. Assim, só Denver cumpre as condições do *kd-tree*, pois está no lado não preenchido e está na sequência de cidades. Denver e Mobile seriam os dois “filhos” de Chicago. O mesmo mecanismo deve ser utilizado para obter os filhos de Denver e Mobile.

Assim, num caso de duas dimensões são comparadas as dimensões X e Y alternativamente e onde cada nó tem dois filhos. Isto quer dizer que para que uma nova ramificação seja criada, tem-se de considerar os eixos alternativamente, criando-se divisões de “esquerda-direita” ou “acima-embaixo”. Não podem ser criadas duas divisões seguidas num lado esquerdo por exemplo.

Fig. 16 – Representação da *Kd-tree*

Fonte: Extraído de Samet (1989)

b) Critérios de Parada para a Construção do Índice

O critério de parada corresponde ao número máximo de pontos da partição do espaço.

c) Vantagens e Desvantagens

A representação da *Kd-tree*, segundo Samet (1989), é uma melhoria do *Quad-tree* que evita que as ramificações geradas sejam demasiadas.

d) Tipo de Geometria Associado

O *Kd-tree* pode ser utilizado para representar dados de pontos em várias dimensões.

3.3.3 *R-tree*

a) Modelo Conceitual do Espaço

O índice *R-tree* é construído utilizando-se apenas uma simples aproximação da geometria do objeto espacial, ou seja o mínimo retângulo envolvente (MRE) à geometria do objeto (no caso de objetos lineares e com superfície). Para um *layer*, um índice *R-tree* consiste numa estrutura hierárquica dos MREs da geometria no *layer*, como mostra Figura 17.

Fig. 17 – Decomposição *R-tree* (Árvore Balanceada)

Fonte: Extraído de Oracle Spatial Training Guide (ORACLE, 2001)

Segundo Rigaux, Scholl e Voisard (2002), diferente dos métodos apresentados anteriormente, na estrutura do *R-tree* a referência de um determinado objeto aparece em apenas um nodo folha. Entretanto, os retângulos associados com os nodos internos, não podem ser considerados como partições exatas do espaço, uma vez que pode haver sobreposição desses retângulos.

b) Critérios de Parada para a Construção do Índice

O critério de parada depende do número de geometrias no *layer*.

c) Vantagens e Desvantagens

Segundo Oracle Spatial User's Guide and Reference (2001), o índice *R-tree*, em comparação com o índice *Quad-tree* requer menos capacidade de armazenamento, são fáceis de serem criados, são mais rápidos em consultas que levam em consideração o vizinho mais próximo, mas não é uma boa escolha se houver grande

quantidade de atualizações nos dados espaciais, o que não afeta a performance do *Quad-tree*.

Segundo Daniel Abugov em Oracle Spatial Discussion Forum (Oracle, 2002), no ORACLE SPATIAL, para uma utilização completa das funcionalidades do sistema, o índice *R-tree* é o mais indicado se os dados envolvidos na consulta forem geodésicos²⁸, pois este índice garante uma resposta mais exata quando se leva em consideração a curvatura da Terra.

d) Tipo de Geometria Associado

O índice *R-tree*, em adição ao *Quad-tree*, pode ser utilizado para indexação de pontos 3D e 4D, como nos problemas críticos de exploração de óleo, arquitetura, engenharia e aplicações científicas. No caso de objetos 3D a aproximação é em relação ao mínimo volume envolvente (MVE), mas serão considerados neste trabalho apenas objetos 2D.

Segundo Daniel Abugov em Oracle Spatial Discussion Forum (Oracle, 2002), no ORACLE SPATIAL²⁹, para dados pontuais, a atualização do índice será sempre mais lenta, em relação aos outros tipos de dados.

²⁸ Dado geodésico são dados consistindo de coordenadas que são definidas em relação a uma particular representação da figura da Terra. Se os dados forem geodésicos e for utilizado o índice *Quad-tree*, algumas funcionalidades não estarão disponíveis, tais como, o uso do parâmetro UNIT para o SDO_WITHIN_DISTANCE e SDO_NN_DISTANCE, bem como o uso do SDO_BATCH_SIZE associado com o SDO_NN.

²⁹ O SPRING não utiliza o índice *R-tree* para dados pontuais.

4 LINGUAGEM DE CONSULTA ESPACIAL

A Linguagem de Consulta³⁰, direta ou indiretamente através das aplicações, é o canal através do qual o usuário se comunica com o banco de dados, seja na fase de projeto (definição do esquema, visões, permissões de acesso), seja na fase de manipulação (consultas, atualizações, etc). As Linguagens de Consulta provêem uma interface de alto nível³¹ através da qual o usuário apenas especifica qual é o resultado esperado, deixando o “como obtê-lo” para o SGBD. Em outras palavras, as Linguagens de Consulta abstraem (encapsulam) detalhes de implementação do banco de dados.

Sabemos que existem SGBDs baseados nos Modelos Relacional, Orientado a Objetos e Objeto-Relacional, como exposto no Capítulo 2. Cada tipo de SGBD necessita uma Linguagem de Consulta específica padrão, uma vez que diferem na forma em que estruturam os dados. A padronização das Linguagens de Consulta para um tipo particular de SGBD³², como observado por Elmasri e Navathe (2000), é importante pois garante a portabilidade e a interoperabilidade das aplicações. A portabilidade significa a capacidade de uma determinada aplicação ser executada em diferentes SGBDs, sem a necessidade de grandes alterações. Se a aplicação for desenvolvida utilizando-se os construtores da linguagem padrão, é possível converter a aplicação (sem muitos investimentos) para diferentes produtos, desde que esses produtos também suportem essa linguagem padrão. A interoperabilidade significa uma mesma aplicação poder acessar dados em diferentes SGBDs (do mesmo tipo ou de tipos diferentes).

A SQL (Structured Query Language) é a Linguagem de Consulta padrão para os SGBDs baseados no Modelo Relacional. Similarmente, o ODMG³³ (Object Data Management Group) é um padrão para SGBDs baseados no Modelo Orientado a

---

³⁰ O termo “Linguagens de Consulta” engloba tanto a DDL como a DML, apesar da palavra “Consulta”.

³¹ Interface de alto-nível significa que a linguagem de consulta tem uma sintaxe mais amigável ao usuário.

³² É importante lembrar que, apesar da padronização da Linguagem de Consulta para determinada família de SGBD, cada ferramenta implementa o padrão de uma forma diferente. A Linguagem de Consulta é apenas uma linguagem e não um programa.

³³ O padrão ODMG consiste das seguintes partes: a ODL (Object Definition Language, semelhante à DDL); a OQL (Object Query Language, semelhante à DML); um Modelo de Objetos, através do qual a ODL e a OQL são baseadas; e as especificações para integração com as Linguagens de Programação C++, Smalltalk e Java. Mais informações sobre o padrão ODMG pode ser encontrado em <http://www.odmg.org>.

Objetos; e a SQL³⁴ é uma extensão da SQL para suportar os SGBDs baseados no Modelo Objeto-Relacional.

Em relação às ferramentas SIG ou em relação aos SGBDs que suportam dados espaciais, as Linguagens de Consulta embutidas são relativas às arquiteturas que essas ferramentas adotam. Por exemplo, o ORACLE é um SGBD convencional que, nas versões mais atuais (ORACLE SPATIAL), suporta dados espaciais integrados aos dados convencionais (o que é conhecido como Arquitetura Integrada), através do Modelo Objeto-Relacional. Por outro lado, o SPRING é uma ferramenta SIG que tem uma Arquitetura Dual, ou seja, os dados convencionais são gerenciados por um SGBD Relacional enquanto que os dados espaciais são gerenciados por um sistema que controla arquivos proprietários. Desse modo, o Oracle Spatial deve embutir a SQL3 e o Spring a OQL.

4.1 Característica do SQL

A SQL inclui algumas características da Álgebra Relacional, mas é baseada, principalmente, no Cálculo Relacional de Tupla³⁵. A Álgebra Relacional consiste em um conjunto de operações tendo como entrada uma ou duas relações e produzindo, como resultado, uma nova relação. As operações são divididas em dois grupos: operações sobre conjuntos (união, interseção, diferença e produto cartesiano) e operações para bancos de dados relacionais (*select*, *project*, *join*, etc). O Cálculo Relacional de Tupla, ao contrário da Álgebra Relacional, é não procedural, ou seja, os usuários não precisam especificar os procedimentos para obter os resultados. O Cálculo Relacional de Tupla é uma Linguagem de Consulta formal declarativa, baseada num ramo da lógica matemática chamado Cálculo de Predicado. Um predicado é uma declaração que avalia uma expressão e provê uma resposta verdadeira ou falsa baseada numa condição. Mais detalhes sobre a Álgebra Relacional e o Cálculo Relacional de Tupla podem ser obtidos em Silberschatz, Korth e Sudarshan (1999), e Elmasri e Navathe (2000). As Figuras 18 e 19 mostram exemplos de utilização do comando *Select* da Linguagem SQL.

---

³⁴ A evolução da SQL seguiu a seguinte sequência de atualizações: SQL86, SQL89, SQL92 (ou SQL2) e SQL3. A SQL92 é a versão da SQL mais atual para banco de dados puramente relacionais. A SQL3 é uma extensão para bancos de dados objeto-relacionais.

A SQL suporta definição de dados, consultas e atualizações. Além disso, facilita definições de visões (tabelas virtuais) no banco de dados, especificações de segurança e autorizações, definições de restrições de integridade e controles de transações. Ainda é possível embutir declarações SQL em Linguagens de Programação de propósito geral, tais como C, C++ Pascal e Java.

```
SELECT *
FROM Estudantes E
WHERE E.IdadeEst = 18
```

CodEst	NomeEst	LoginEst	IdadeEst
23421	Lucas	lucas@ca.com	18
24519	Clarissa	clarrissa@mt.com	18
21953	Otávio	otavio@mg.com	18

Fig. 18 – Sintaxe do Comando *Select* em apenas uma tabela relacional: “*Encontrar todos os estudantes com 18 anos*”

```
SELECT M.NomeMar
FROM Marinheiros M
WHERE EXISTS (SELECT R.CodRes
              FROM Reservas R
              WHERE R.CodBarco = 103 AND M.CodMar = R.CodMar)
```

Fig. 19 – Sintaxe do Comando *Select* em duas tabelas relacionais correlacionadas: “*Encontrar os nomes dos marinheiros que reservaram o barco 103*”

4.2 Operadores Espaciais

Para que os sistemas de Arquitetura Dual, bem como os sistemas integrados, pudessem suportar dados espaciais, foi necessário propor um conjunto de tipos de dados espaciais de propósito geral, e operações para manipular esses dados. Os elementos básicos desse conjunto, como citado no capítulo anterior, são: objetos pontuais, objetos lineares e objetos com superfície. A Figura 20 ilustra um conjunto de tipos de dados espaciais básico, utilizado pelo DB2 da IBM. A partir desses elementos primitivos, novos tipos abstratos de dados podem ser construídos pelo usuário, de modo a atender aos requisitos de uma determinada aplicação.

³⁵ Na terminologia formal do Modelo Relacional, uma linha de uma tabela é chamada de tupla, o cabeçalho de uma coluna da tabela é chamado de atributo e a tabela é chamada de relação.

Fig. 20 – Geometrias suportadas pelo DB2. Os tipos de geometrias cujos nomes estão nos retângulos brancos são instanciáveis, no desenho inferior estão expressas como símbolos visuais.

Fonte: Extraído de DB2 User's Guide (IBM, 2001).

As operações, por sua vez, segundo Rigaux, Scholl e Voisard (2002), podem ser classificadas em:

- Operações unárias³⁶ com resultado *booleano*: testam se o objeto espacial atende a determinada propriedade, por exemplo, dado um objeto do tipo polígono, testar se este polígono é convexo³⁷;
- Operações unárias com resultado escalar: as operações mais comuns desta categoria são para determinar o comprimento, área e perímetro dos objetos;
- Operações unárias com resultado espacial: são operações de transformação no objeto, de variadas naturezas, tais como: transformações topológicas; operações que transformam objetos de uma dimensão em objetos de outra dimensão; e outras operações como determinar o mínimo retângulo envolvente, determinar o centróide e *buffer*³⁸;
- Operações binárias com resultado booleano: são também chamadas de predicados espaciais e são a base para as consultas espaciais. Os predicados espaciais podem ser topológicos (verificar se está contido, se contém, se é disjunto, se intercepta, se é adjacente), métricos (testar se a distância entre dois objetos é menor que um determinado valor, por exemplo) ou direcionais (acima de ou norte de);
- Operações binárias com resultado escalar: uma operação típica desta classe é a distância entre dois objetos; e
- Operações binárias com resultado espacial: essas operações são utilizadas quando se quer gerar um novo objeto espacial. A maioria dessas operações são operações em conjuntos, tais como interseção – que é a base para a operação de *overlay*³⁹ - união e diferença.

As operações são implementadas através de funções. As funções, como colocado em DB2 User's Guide (IBM, 2001), podem ser classificadas em funções do tipo

---

³⁶ As operações unárias estão associadas ao tipo de geometria e cada geometria deve ter um conjunto de operações unárias pré-definido, suportada pelo SGBD. Por exemplo, a função para determinação do centróide só faz sentido para geometrias 2D (polígono e derivados, considerando o espaço bidimensional), enquanto que a função de determinação do comprimento é definida em geometrias 1D (*lines and polylines*).

³⁷ Um polígono convexo P é tal que para todo par de pontos A e B em P, o seguimento AB é completamente incluso em P.

³⁸ As operações de *buffer* geram entornos ao redor do objeto espacial, de determinados raios de distância.

³⁹ A operação de *overlay* compara dois objetos de mesma dimensão e retorna um terceiro objeto, correspondente à interseção desses dois objetos, de mesma dimensão dos objetos testados.

Calcular, Comparar, Converter e Transformar. Ou ainda, podem ser divididas nos seguintes grupos:

- Funções que retornam informações sobre as propriedades dos objetos espaciais;
- Funções que operam sobre objetos espaciais;
- Funções que mostram tipos de relacionamentos e fazem comparações entre objetos espaciais;
- Funções que geram objetos espaciais; e
- Funções que convertem objetos espaciais de um tipo de geometria em outro tipo⁴⁰.

As funções espaciais são utilizadas nas Linguagens de Consulta de modo que as consultas espaciais possam ser realizadas. O ORACLE SPATIAL, como observado em Oracle Spatial Training Guide (ORACLE, 2001) e em Oracle Spatial User's Guide and Reference (ORACLE, 2002), faz uma distinção entre operador e função. Os operadores tiram proveito dos índices espaciais, requerem que o índice espacial exista na primeira geometria especificada no operador e só podem ser utilizadas na cláusula WHERE da SQL3. Já as funções, não tiram proveito dos índices espaciais, só podem ser utilizadas em pequenas tabelas que não são indexadas e podem ser utilizadas numa lista SELECT ou na cláusula WHERE. Por exemplo, a operação SDO_WITHIN_DISTANCE gera um *buffer* ao redor da geometria e executa um índice primário e, opcionalmente, um índice secundário. A função SDO_GEOM.WITHIN_DISTANCE, gera um *buffer* ao redor da geometria e executa um filtro secundário⁴¹. O exemplo abaixo ilustra a diferença dessas duas abordagens, onde na coluna do lado esquerdo está apresentada a sintaxe para ambos os casos e do lado direito, um exemplo utilizando a Linguagem SQL3.

---

⁴⁰ Um exemplo para este caso são as funções de importação e exportação entre formatos de dados.

⁴¹ Um filtro secundário corresponde ao processo de refinamento explicado no capítulo anterior.

SDO_WITHIN_DISTANCE: “Ache todas as cidades dentro de uma distância inter-estadual”

SDO_WITHIN_DISTANCE operator:

```
boolean :=
MDSYS.SDO_WITHIN_DISTANCE
( <geometry-142>,
  <geometry-2>,
  'DISTANCE=<n>,
  QUERYTYPE=<querytype>
  [other optional
   parameters]'
)
```

```
SELECT C.Nome
FROM Interestatal I, Cidades C
WHERE I.Highway = 'I 170'
AND MDSYS.SDO_WITHIN_DISTANCE (
  C.Location, I.Geom,
  'distance=0.5 layer_gtype43=POINT') =
'TRUE';
```

SDO_GEOM.WITHIN_DISTANCE: “Ache todas as cidades dentro de uma distância inter-estadual”

SDO_GEOM.WITHIN_DISTANCE Function:

```
boolean :=
MDSYS.SDO_GEOM.WITHIN_DISTANCE
( <geometry-1>, <diminfo-1>,
  <distance>,
  <geometry-2>, <diminfo-2> )
or
boolean :=
MDSYS.SDO_GEOM.WITHIN_DISTANCE
( <geometry-1>, <distance>,
  <geometry-2>, <tolerance> )
```

```
SELECT C.Nome
FROM Interestatal I, Cidades C
WHERE I.Highway = 'I 170'
AND
MDSYS.SDO_GEOM.WITHIN_DISTANCE (
  C.Location,
  0.5,
  I.Geom, 0.00000005 ) = 'TRUE';
```

No DB2, segundo DB2 User's Guide (IBM, 2001), funções do tipo predicado espacial (tal como ST_OVERLAPS, que compara dois polígonos para verificar se eles se interceptam) só podem ser utilizadas na cláusula WHERE e, por serem consideradas custosas (demandam tempo de processamento e consomem memória), normalmente utilizam índices para otimização das consultas. Outros exemplos de

⁴² GEOMETRY-1 tem de ter um índice associado. Se o dado é geodésico, o índice espacial tem de ser o *R-tree*.

⁴³ LAYER_GTYPE é opcional. Quando recebe valor “ponto” significa que a consulta é numa coluna cuja geometria tem o tipo “ponto” e é utilizado somente o índice *Quad-tree*, para obter melhor desempenho. Existem outros dois argumentos relacionados com o índice que, quando não especificados adotam o valor *default*. O primeiro deles é QUERY_TYPE que quando recebe o valor “filter” significa que somente o primeiro filtro será executado, o *default* é ambos os filtros serem executados. O segundo é IDXTAB1 que especifica qual o nome da tabela de indexação, caso haja mais de um índice para a *geometry-1*.

predicados que utilizam índices espaciais são: ST_Contains, ST_Crosses, ST_Disjoint, ST_Distance, ST_Equals, ST_Intersects, ST_Touches e ST_Within. Para ver a descrição da sintaxe e semântica dessas funções, consultar DB2 User's Guide (IBM, 2001). O exemplo a seguir mostra uma consulta SQL3 utilizando-se o predicado espacial WITHIN, que retorna 1 (Verdadeiro) se a uma geometria está completamente dentro de uma segunda geometria e 0 (Falso), caso contrário. Neste exemplo, deseja-se “selecionar os clientes que estão na área determinada”.

```
SELECT *
FROM Clientes C
WHERE db2gse.ST_WITHIN(C.Location, :BayArea)=1
```

Observa-se que tanto os tipos espaciais básicos como o conjunto de operações definido, variam de uma ferramenta para a outra. Além disso, a semântica das operações também pode diferir de uma ferramenta para outra. A Figura 21, ilustra essa idéia para o caso da operação de interseção entre dois objetos espaciais.

Fig. 21 – O Resultado de uma interseção: $A \cap B$ (a) pode ser dois polígonos (b); dois polígonos e uma linha (c); ou dois polígonos, uma linha e um ponto (d).

Fonte: Extraída de Rigaux, Scholl e Voisard (2002)

Como observado anteriormente, os predicados espaciais são a base para as consultas espaciais e são um tipo especial de função espacial. No caso particular dos predicados topológicos, a maioria das ferramentas que implementam a topologia, utilizam uma definição formal dos relacionamentos espaciais, baseada no *The Dimensionally Extended 9 Intersection Model (DE-9IM)* - este modelo foi publicado em Clementini et al. (1993), a partir das idéias de Egenhofer e Herring (1991) - uma abordagem matemática que define relacionamentos espaciais entre pares de objetos espaciais de diferentes tipos e dimensões, através da matriz de interseção de suas

propriedades interior, fronteira e exterior. Todos os objetos espaciais ocupam uma posição no espaço definida por essas três propriedades. O exterior é todo o espaço não ocupado pelo objeto espacial. A fronteira serve de interface entre o interior e o exterior, representando a demarcação onde o conteúdo termina e o espaço ao redor começa. O interior é o espaço ocupado pelo objeto espacial. Em relação à dimensão, como observado no capítulo anterior, os objetos espaciais podem pertencer a uma das três categorias abaixo, de acordo com a geometria:

- dimensão zero (0): os objetos desta classe não têm comprimento nem área. Em outras palavras, a fronteira e o interior é o próprio ponto.
- dimensão um (1): os objetos desta classe têm um comprimento. Neste caso, a fronteira são os *end-points*, e o interior são os segmentos que estão entre os *end-points*. No caso de anéis, a fronteira é vazia e o interior é contínuo.
- dimensão dois (2): os objetos desta classe contêm área. Neste caso, a fronteira são os anéis interiores (caso existam) ou exteriores do objeto espacial. O interior é o espaço ocupado entre esses anéis.

A Figura 22 representa dois polígonos adjacentes, que podem estar representando dois municípios, dois bairros ou duas zonas de uso do solo distintas e vizinhas. Este exemplo ilustra como os relacionamentos topológicos são testados utilizando o modelo citado anteriormente. Suponha que se queira verificar se A e B são adjacentes e se A e B são iguais. Assume-se a definição de que A é adjacente a B quando $F(A) \cap F(B) \neq \emptyset$, ou seja, existe interseção entre as fronteiras de A e B; e de que A é igual a B quando $F(A) \cap F(B) \neq \emptyset$ e $I(A) \cap I(B) \neq \emptyset$, ou seja, existe interseção entre as fronteiras e os interiores de A e B. Desse modo, observando-se a matriz de adjacência de A e B, onde “V” (Verdadeiro) corresponde à existência de interseção e “F” (Falso) à não existência de interseção, podemos concluir que A é adjacente a B, mas A e B não são iguais.

Fig. 22 – Matriz de Interseção entre as propriedades Interior (I), Exterior (E) e Fronteira (F) de A e B.

Além das propriedades interior, exterior e fronteira, a dimensão também é uma propriedade importante na determinação de relacionamentos topológicos porque, através dela, sabe-se como os objetos devem ser comparados. Uma definição mais rigorosa da semântica dos relacionamentos espaciais deve levar em consideração a dimensão dos objetos sendo testados e também a dimensão resultante da interseção das propriedades (interior, exterior e fronteira) desses objetos. Por exemplo, pode-se definir que, duas linhas se cruzam quando a interseção dessas linhas é um objeto de dimensão zero, ou seja, o resultado da interseção do interior da linha A com o interior da linha B deve ser um ponto.

Alguns exemplos de relacionamentos topológicos, são apresentados nas Figuras 23.

Fig. 23 – Exemplos de relacionamentos topológicos

Fonte: Extraído de Oracle Spatial Training Guide (ORACLE, 2001)

4.3 Extensão da Linguagem SQL

Segundo Egenhofer et al. (1999), houve uma longa discussão acerca de se a Linguagem SQL era ou não viável para ser utilizada em bancos de dados espaciais e, segundo indicam as pesquisas citadas neste artigo, a Álgebra Relacional e a SQL sozinhas não eram capazes de suportar esse tipo de banco de dados, devido a algumas expressivas limitações, tais como a dificuldade de combinar recuperação de dados com aspectos de exibição em uma única consulta. Assim, muitas extensões para a SQL, através da adição de novos operadores e funções espaciais, foram propostas, até que se chegou ao padrão SQL3.

A SQL3 é uma extensão da SQL através da inclusão dos conceitos de orientação a objetos (novas operações e funções), dentre outras características.

A seguir é apresentado um exemplo⁴⁴ (IBM, 2001) em que um dono de loja de jóias está interessado em ampliar o seu negócio, e quer decidir a melhor localização para uma loja nova. Ele sabe o endereço e renda de cada cliente, e quer uma resposta à

⁴⁴ A sintaxe do exemplo é do *software* DB2 da IBM.

pergunta “*Quem são os clientes que estão há mais de 10 milhas das minhas lojas já existentes?*” Neste exemplo, a distância entre os clientes e as lojas é determinada e são retornadas aquelas maiores que 10.

```
SELECT *
FROM Clientes C
WHERE C.Renda > 50000 AND
      NOT EXIST (SELECT *
                  FROM Lojas L
                  WHERE Distance(C.Location, L.Location) < 10)
```

Observe que a sintaxe da SQL é muito semelhante à apresentada no Item 4.1. A localização dos clientes e das lojas são armazenadas como pontos, e devido à Arquitetura Integrada, são armazenados na mesma tabela dos dados de negócios dos clientes, tal como a renda do cliente.

Dolton (2001) fez um estudo comparativo das principais ferramentas que utilizam a SQL3, são elas DB2 (da IBM), ORACLE (da Oracle) e Informix (foi adquirido recentemente pela IBM, mas não sofreu alterações). Deste estudo ele concluiu que essas ferramentas têm algumas características comuns importantes, por exemplo, os tipos de dados espaciais geralmente são os mesmos, e incluem alguns tipos complexos tais como polígonos com “buracos”. Todos também contém um conjunto básico de funções espaciais tais como *distance*, *intersect*, *contains* e cálculo de área e comprimento. Mas quando se começa a olhar para um aspecto mais técnico, algumas diferenças começam a aparecer. Criar tipos de dados espaciais e funções em tabelas relacionais é um processo complexo e cada vendedor desenvolveu técnicas variadas de formato, uso das tabelas e funções de interação. Em relação à Linguagem de Consulta, embora a Linguagem seja a SQL, a forma de implementação varia de uma ferramenta para outra, como é mostrado a seguir:

IBM DB2 SPATIAL EXTENDER

```

SELECT A.se_row_id as AA, B.se_row_id as BB
FROM parque A, planicie B
WHERE ST_OVERLAPS (B.Geometry, A.Geometry);

```

INFORMIX SPATIAL DATABLADE MODULE

```

SELECT A.se_row_id as AA, B.se_row_id as BB
FROM parque A, planicie B
WHERE ST_OVERLAPS (B.Geometry, A.Geometry);

```

ORACLE 9i SPATIAL

```

SELECT A.sw_member as AA, B.sw_member as BB
FROM parque A, planicie B
WHERE MDSYS.SDO_RELATE (A.sdo_geometry, B.sdo_geometry,
                        'mask=ANYINTERACT querytype=JOIN')='TRUE';

```

5 SISTEMAS PARA INFORMAÇÃO GEOGRÁFICA

Neste capítulo será apresentada uma visão geral das ferramentas propostas, em relação a seus Modelos Conceitual, de Representação e Físico. No Modelo Físico, dentre outros itens, a funcionalidade será apresentada através de um conjunto de operações espaciais oferecido por essas ferramentas. Para as operações ou funções utilizadas nos testes práticos, também serão apresentadas a sintaxe e a semântica correspondentes.

5.1 SPRING

O SPRING é uma ferramenta SIG de Arquitetura Dual, desenvolvida pelo INPE – Instituto Nacional de Pesquisas Espaciais. Atualmente se encontra na versão 3.6.

5.1.1 Modelo Conceitual do SPRING

O Modelo de Dados do SPRING é apresentado na Figura 24, onde as classes básicas são apresentadas. Para o SPRING, um Banco de Dados Geográficos corresponde a um diretório (do disco rígido) no qual todos os dados de uma determinada aplicação são armazenados⁴⁵. Um Banco de Dados pode ter vários Projetos associados, por exemplo quando os dados estão em mais de um Sistema de Referência, podendo-se utilizar dois ou mais Projetos distintos para o mesmo banco. Um Projeto delimita a área física de trabalho através da determinação do retângulo envolvente, e corresponde a um sub-diretório dentro do diretório onde se localiza o banco de dados.

O Plano de Informação (PI) serve de suporte para a representação de diferentes tipos de dados geográficos. Segundo o Manual do SPRING (INPE, 2002), essa classe foi criada para fins de definição de interface e operações. Segundo Thomé (1998), está relacionada com a resolução do dado (no caso de PIs do tipo Imagem ou Numérico) e com a escala (no caso de PIs do tipo Temático, Numérico ou Cadastral). Além disso, estão relacionados a um determinado Projeto e são classificados de acordo com algum tipo de Categoria. A escolha da Categoria para um determinado PI depende do tipo do dado contido neste PI. Por exemplo, uma banda de uma imagem de satélite,

associada a um dado Sistema de Referência (cujas características estão definidas no Projeto), será mapeada para um PI, com um nome a ser determinado, e certamente pertencerá à Categoria do Modelo Imagem. Já uma rede viária, por exemplo, pode ser mapeada para a Categoria Cadastral ou para a Categoria Redes (para estudos de transporte, por exemplo), dependendo dos requisitos da aplicação em questão.

A classe Categoria é especializada em Geo-campos e Mapa de Objetos. Este trabalho concentra-se, principalmente, na classe Mapa de Objetos, que representa a Visão de Objetos.

Fig. 24 – Modelo Conceitual do SPRING.

Fonte: Extraído de Thomé (1998)

Um Mapa de Objetos, segundo o Manual do SPRING (INPE, 2002), armazena um Geo-objeto (ou Objeto) em conjunto com seus vizinhos, mantendo as relações de topologia. A classe Mapa de Objetos está relacionada com a classe Geo-objetos que por sua vez está associada a tabelas de atributos descritivos. As especializações da classe Mapa de Objetos são: Cadastral e Rede. A classe Cadastral utiliza a topologia arco-nó-polígono, sendo mais indicada para representar dados como divisas de município, cadastro urbano, etc. A classe Rede utiliza a topologia arco-nó, sendo mais indicada para representar redes em geral, como redes de água, esgotos, elétrica,

⁴⁵ Para o caso de bancos de dados Access e Dbase.

etc. A classe Geo-objeto (objeto), segundo Câmara (1995), representa um elemento único, que possui atributos descritivos (tabela de atributos) e está associado a múltiplas localizações geográficas e a classe Objeto-não-espacial (Não Espacial) engloba qualquer tipo de informação que não seja georreferenciada e que se queira agregar a um SIG.

Internamente, o SPRING consiste de um conjunto de tabelas relacionadas, conforme mostra Figura 25. Observe que o objeto (tabela “geoobject”) se conecta com os dados espaciais, representados pela estrutura espacial (arquivos binários), através da âncora (tabela “geoancho”). Uma âncora pode ser composta de uma lista de linhas, uma lista de polígonos ou uma lista de pontos. Cada Categoria de Objetos possui uma tabela de atributos associada. A tabela “CG000001” corresponde à tabela de atributos de uma categoria de objetos (por exemplo, a tabela Distritos) com os campos ‘geoid’ (identificador do Distrito) e ‘pop’ (população do Distrito). A tabela “infolaye” corresponde à classe Plano de Informação.

infolaye : Tabela									
		id	nome	modelo	contents	categoria	projeto	projecao	
	+	1	Distrito	64	12	1	1		-0,8
	*								

geobject : Tabela								
		id	nome	rotulo	categoria	visid	data	hora
	+	1	53	53	2			
	+	2	60	60	2			
	+	3	61	61	2			
	+	4	47	47	2			
	+	5	33	33	2			
	+	6	79	79	2			

CG000002 : Tabela								
		GEOID	CODP	SIGLA	NOME	CODAR	AREA	GMIPRIMARY
	+	1	DI53	MOE	Moema	VM	9	1
		2	DI60	PEN	Penha	PE	11	2
		3	DI61	PRD	Perdizes	LA	6	3
		4	DI47	LAJ	Lajeado	G	9	4
		5	DI33	IPI	Ipiranga	IP	10	5
		6	DI79	SAU	Saúde	VM	9	6

geoancho : Tabela									
		geoid	infoid	ganchorid	gvisid	area	perimetro	demanda	imp
		1	1	1	3	9082106,1875	12735,676504	0	
		2	1	2	3	11527763,344	18894,416427	0	
		3	1	3	3	6325294,9375	10663,965359	0	
		4	1	4	3	9013225,6875	16697,371269	0	
		5	1	5	3	10931919,75	16954,192754	0	
	+	6	1	6	3	9339261	12972,87224	0	
		7	1	7	3	2191568,5156	6618,129728	0	
		8	1	8	3	8284481,0938	13745,096767	0	

Na tabela intermediária (“geoancho”) um mesmo objeto pode estar associado a mais de um Plano de Informação, e, conseqüentemente, a várias representações gráficas. Seja, por exemplo, um banco de dados contendo um PI Cadastral com o Estado de São Paulo (EstadoSP) e outro PI Cadastral com os Municípios do Estado de São Paulo (MunicipioSP). Pode existir um objeto município, associado à tabela Município com informações demográficas, e este objeto pode estar tanto associado ao PI EstadoSP como ao PI MunicipiosSP. Este exemplo está representado na Figura 26 onde pode-se observar que apenas o código atribuído ao PI e o identificador da âncora são variáveis, sendo o identificador do objeto o mesmo para ambos os PI’s.

Fig. 26 – Exemplo de Associação de um objeto a mais de um PI

Por outro lado, um Plano de Informação também pode conter várias representações de objetos, como mostra a Figura 27, onde um Mapa de Lotes pode estar associado a vários objetos (Hospital, Residência ou Hotel).

Fig. 27 – Exemplo de Associação de um PI a mais de um Objeto

Fonte: Extraído do Manual do SPRING (INPE, 2002)

Os dados espaciais em formato binário estão estruturados segundo o Modelo Topológico (arco-nó-polígono), onde um polígono é formado por uma lista de arcos,

um arco é formado por uma lista de nós e pelos polígonos esquerdo e direito, e um nó é constituído de uma coordenada e de uma lista de linhas. Ao importar, por exemplo, os dados dos Distritos (representados por polígonos) para o Projeto São Paulo, os seguintes arquivos são gerados:

Pode-se observar no diagrama da Figura 25 que a consulta espacial seria mais eficiente se a tabela intermediária apontasse diretamente para o dado espacial (para a lista de linhas, polígonos ou pontos), sem precisar do identificador ‘ganchoid’ para fazer a conexão com os dados, o que diminuiria o número de acessos, e, conseqüentemente, demandaria menos tempo para ser processada. A tabela ‘geoancho’ corresponde à representação geométrica dos objetos e é o elemento chave através do qual se acessa a estrutura espacial, representando claramente a Arquitetura Dual do SPRING.

Mais informações sobre o Modelo de Dados do SPRING podem ser obtidas em Tutorial do SPRING (INPE, 2000), Manual do SPRING (INPE, 2002), Câmara (1995) e Thomé (1998).

5.1.2 Modelo de Representação do SPRING

a) Representação das Geometrias

Neste item será considerada apenas a Representação Vetorial. Segundo o Tutorial do SPRING (INPE, 2000), na representação vetorial, qualquer entidade ou elemento gráfico de um mapa é reduzido a três formas básicas: pontos, linhas e áreas (ou

polígonos). Um ponto é um par (X,Y) de coordenadas espaciais. As linhas, arcos, ou elementos lineares são um conjunto de pontos conectados. Um polígono é a região do plano limitada por uma ou mais linhas poligonais conectadas de tal forma que o último ponto de uma linha seja idêntico ao primeiro da próxima.

A Figura 28 mostra a hierarquia de classes da Representação Vetorial do SPRING.

Fig. 28 – Hierarquia de classes para representação vetorial

Fonte: Extraído de Tutorial do SPRING (INPE, 2000)

Antes de explicar cada uma das classes vetoriais do SPRING, serão definidas as primitivas geométricas, às quais essas classes fazem referência (extraído de Tutorial do SPRING (INPE, 2000)):

- Coordenada2D – Uma coordenada 2D é um objeto composto por uma localização singular $(x_i, y_j) \in \mathbb{R}$;
- Coordenada3D – Uma coordenada 3D é um objeto composto por uma localização singular (x_i, y_j, z) , onde $(x_i, y_j) \in \mathbb{R}$;
- Ponto2D – Um ponto 2D é um objeto que possui atributos descritivos e uma coordenada 2D;
- Linha2D – Uma linha 2D possui atributos e inclui um conjunto de coordenadas 2D;
- Isolinha – uma isolinha contém uma linha 2D associada a um valor real (cota);

- Arco Orientado – um arco orientado contém uma linha 2D associada a uma orientação de percorrimeto;
- Nó2d – um nó 2D inclui uma coordenada2D $(x_i, y_i) \in R$ e uma lista L de linhas 2D (trata-se da conexão entre duas ou mais linhas, utilizada para manter a topologia da estrutura);
- Nó Rede – um nó de rede contém um nó 2D e uma lista de arcos orientados, onde a cada instância associa-se uma impedância e um custo de percorrimeto;
- Nó 3d – um nó 3D instância desta classe contém uma coordenada 3D (x_i, y_i, z_i) e uma lista L de linhas 2D (trata-se da conexão entre três ou mais linhas de uma grade triangular);
- Polígono – um polígono contém uma lista de linhas 2D e uma lista de nós 2D que descrevem as coordenadas da área externa e das áreas internas que compõem o polígono.

A seguir, as definições das especializações das classes vetoriais são apresentadas (extraído do Tutorial do SPRING (INPE, 2000)):

- Conjunto de Pontos 2D – uma instância desta classe é um conjunto de pontos 2D utilizados para guardar localizações isoladas no espaço (p.ex. no caso de poços de petróleo);
- Conjunto de Isolinhas – uma instância desta classe é um conjunto de linhas, onde cada linha possui uma cota e as linhas não se interceptam;
- Subdivisão Planar – para uma região geográfica R qualquer, uma subdivisão planar contém um conjunto Pg de polígonos, L de linhas 2D e N de nós 2D;
- Grafo Orientado – uma instância desta classe é uma representação composta de um conjunto de nó de rede e de um conjunto de arco orientado 2D;
- Grade Triangular – uma instância desta classe contém um conjunto de nós 3D e um conjunto L de linhas 2D tal que todas as linhas se interceptam, mas apenas em seus pontos iniciais e finais;
- Mapa Pontos3d – uma instância desta classe é um conjunto de coordenadas 3d. Trata-se de um conjunto de amostras 3D.

b) Representação da Topologia

No SPRING, a classe Cadastral é representada com uma topologia do tipo arco-nó-

polígono. Já a classe Rede é representada por uma topologia do tipo arco-nó.

5.1.3 Modelo Físico do SPRING (Detalhes de Implementação)

a) Conjunto de Operações Implementadas

As funcionalidades do SPRING aqui exploradas dizem respeito, principalmente, às operações topológicas sobre os objetos espaciais, sobretudo os pertencentes à Categorias Cadastral.

O SPRING tem implementado as seguintes operações topológicas: TOCA, DENTRO, INTERCEPTA, DISJUNTO, CONTÉM, COBERTO POR, COBRE e IGUAL. Essas operações têm como entrada um plano de informação do tipo cadastral, ou uma coleção de objetos deste plano de informação, e geram como saída outro conjunto de objetos espaciais, que podem ser salvos em um novo plano de informação.

Além dessas operações topológicas, o SPRING ainda permite fazer operações métricas e direcionais.

Para outros Planos de Informação não Cadastrais, e em alguns casos não solucionáveis apenas acessando as opções dos Menus, é possível utilizar a Linguagem LEGAL (Linguagem Espacial para Geoprocessamento Algébrico) para desenvolver um procedimento que resolva o problema. A Linguagem LEGAL possui os seguintes operadores implementados: Atribua, Fatie, Pondere, Reclassifique, Operador Condicional, ReclAtrib, MediaZonal, MaxZonal, Espacialize, Atualize, MinZonal, VariedadeZonal e FaixaZonal.

b) Semântica das operações

A semântica diz respeito ao significado das operações, isto é, o resultado de uma operação depende de como ela é interpretada pelo *software*.

A seguir são apresentados os operadores topológicos obtidos do Manual do SPRING (INPE, 2002):

- TOCA: diz-se que um conjunto de pontos w1 toca outro conjunto w2 se a única coisa em comum entre eles está contida na união de suas fronteiras.

- DENTRO: diz-se que um conjunto de pontos $w1$ está dentro de outro conjunto $w2$ quando a intersecção dos dois conjuntos de pontos é o próprio $w1$.

- INTERCEPTA: o relacionamento intercepta engloba os relacionamentos CRUZA e SOBREPÕE.

CRUZA: duas linhas se cruzam se sua intersecção ocorre em um ponto interno de ambas (a intersecção de seus pontos limites será definida como toca); de forma similar, uma linha cruza uma área se o interior da linha está parcialmente interno e parcialmente externo à esta área.

SOBREPÕE: diz-se que dois conjuntos de pontos $w1$ e $w2$ possuem sobreposição quando o resultado de sua intersecção é uma figura da mesma dimensão de ambos. Este relacionamento é aplicável apenas a casos de elementos homogêneos.

- DISJUNTO: duas figuras são disjuntas se sua intersecção é vazia.
- CONTEM: diz-se que um conjunto de pontos $w1$ contém outro conjunto $w2$ quando o conjunto $w2$ está totalmente dentro do conjunto $w1$.
- COBRE e COBERTO POR: diz-se que um conjunto de pontos $w1$ cobre um conjunto $w2$ quando parte dos pontos (um ou mais) de $w2$ está dentro de $w1$, e parte está na fronteira de $w1$. O relacionamento coberto por é o oposto do relacionamento cobre.

c) Sintaxe das operações a serem utilizadas na análise

As análises, apresentadas no Capítulo 6, foram intercaladas com a utilização das seguintes funções extraídas do Manual do SPRING (INPE, 2002):

- MAPA DE DISTÂNCIA: um mapa de distância é um tipo de análise de proximidade (medida de distância entre objetos, comumente medida em unidade de comprimento) que apresenta zonas com larguras especificadas (distâncias) em

torno de um ou mais elementos do mapa. Pode ser utilizado para gerar “buffers” de distância, em análises de áreas de influência de determinado Equipamento Urbano, ou em estudos de distâncias a partir de um determinado eixo de logradouro, linha ou estação do Metrô, por exemplo. O Mapa de Distância pode ser gerado a partir de um PI Temático, Cadastral ou Numérico (MNT). No entanto, pode ter como saída um PI Temático, com as classes previamente definidas, ou um PI Numérico. Quando Numérico, gera-se uma grade de distâncias e uma imagem de distâncias. Assim, o primeiro passo para utilização dessa função é a criação de uma Categoria do tipo Temática com as respectivas classes, e/ou uma Categoria do tipo Numérica, de acordo com a análise que se efetuar.

- **RECORTAR PLANO DE INFORMAÇÃO:** permite subtrair planos de informações. O recorte ao PI de entrada é feito através de uma máscara predefinida (que pode ser um outro PI), ou através de uma área definida pelo usuário em tempo de execução. Através da máscara (ou da área selecionada), a função faz um recorte interno, ou seja, o PI de entrada é recortado de acordo com o conteúdo da máscara; ou externo, ou seja, o PI de entrada possuirá todos os dados exceto os que coincidirem com a máscara. Esta função está disponível no Menu Ferramentas.
- **INTERCEPTA, DENTRO, TOCA:** todos esses relacionamentos são acessíveis através do Painel de Controle opção Consultar-Objeto. A Figura 29 mostra a interface de consulta espacial do SPRING, onde os relacionamentos topológicos são efetuados. O exemplo está sendo ilustrado com o relacionamento intercepta, na consulta “selecionar os distritos que são cortados pela linha Vermelha do Metrô”.

Fig. 29 – Interface de Consulta Espacial do SPRING

d) Métodos de Indexação no SPRING

O SPRING utiliza o índice *Kd-tree* para pontos 2D e 3D e o *R-tree* para linhas, polígonos e âncoras. Os índices são criados automaticamente à medida que as geometrias são inseridas no sistema.

Dada uma consulta, por exemplo “determinar quais os polígonos adjacentes a um determinado objeto do tipo polígono”, inicialmente faz-se uma consulta ao retângulo envolvente (“box”) do objeto; em seguida, utiliza-se indexação para recuperar os elementos dentro do “box”; e por último se aplica o operador de topologia (neste caso, utiliza-se o operador “toca”).

e) Linguagens de Consulta e Ambiente de Consulta Espacial no SPRING

LEGAL (Linguagem Espacial para Geoprocessamento Algébrico) é a Linguagem de Consulta e Manipulação Espacial do SPRING. LEGAL é uma linguagem orientada a objetos, baseada no padrão ODMG, onde uma expressão de busca é construída a

partir da linguagem OQL.

Segundo Câmara (1995), a consulta espacial é aplicável a geo-objetos e geo-campos. No caso de geo-campos, dada a sua natureza contínua, os operadores disponíveis permitem apenas a recuperação baseada em seus atributos. O mesmo procedimento vale para objetos complexos (objetos cadastrais e de rede). Para geo-objetos, LEGAL oferece mecanismos para recuperação baseada em atributos e em restrições espaciais.

5.2 ORACLE SPATIAL

O ORACLE é um SGBD convencional da Oracle Corporation que, através do Módulo Spatial, suporta aplicações SIG. O ORACLE SPATIAL possui Arquitetura Integrada, implementa o Modelo Objeto-Relacional, e atualmente se encontra na versão 9i.

5.2.1-Modelo Conceitual do ORACLE SPATIAL

Segundo Sharma (2001), o Modelo de Dados do ORACLE SPATIAL, como mostra a Figura 30, é uma estrutura hierárquica consistindo de elementos, geometrias e camadas (*layers*), que correspondem à representação dos dados espaciais. *Layers* são compostos de geometrias, que por sua vez são compostas por elementos.

Um elemento é o componente básico de construção das geometrias do ORACLE SPATIAL, ou seja, são os tipos primitivos de dados suportados pelo ORACLE e que serão apresentados adiante. Os elementos são construídos utilizando-se coordenadas e, dependendo do tipo do elemento, pode ter um ou vários pares de coordenadas.

Uma geometria é uma representação de um objeto espacial, modelada como um conjunto ordenado de elementos primitivos (homogêneos ou heterogêneos). Cada geometria possui um identificador único, conhecido como Numeric Geometry Identifier (GID), que associa a geometria com o correspondente conjunto de atributos.

Um *layer* é uma coleção heterogênea de geometrias que compartilham o mesmo conjunto de atributos.

Fig. 30 – Modelo de Dados do ORACLE SPATIAL. O Estado do Hawaii é um exemplo de geometria que consiste de um conjunto de elementos (ilhas).

Fonte: Adaptado de Sharma (2001) e Oracle Spatial Training Guide (ORACLE, 2001)

5.2.2 - Modelo de Representação do ORACLE SPATIAL

a) Representação da Geometria

O ORACLE SPATIAL suporta três tipos primitivos de geometrias (ponto, linha e polígono) e uma coleção de outras geometrias compostas desses tipos primitivos, tais como: arcos circulares, círculos, linhas compostas e polígonos compostos, como mostra Figura 31.

Pontos são elementos compostos entre dois e quatro coordenadas. Quando o ponto é representado apenas pelo par (X,Y), freqüentemente o X corresponde à latitude do ponto e o Y corresponde à longitude do ponto, num determinado sistema de coordenadas.

Linhas são compostas de uma seqüência ordenada de dois ou mais pontos que definem os segmentos de linha. Auto-cruzamento de linhas são suportados, mas o polígono formado no cruzamento não tem um interior implícito. Arcos também não têm nenhum interior implícito. As coordenadas podem ser armazenadas seguindo-se a ordem do sentido horário ou anti-horário.

Polígonos são compostos de linhas conectadas que formam um anel, e cujo interior do polígono é implícito. Polígonos podem conter “buracos” que são construídos, por

definição, internos aos polígonos. Neste caso, o anel exterior e o anel interior do polígono são considerados como dois elementos distintos que juntos formam um polígono complexo. Ambos os tipos, exterior e interior, devem ser armazenados seguindo a ordem do sentido horário. Auto-cruzamento de polígonos não é permitido, embora o seja para linhas. Mas, se uma linha cruza ela mesma, então ela não pertence a um polígono.

Uma linha composta é uma combinação de segmentos retos e curvos. Seguem as mesmas restrições para linhas.

Nos polígonos compostos algumas extremidades podem ser formadas por linhas retas e outras por linhas curvas. Seguem a mesma semântica do tipo polígono.

Fig. 31 – Tipos de Dados Espaciais Primitivos do ORACLE SPATIAL

Fonte: Extraído de Oracle Spatial Training Guide (ORACLE, 2001)

b) Representação da Topologia

O ORACLE SPATIAL não implementa a topologia arco-nó-polígono explicitamente, como o SPRING, tendo dessa forma uma estrutura mais simples. Dada uma consulta espacial envolvendo elementos de uma mesma tabela, os relacionamentos são testados com operações de geometria computacional.

5.2.3 Modelo Físico do ORACLE SPATIAL (Detalhes de Implementação)

a) Objeto Espacial no ORACLE SPATIAL

No ORACLE SPATIAL, um *layer* corresponde à tabela espacial, cuja coluna é declarada como sendo do tipo geometria (MDSYS.SDO_GEOMETRY). Uma geometria é uma instância do tipo MDSYS.SDO_GEOMETRY e é armazenada em uma linha e uma coluna espacial (ou layer) particular da tabela. A Figura 32 ilustra um exemplo da estrutura de uma tabela (Distrito) e a Figura 46 ilustra um exemplo de consulta no SQL*PLUS, a interface de comandos SQL do ORACLE SPATIAL, tendo como resultado a geometria do polígono do Distrito cujo código é 53. Observe que o resultado é um conjunto de coordenadas correspondentes aos vértices do polígono. Na Figura 33 o resultado está apenas parcialmente apresentado, visto que o número de coordenadas é muito extenso.

A sintaxe do MDSYS.SDO_GEOMETRY é apresentada a seguir:

Parâmetros de SDO_GEOMETRY Object

```
SDO_GEOMETRY(SDO_GTYPE, SDO_SRID, SDO_POINT, SDO_ELEM_INFO,
SDO_ORDINATES)
```

SDO_GTYPE	NUMBER	Define o tipo de geometria armazenada no objeto
SDO_SRID	NUMBER	Define o Sistema de Referência Espacial, através de um identificador
SDO_POINT	SDO_POINT_TYPE	Utilizado para otimização do espaço quando a geometria é ponto
SDO_ELEM_INFO	SDO_ELEM_INFO_ARRAY	Elemento descritor de ordem
SDO_ORDINATES	SDO_ORDINATE_ARRAY	Contém as coordenadas dos elementos geométricos

Fig. 32 – Estrutura de uma tabela espacial no ORACLE SPATIAL

Fig. 33 – Instância de um polígono na tabela Distrito. Exemplo de consulta à coluna espacial (Resultado Parcial)

b) Conjunto de Operações Implementadas

As operações espaciais no ORACLE são implementadas, basicamente, através dos operadores espaciais (*spatial operators*) e das funções espaciais (*geometry functions*). A Tabela 2 (ORACLE, 2002) apresenta algumas dessas operações, agrupadas por operadores e funções, com uma descrição sucinta de suas funcionalidades. A diferença básica entre essas duas abordagens é que, no segundo caso (funções) não há utilização dos índices das tabelas espaciais, caso existam. Já para os operadores, é necessária a existência de índices nas tabelas espaciais e,

conseqüentemente, as consultas efetuadas com operadores são mais eficientes. Uma outra diferença é que os operadores só podem ser utilizados na cláusula WHERE de uma consulta SQL, enquanto que as funções podem ser utilizadas tanto na cláusula WHERE como na cláusula SELECT.

Tabela 2 – Operadores Espaciais e Funções Espaciais do ORACLE SPATIAL.

OPERADORES ESPACIAIS	
SDO_NN	Determina os vizinhos mais próximos a uma geometria
SDO_NN_DISTANCE	Determina quão distante estão os objetos retornados pelo operador SDO_NN de uma dada geometria
SDO_RELATE	Determina se duas geometrias se interagem (ou não) de algum modo
SDO_WITHIN_DISTANCE	Determina se uma geometria está a uma dada distância de outra
FUNÇÕES ESPACIAIS	
SDO_GEOM.RELATE	Determina como duas geometrias se interagem
SDO_GEOM.SDO_AREA	Calcula a área de um polígono de duas dimensões
SDO_GEOM.SDO_BUFFER	Gera um <i>buffer</i> (em forma de polígono) ao redor de uma geometria
SDO_GEOM.SDO_DIFFERENCE	Retorna a geometria correspondente à diferença topológica entre duas geometrias
SDO_GEOM.SDO_DISTANCE	Calcula a distância entre duas geometrias
SDO_GEOM.SDO_INTERSECTION	Retorna a geometria correspondente à interseção topológica entre duas geometrias
SDO_GEOM.SDO_LENGTH	Calcula o comprimento ou perímetro de uma geometria
SDO_GEOM.SDO_UNION	Retorna a geometria correspondente à união topológica entre duas geometrias
SDO_GEOM.VALIDATE_GEOMETRY	Determina se uma geometria é válida
SDO_GEOM.VALIDATE_LAYER	Determina se todas as geometrias armazenadas em uma coluna espacial são válidas
SDO_GEOM.WITHIN_DISTANCE	Determina se uma geometria está a uma distância específica (distância Euclidiana) de outra

Além das funcionalidades apresentadas acima, o ORACLE SPATIAL ainda possui outros recursos, como por exemplo, aqueles específicos para suportar aplicações que utilizam redes lineares, tais como rotas de ruas (para internet), transportes, *utilities* e redes de telecomunicações. No entanto, estes outros recursos não serão explorados

neste trabalho.

c) Semântica das operações

A seguir, são apresentados os relacionamentos topológicos utilizados no operador SDO_RELATE e na função SDO_GEOM.RELATE, extraídos de Oracle Spatial User's Guide and Reference (ORACLE, 2002):

- TOUCH: dois objetos se tocam quando suas fronteiras se interceptam, mas o interior não. Em outras palavras, suas fronteiras compartilham pelo menos um ponto comum, mas não há nenhum ponto comum a ambos os interiores.
- OVERLAPBDYDISJOINT: dois objetos têm um relacionamento do tipo 'OVERLAPBDYDISJOINT' (Overlap Boundaries Disjoint) quando o interior de um objeto intercepta a fronteira e o interior do outro, mas as duas fronteiras não se interceptam. É aplicável quando o teste é efetuado entre objetos do tipo linha e polígono.
- OVERLAPBDYINTERSECT: dois objetos têm um relacionamento do tipo 'OVERLAPBDYINTERSECT' (Overlap Boundaries Intersect) quando a fronteira e o interior de ambos se interceptam. É aplicável quando ambos os objetos testados são do tipo polígono.
- EQUAL: dois objetos são iguais quando elas possuem a mesma fronteira e o mesmo interior.
- INSIDE: ocorre quando o primeiro objeto está totalmente dentro do segundo e suas fronteiras não se tocam. É o oposto de CONTAINS.
- CONTAINS: ocorre quando o segundo objeto está totalmente dentro do primeiro e suas fronteiras não se tocam.
- COVERDBY: ocorre quando o primeiro objeto está totalmente dentro do

segundo e suas fronteiras se tocam em um ou mais pontos. É o oposto de COVERS.

- COVERS: ocorre quando a segundo objeto está totalmente dentro da primeiro e suas fronteiras se tocam em um ou mais pontos. É o oposto de CONTAINS.
- ANYINTERACT: dois objetos tem algum tipo de interação quando não são disjuntos.
- DISJOINT: dois objetos são disjuntos quando nem o interior nem a fronteira de ambos se interceptam, ou seja, não há relacionamento entre eles.

d) Sintaxe das operações a serem realizadas na análise

As análises apresentadas no Capítulo 6, foram realizadas com a utilização dos seguintes operadores e funções espaciais:

- SDO_RELATE
- SDO_GEOM.RELATE
- SDO_GEOM.SDO_BUFFER
- SDO_GEOM.SDO_AREA
- SDO_GEOM.SDO_UNION
- SDO_WITHIN_DISTANCE

Observa-se que algumas operações podem ser efetuadas utilizando o operador SDO_RELATE ou a função SDO_GEOM.RELATE, produzindo o mesmo resultado, porém no primeiro caso recorrendo-se aos índices e no segundo caso não. A função SDO_GEOM.RELATE somente compara duas geometrias, ou um pequeno número de geometrias, desse modo, o uso do índice não se faz necessário. Segundo Daniel Abugov em Oracle Spatial Discussion Forum (ORACLE, 2002), existem algumas razões pelas quais a função SDO_GEOM.RELATE existe, por exemplo, no caso de geometrias que são parte de uma aplicação e não estão armazenadas em tabelas, e também não possuem índices, a função SDO_GEOM.RELATE é um modo conveniente de determinar o relacionamento existente entre elas.

A seguir, as sintaxes das operações listadas acima são apresentadas.

SDO_RELATE

O operador SDO_RELATE utiliza o índice espacial para identificar os objetos espaciais que têm uma interação espacial particular com um dado objeto, tal como uma área de interesse, ou pares de objetos espaciais que têm interações espaciais particulares.

Esta operação executa ambos os filtros, primário e secundário.

O operador SDO_RELATE é booleano, ou seja, determina se uma determinada interação espacial ocorre, e só pode ser utilizado na cláusula WHERE.

Sintaxe: **SDO_RELATE** (*geometry1*, *geometry2*, *params*) = 'TRUE' or 'FALSE'

onde:

- *geometry1* : representa a coluna correspondente à geometria do objeto espacial. Deve ser espacialmente indexada. Tipo: MDSYS.SDO_GEOMETRY
- *geometry2*: representa a coluna correspondente à geometria do objeto espacial (que pode ou não ser indexada) ou a uma instância do objeto espacial. Tipo: MDSYS.SDO_GEOMETRY
- *params*: determinam o comportamento do operador SDO_RELATE. É subdividido em:
  - o *mask*: especifica o relacionamento topológico de interesse, que pode ser de um dos tipos: TOUCH, OVERLAPBDYDISJOINT, OVERLAPBDYINTERSECT, EQUAL, INSIDE, CONVERDBY, CONTAINS, COVERS, ANYINTERACT, ON. Permite, ainda, que hajam combinações desses relacionamentos, tais como INSIDE+TOUCH, onde '+' representa o operador *booleano* OR. No entanto, para obter melhor performance, recomenda-se efetuar as consultas individualmente e combinar os resultados utilizando o comando UNION ALL.
  - o *querytype*: determina o tipo da consulta, que pode ser WINDOW ou JOIN.

Se a consulta é do tipo WINDOW, *geometry2* pode ser uma ou mais instâncias de uma tabela, uma variável *bind* ou um construtor

SDO_GEOMETRY. WINDOW é utilizada para comparar instâncias de uma geometria (*geometry2*) com todas as geometrias em uma coluna (*geometry1*). JOIN é utilizada quando se quer comparar todas as geometrias de uma coluna com todas as geometrias de outra coluna. Neste caso, a coluna correspondente à *geometry2* também deve ser indexada e, para melhor performance, os índices devem ser do mesmo tipo e se o índice for *quadtree*, ambos devem ter o mesmo número de níveis (o mesmo valor para *sdo_level*).

- *idxtab1*: especifica o nome do índice, quando a *geometry1* tem mais de um índice;
- *idxtab2*: especifica o nome do índice, quando a *geometry2* tem mais de um índice (utilizado quando a consulta é do tipo JOIN).

SDO_GEOM.RELATE

Determina o relacionamento espacial existente entre dois objetos espaciais.

É indicado para bases de dados não volumosas, uma vez que não utiliza índices.

Pode ser utilizada tanto na cláusula SELECT como na cláusula WHERE.

Sintaxe: **SDO_GEOM.RELATE** (*geometry1*, *mask*, *geometry2*, *tol*) = **VARCHAR2**

onde:

- *geometry1* : objeto espacial;
- *mask* :
  - o se uma lista de relacionamentos for especificada, a função irá retornar o nome do relacionamento, caso exista. Se todos os relacionamentos forem falsos, a função retornará FALSE;
  - o se for utilizada a palavra-chave DETERMINE, a função irá retornar o nome do relacionamento mais evidente existente entre as geometrias. Esta opção só pode ser utilizada na cláusula SELECT;
  - o se for utilizada a palavra-chave ANYINTERACT, a função irá retornar TRUE se as geometrias não forem disjuntas;
  - o para os relacionamentos CONTAINS, COVEREDBY, COVERS,

DISJOINT, EQUAL, INSIDE, OVERLAPBDYDISJOINT, OVERLAPBDYINTERSECT, TOUCH e agrupamentos destes relacionamentos, a função retornará o nome do próprio relacionamento se o mesmo ocorrer entre as geometrias ou FALSE, caso contrário.

- *geometry2* : objeto espacial;
- *tol*: valor de tolerância utilizado para associar um nível de precisão aos dados espaciais. Deve ser um número não negativo maior que zero. Por exemplo, para bases geo-referenciadas, um *tol* de valor 5 indica uma tolerância de 5 metros.

SDO_GEOM.SDO_BUFFER

Gera um “buffer” em forma de polígono ao redor da geometria.

Sintaxe: **SDO_GEOM.SDO_BUFFER** (*geometry1*, *geometry2*, *tol*) =

MDSYS.SDO_GEOMETRY

onde:

- *geometry1* : objeto espacial;
- *geometry2* : objeto espacial;
- *tol*: tolerância.

SDO_GEOM.SDO_AREA

Retorna a área de um polígono bi-dimensional.

Sintaxe: **SDO_GEOM.SDO_AREA** (*geometry1*, *tol*) =

MDSYS.SDO_GEOMETRY

onde:

- *geometry1* : objeto espacial;
- *tol*: tolerância.

SDO_GEOM.SDO_UNION

Retorna a geometria correspondente à união topológica de duas geometrias.

Sintaxe: **SDO_GEOM.SDO_UNION** (*geometry1*, *geometry2*, *tol*) =

MDSYS.SDO_GEOMETRY

onde:

- *geometry1* : objeto espacial;
- *geometry2* : objeto espacial;
- *tol*: tolerância.

SDO_WITHIN_DISTANCE

Utiliza o índice espacial para identificar um conjunto de objetos espaciais que estão a uma dada distância de determinado objeto espacial (tais como uma área de interesse ou um ponto de interesse).

Sintaxe: **SDO_WITHIN_DISTANCE** (*geometry1*, *geometry2*, *params*) = 'TRUE'

onde:

- *geometry1* : coluna contendo as geometrias de uma tabela espacial;
- *geometry2* : objeto espacial a partir do qual será verificada a distância;
- *params*:
  - o *distance*: especifica o valor da distância;

d) Métodos de Indexação no ORACLE SPATIAL

O ORACLE SPATIAL fornece duas opções de índice: o *Quad-tree* e o *R-tree*. Segundo Sharma (2001), o índice *R-tree* pode ser utilizado em lugar do *Quad-tree*, ou em conjunto com ele. Em adição, o *R-tree* pode ser utilizado para dados 3D e 4D.

e) Linguagens de Consulta e Ambiente de Consulta Espacial no ORACLE SPATIAL

A Linguagem de Consulta do ORACLE SPATIAL é a SQL. O ORACLE SPATIAL possui um ambiente de Consulta Espacial muito restrito (Oracle Spatial Index Advisor, conforme apresentado na Figura 34), onde são possíveis consultas envolvendo apenas um *layer*. A Figura 34 ilustra a interface de consulta espacial do ORACLE SPATIAL, onde no exemplo estão selecionados (na cor branca) os Distritos que têm alguma interação com a janela definida pelo usuário. As consultas SQL mais elaboradas, são formuladas utilizando-se uma das interfaces SQL*PLUS ou SQLPLUS Worksheet.

Fig. 34 – Interface de Consulta Espacial do ORACLE SPATIAL. Exemplo de Consulta utilizando o relacionamento ANYINTERACT

Os comandos SQL podem ser embutidos na Linguagem de Programação do ORACLE (PL/SQL) ou, ainda, em Linguagem C/C++.

Para visualização dos resultados das consultas, utiliza-se o Enterprise Manager Console, ou outro visualizador externo, compatível com o ORACLE. O Enterprise Manager Console é o ambiente de gerenciamento de tabelas (tanto espaciais quanto não espaciais) do ORACLE.

6 APLICAÇÃO PRÁTICA E TESTES EFETUADOS

Para demonstrar como os conceitos apresentados anteriormente são importantes e influenciam diretamente na eficiência dos Sistemas Gerenciadores de Banco de Dados Geográficos, foram desenvolvidos testes de funcionalidade sob as ferramentas propostas.

Dada a especificação de um conjunto de operações a serem efetuadas com uma determinada base de dados espacial, os testes de funcionalidade envolvem:

- Verificar se as operações estão implementadas nas ferramentas e, caso estejam, quais os procedimentos para efetuá-las.
- Verificar quais as implicações durante a geração das consultas espaciais, em relação a:
  - Se a interface com o usuário e/ou a Linguagem de Consulta são amigáveis ao usuário, ou seja, são fáceis de usar;
  - Se houve necessidade de customização (implementação) na geração das consultas;
  - Se o resultado das consultas é confiável (correto) e satisfaz as necessidades da aplicação;
  - Se houve necessidade de ferramentas para visualização dos resultados;
- Fazer observações sobre:
  - As operações utilizadas;
  - A semântica e sintaxe dos operadores ou das funções;
  - A utilização de índices espaciais;
- Analisar as implicações das consultas em relação às Arquiteturas das ferramentas (Dual e Integrada), quando envolvendo dados espaciais e não espaciais simultaneamente.

6.1 Descrição dos Dados

Para os testes de funcionalidade serão utilizados os dados da Base Cartográfica do Município de São Paulo, os Distritos Municipais, as Linhas do Metrô e informações sobre Equipamentos Urbanos, oriundos do MOC – Mapa Oficial da Cidade de São

Paulo; e informações censitárias do IBGE, disponíveis no *site* do IBGE. Embora ambos os sistemas trabalhem com Projeção Cartográfica, este detalhe não será levado em consideração neste trabalho. Uma descrição mais detalhada dos dados está apresentada no Apêndice A.

6.2 Testes de Funcionalidade

Os testes de funcionalidade objetivaram verificar se e como as ferramentas em estudo, com arquiteturas distintas, suportam determinados tipos de consultas espaciais. Para tanto foi escolhido o cenário de Planejamento Urbano e selecionados alguns tipos de consultas, envolvendo componentes espaciais, que normalmente são implementadas neste tipo de aplicação.

A título de organização, os testes foram divididos em estudos de casos que não têm, necessariamente, correlação entre si, embora compartilhem a mesma base de dados do Município de São Paulo. Torna-se claro que o objetivo do trabalho não é explorar os problemas do Planejamento Urbano, ou elaborar soluções para os mesmos. O foco desses estudos de casos estará no comportamento das ferramentas frente às necessidades de consultas espaciais específicas em cada situação. Desse modo, é apresentado sucintamente o problema de cada estudo de caso e, em seguida, as consultas espaciais envolvidas, serão exploradas, comentando-se as soluções no ORACLE SPATIAL e no SPRING.

Os casos a serem analisados são:

Caso 1 - Alocação de Equipamentos Urbanos;

Caso 2 - Identificação dos Distritos que são servidos pelo Metrô;

Caso 3 - Análise de proximidade de Equipamentos Urbanos em relação aos meios de transporte;

Caso 4 - Identificação dos Equipamentos pertencentes a um determinado Distrito Municipal;

Caso 5 - Identificação dos Distritos adjacentes a um Determinado Distrito; e

Caso 6 – Consulta Direcional.

6.2.1 Alocação de Equipamentos Urbanos (Caso 1)

O Caso 1 se enquadra na temática de localização de áreas prioritárias para implantação de Equipamentos Sociais. Neste trabalho foi escolhida a área de Saúde, para a qual foram levantados os Equipamentos do Tipo 1⁴⁶, conforme mostra a Figura 35.

Fig. 35 – Município de São Paulo: Distritos e Unidades de Saúde

Fonte: Fundação do Desenvolvimento Administrativo (FUNDAP)

O problema consiste em determinar supostas áreas carentes por novas instalações de Unidades de Saúde, também chamadas de áreas de demanda reprimida ou vazios de atendimento, para o Município de São Paulo.

A determinação dessas supostas áreas carentes envolve o conhecimento da área de abrangência dos Equipamentos Urbanos já existentes no município - para que se possa analisar o restante não atendido - e o conhecimento sobre o perfil populacional distribuído no Município.

---

⁴⁶ O Tipo 1 são hospitais, mas para o intuito deste trabalho estamos considerando-os apenas como Unidades de Saúde.

Dividiu-se o estudo do Caso 1 em duas partes:

- Parte 1 - Geração de raios de influência de Equipamentos Urbanos
- Parte 2 – Cálculo do Vazio de Atendimento

6.2.1.1 Geração de raios de influência de Equipamentos Urbanos (Caso 1/Parte 1)

Uma das metodologias adotadas para determinação das Áreas Prioritárias, tem como base a geração dos Raios de Influência das Unidades de Saúde, através dos quais é possível gerar “buffers” que correspondem à proporção da população atendida ao redor da Unidade.

O Cálculo dos Raios de Influência (R), segundo Barbosa (1999), para a área de Saúde, é dado pela seguinte fórmula:

$$R = \sqrt{\frac{(CapacidadeDaUnidade \times AreaDoDistrito)}{(PopulacaoTotalDoDistrito) \times \pi}}$$

Como hipótese simplificadora para atender aos objetivos desta dissertação, e considerando-se as Unidades de Saúde como genéricas, serão considerados os seguintes parâmetros:

- A População Total do Distrito, considerando que todos os municípios, independente do perfil, podem utilizar todas as Unidades de Saúde. O Distrito foi adotado como unidade de estudo. A informação sobre a população estava inicialmente agrupada por Setores Censitários do IBGE e foi também incorporada aos Distritos da Prefeitura Municipal de São Paulo;
- A Capacidade de Atendimento da Unidade de Saúde, segundo Barbosa (1999), expressa a possibilidade de oferta de serviços de cada unidade de saúde e é calculada através do cruzamento entre o número de consultórios médicos existentes em cada equipamento, o número de turnos de funcionamento e o padrão internacional de duas consultas por ano por habitante. Para este trabalho, considerou-se a Capacidade de Atendimento de cada unidade como sendo constante e igual a 20.000 habitantes, o valor da capacidade geral de uma unidade de saúde.

Conhecendo-se bem o tipo de unidade de saúde, pode-se classificar a população usuária deste tipo de unidade e assim, o conjunto população será mais reduzido. Considerando estas unidades como genéricas, significa que não se levará em conta

esse tipo de classificação, e será adotada a população total para os testes a serem efetuados, o que não compromete a análise funcional das ferramentas, mas deve ser levado em conta na avaliação dos resultados.

Além disso, admitiu-se que não há nenhuma restrição em relação ao uso de uma Unidade de Saúde, situada em determinado Distrito, por munícipes residentes em outro Distrito.

De acordo com a fórmula acima, percebe-se que as Unidades de Saúde terão raios de influência distintos e, dependendo da sua localização, podem se sobrepor uns aos outros, podendo, inclusive, ultrapassar os limites dos Distritos.

Solução ORACLE SPATIAL (Caso1/Parte1)

Inicialmente será criada uma tabela de saída para armazenar os raios de influência, juntamente com seus “buffers” correspondentes, para utilização em análises futuras.

Os procedimentos a seguir estão escritos em Linguagem PL/SQL (PL/SQL User's Guide and Reference, 2002) que é a Linguagem de Programação nativa do ORACLE.

```

/*
  Criação da tabela 'raioinf' que contém os Raios de Influência e respectivos "buffers"
*/

DROP TABLE raioinf;

CREATE TABLE raioinf(
id number primary key,
raio number,
popdist number,
areadist number,
geom MDSYS.SDO_GEOMETRY);

/* Atualização do Arquivo de Metadados */

DELETE FROM USER_SDO_GEOM_METADATA
WHERE TABLE_NAME = 'raioinf' AND COLUMN_NAME = 'geom';

INSERT INTO USER_SDO_GEOM_METADATA values(
'raioinf',
'geom',

```

```

MDSYS.SDO_DIM_ARRAY (
MDSYS.SDO_DIM_ELEMENT ('X',308000.000,369000.000,0.05),
MDSYS.SDO_DIM_ELEMENT ('Y',7345000.000,7417000.000,0.05)
),
NULL
);
COMMIT;

```

Segue abaixo a descrição do procedimento GERA_BUFFERS. Este procedimento irá gerar “buffers” para todas as Unidades de Saúde, por isso foi utilizada uma estrutura de ‘loop’. Observe que a função SDO_GEOM.SDO_AREA foi utilizada na cláusula SELECT de modo que, à medida que os Distritos (cujo acesso à informação espacial se dá através da referência d.geom) vão sendo selecionados, a área dos mesmos já é calculada e armazenada na variável AreaDist. No mesmo SELECT, a população do Distrito (referenciada por d.PopTotal) é lida e armazenada na variável PopDist. Essa é a característica principal da Arquitetura Integrada, onde, através de um mesmo SELECT se acessa simultaneamente o conjunto de informações espaciais e os atributos de um objeto, uma vez que estão armazenados no mesmo registro da mesma tabela, porém em colunas distintas. Na cláusula WHERE do SELECT foi utilizada a função SDO_GEOM.RELATE que garante que o Distrito selecionado é aquele onde a Unidade de Saúde ativa (referenciada pelo cursor implícito TabAux) está dentro (INSIDE) do Distrito.

Após calculada a área do Distrito e obtida a população do mesmo, pode-se gerar o “buffer” para a Unidade de Saúde ativa utilizando-se a fórmula do cálculo do Raio de Influência, através da função SDO_GEOM.SDO_BUFFER. Os “buffers” são calculados para todas as unidades, até que se encerre o “loop”.

```

CREATE OR REPLACE PROCEDURE GERA_BUFFERS IS

/*
Gera “buffers” representativos para os Raios de Influência das Unidades de Saúde, em
função da População Total do Distrito e da Capacidade da Unidade.
*/

BEGIN

DECLARE

```



```

/* Declaração de Variáveis */

AreaDist distrito.Area%TYPE;      -- Área do Distrito
PopDist distrito.PopTotal%TYPE;    -- População Total do Distrito
R NUMBER;                          -- Raio de Influência
Unidade saudetp1.Id11%TYPE;        -- Unidade de Saúde
Buf raioinf.GEOM%TYPE;             -- Buffers a serem gerados
TabAux saudetp1%ROWTYPE;           -- Tabela Auxiliar

/* Declaração de Constantes */

CAP CONSTANT NUMBER := 20000; -- CAP = Capacidade de Atendimento da Unidade
Pi CONSTANT NUMBER := 3.14;    -- Valor de Pi utilizado no cálculo do Raio

BEGIN

/* Corpo da Procedure */

FOR TabAux IN (SELECT * FROM saudetp1)
LOOP
    SELECT SDO_GEOM.SDO_AREA (d.geom,0.005),d.PopTotal
    INTO AreaDist, PopDist
    FROM distrito d
    WHERE SDO_GEOM.RELATE(TabAux.Geom,'INSIDE',d.Geom,0.05)='INSIDE';

    Unidade:=TabAux.Id11;
    R:= SQRT((CAP*AreaDist)/(PopDist*Pi));
    Buf := SDO_GEOM.SDO_BUFFER(TabAux.Geom,R,0.05);
    INSERT INTO raioinf values(Unidade, R, PopDist, AreaDist, Buf);
END LOOP;
COMMIT;

END;

END;

```

Observa-se que no procedimento anterior foi utilizada a função SDO_GEOM.RELATE, que não recorre a índices. Isso faz com o procedimento fique mais lento. Do mesmo modo, a geração de “buffers” no ORACLE SPATIAL não utiliza índices, uma vez que o “buffer” só pode ser calculado através da função SDO_GEOM.SDO_BUFFER.

O procedimento parcialmente apresentado a seguir é idêntico ao anterior, porém,

utilizando-se o operador MDSYS.SDO_RELATE, que necessita da criação de um índice (no caso foi utilizado o *R-tree*) para a tabela Distrito.

No procedimento a seguir é importante observar, na sintaxe do operador MDSYS.SDO_RELATE, a ordem dos parâmetros *geometry1* e *geometry2*, bem como o tipo de consulta utilizado ('Window' ou 'Join'). Observe que para o parâmetro *geometry1* foi utilizado como referência a coluna espacial da tabela Distrito (d.geom) que é a coluna indexada. Para o parâmetro *geometry2* foi utilizada a geometria do objeto correspondente à Unidade de Saúde ativa (TabAux.geom). Foi escolhido o tipo 'Window' pois o parâmetro *geometry2* correspondia a apenas um objeto, não a uma tabela inteira. Observe ainda que, diferentemente do procedimento anterior, que utilizou o relacionamento 'INSIDE' para selecionar o Distrito cuja Unidade de Saúde estava dentro dele, neste caso foi utilizado o relacionamento 'CONTAINS' para selecionar o Distrito que contém a Unidade de Saúde ativa, devido à ordem dos parâmetros *geometry1* e *geometry2* no operador.

```
/* Corpo da Procedure */

FOR TabAux IN (SELECT * FROM saudetp1)
LOOP
    SELECT SDO_GEOM.SDO_AREA (d.geom, 0.005),d.PopTotal
    INTO AreaDist, PopDist
    FROM distrito d
    WHERE MDSYS.SDO_RELATE(d.geom, TabAux.geom,
                           'mask=CONTAINS querytype=WINDOW') = 'TRUE';

    Unidade:=TabAux.Id11;
    R:= SQRT((CAP*AreaDist)/(PopDist*Pi));
    Buf := SDO_GEOM.SDO_BUFFER(TabAux.geom,R*0.05);
    INSERT INTO raioinf values(Unidade, R, PopDist, AreaDist, Buf);
END LOOP;
```

Resultado: “buffers” gerados

Solução SPRING (Caso1/Parte1)

A geração de “buffers” no SPRING se dá através da Função Mapa de Distância, apresentada no Capítulo 5. O resultado da função Mapa de Distância, independente do tipo do PI de entrada, é do tipo Temático ou Numérico. Neste exemplo, será utilizada apenas a saída Temática.

A aplicação demanda a geração de “buffers” para todas as Unidades de Saúde. Para efetuar essa tarefa no SPRING, num único passo, é preciso que os dados de entrada estejam num PI Temático. No PI Cadastral, até seria possível efetuar esse processo num único passo, mas, para esta Categoria, não está implementada uma opção que seleciona todos os objetos de uma só vez. Desse modo os objetos teriam de ser selecionados um a um, o que não é viável.

No SPRING não é possível gerar “buffers” a partir de uma fórmula cujos parâmetros devam ser obtidos de valores de campos das tabelas não espaciais, como especificado anteriormente. Em outras palavras, no SPRING não é possível fazer uma consulta espacial envolvendo o objeto espacial e o atributo ao mesmo tempo. Os “buffers” são gerados através de um valor que deve ser informado pelo usuário do sistema, relativo a uma dada distância, por isso o nome “Mapa de Distância”. Assim, para a análise da função Mapa de Distância do SPRING, utilizou-se um valor médio para o Raio de Influência, calculado em função dos Raios de Influência gerados no ORACLE SPATIAL, cujo valor é de 813 metros.

Resultado: “buffers” gerados

Observa-se que o resultado final do processamento é a união dos “buffers” que se interceptam.

Considerações do Caso 1/Parte 1

A principal operação testada neste estudo de caso foi a geração de “buffers” ao redor dos objetos espaciais, apesar de terem sido utilizadas também as funções `SDO_GEOM.SDO_AREA` e `SDO_GEOM.SDO_RELATE` no teste do ORACLE SPATIAL. Observa-se que ambos os sistemas têm recursos implementados para essa finalidade. No SPRING, geram-se “buffers” através da função Mapa de Distância e no ORACLE SPATIAL, através da função `SDO_GEOM.SDO_BUFFER`.

O problema apresentou, ainda, a necessidade de geração de “buffers” onde o parâmetro ‘Raio’ era calculado a partir de uma fórmula envolvendo campos das

tabelas Distrito (População Total) e Unidades de Saúde (Capacidade). A Arquitetura Integrada do ORACLE SPATIAL permitiu, em conjunto com os recursos da Linguagem PL/SQL, a geração dos Raios de Influência, mas esta operação não foi possível no SPRING, onde foi utilizado um Raio Médio para geração dos “buffers”, fazendo com que todas as Unidades de Saúde tivessem o mesmo raio, o que não é verdadeiro. Poder-se-ia ter gerado o “buffer” representativo da área de influência para cada unidade, individualmente, fornecendo-se o valor correto do Raio de Influência (previamente calculado) para aquela unidade, na execução da função Mapa de Distância. Mas percebe-se que, para o volume de dados de Unidades de Saúde (152 unidades) esse trabalho se torna inviável. Em resumo, para processar toda a base de Unidades de Saúde num único passo, o SPRING somente permite que se entre com um valor único para o Raio.

Em relação ao resultado gerado, o ORACLE SPATIAL não produz um resultado com visualização espacial, após a execução de um comando SQL, como ocorre na maioria das ferramentas SIGs. Assim, foi criada uma tabela auxiliar (Raioinf) para armazenar o resultado dos “buffers” gerados. No SPRING, os resultados foram gerados para o PI Temático definido, que pode ser acessado posteriormente via Painel de Controle. No entanto, o resultado de saída do SPRING encontra-se na forma de união dos buffers que se interceptam, fazendo com que se perca a referência à Unidade de Saúde, isto é, os “buffers” não podem ser individualizados, tanto utilizando-se uma Categoria Temática como utilizando-se uma Categoria Cadastral como entrada.

Seria interessante acrescentar no SPRING a possibilidade de conversão dos dados temáticos, gerados a partir dos Mapas de Distância, em cadastrais, uma vez que ambos são vetoriais, gerando-se automaticamente uma Categoria Objeto correlacionada à Categoria Cadastral, com os identificadores das classes temáticas ou com os identificadores dos objetos a partir dos quais foram gerados.

6.2.1.2 Cálculo do Vazio de Atendimento (Caso 1/Parte 2)

O Vazio de Atendimento corresponde à área não atendida pelas Unidades de Saúde. O caso mais simples de cálculo se dá através da subtração da Área Total do Município, do somatório das áreas dos “buffers” gerados no exemplo anterior.

No entanto, para o cálculo mais correto dos Vazios de Atendimento, seria necessário analisar as interseções entre os “buffers”, uma vez que significam redundância de áreas de atendimento, ou seja, as interseções significam regiões que são cobertas mais de uma vez pelas Unidades de Saúde. Se essas áreas de sobreposição forem levadas em consideração, a área do Vazio de Atendimento será menor, uma vez que as áreas de abrangência das Unidades de Saúde podem ser redistribuídas para o restante do Município. A Figura 36 ilustra os possíveis casos de interseção. O caso mais simples seria a ocorrência de interseções em apenas dois “buffers”. O caso mais complexo seria a ocorrência de interseção entre um aglomerado de “buffers”.

Fig. 36 – Casos de Interseção entre ‘buffers’

Além disso, pode ocorrer a extrapolação de um “buffer” em relação ao Limite do Município, como mostra Figura 37.

Fig. 37 – Casos de Extrapolação dos ‘buffers’ em relação ao Limite do Município (Zona Leste)

Nestes tipos de análise é importante observar bem o que as figuras geométricas, tais como os “buffers”, representam, e quão real ou aproximada é a resposta obtida, pois, dependendo das necessidades da aplicação o problema pode ser simplificado e um resultado aproximado poderá ser satisfatório. Caso contrário, é preciso definir uma solução para um problema mais complexo, que leve em consideração as áreas das interseções e a área extrapolada, para se obter um resultado mais próximo do real.

O intuito desse trabalho, entretanto, é testar a funcionalidade das ferramentas, e, por isso, optou-se pela redução da complexidade do problema, considerando que um resultado aproximado seja satisfatório. Assim, a determinação do Vazio de Atendimento será feita pela formulação mais simples, citada anteriormente, explorando as operações de ‘União’ e ‘Subtração’ de objetos espaciais.

Solução ORACLE SPATIAL (Caso 1/ Parte 2)

Após gerados os “buffers” correspondentes aos Raios de Influência, foi obtida uma figura representativa da União desses “buffers” que será utilizada posteriormente para o cálculo da Área Carente de Unidades de Saúde.

Mostra-se a seguir a descrição do procedimento UNE_BUFFERS utilizando as funções SDO_GEOM.SDO_UNION e SDO_GEOM.SDO_AREA. A união é feita com todos os “buffers”, independente se eles se interceptam ou não. A função SDO_GEOM.SDO_AREA foi utilizada apenas para calcular a área resultante da união dos “buffers” e armazená-la na tabela de saída (TabUniaoBuf).

```
CREATE OR REPLACE PROCEDURE UNE_BUFFERS IS
```

```
/*
```

```
Calcula a União de todos os raios de influência ("buffers") e armazena a geometria resultante na tabela TAB_UNIAO_BUF.
```

```
Por último, a área da figura resultante é calculada e também armazenada na tabela TAB_UNIAO_BUF.
```

```
*/
```

```
BEGIN
```

```
DECLARE
```

```

/* Declaração de Variáveis */

TabAux raioinf%ROWTYPE;           -- Tabela Auxiliar
UNIAO_BUF MDSYS.SDO_GEOMETRY;     -- Geometria resultante da união dos buffers
AREA_RES NUMBER;                  -- Área resultante da união dos buffers

BEGIN

/* Corpo da Procedure */

UNIAO_BUF:=NULL;
FOR TabAux IN (SELECT * FROM raioinf)
LOOP
    UNIAO_BUF := SDO_GEOM.SDO_UNION(UNIAO_BUF,TabAux.Geom,0.005);
END LOOP;
AREA_RES := SDO_GEOM.SDO_AREA(UNIAO_BUF,0.005);
INSERT INTO TAB_UNIAO_BUF VALUES(UNIAO_BUF,AREA_RES);
COMMIT;

END;

END;

```

Resultado: União dos “buffers”

O Cálculo dos Vazios de Atendimento, ou seja, a área (aproximada) do Município

não atendida pelas Unidades de Saúde, se dá, simplificada, através da Subtração da Área Total do Município pela Área Resultante da União dos “buffers”.

O módulo a seguir descreve o procedimento GERA_VAZIOS utilizando a função e SDO_GEOM.SDO_DIFFERENCE. A função SDO_GEOM.SDO_AREA foi utilizada apenas para calcular a área do Vazio de Atendimento e armazená-la na tabela de saída (Tab_Vazios).

```
CREATE OR REPLACE PROCEDURE GERA_VAZIOS IS

/*
Calcula a Área do Município não atendida pelas Unidades de Saúde, subtraindo a Área do
Município da Área Correspondente à União dos “Buffers” previamente calculada.
*/

BEGIN

DECLARE

/* Declaração de Variáveis */

LIMITE MDSYS.SDO_GEOMETRY;          -- Limite do Município
AREA_MUN NUMBER;                    -- Área do Município
AREA_VAZIO NUMBER;                   -- Área Total de Vazios de Atendimento
VAZIO MDSYS.SDO_GEOMETRY;           -- Vazio de Atendimento
UNIAO_BUFF MDSYS.SDO_GEOMETRY;      -- União dos Buffers

BEGIN

/* Corpo da Procedure */

SELECT L.GEOM
INTO LIMITE
FROM LIMITE_SP L;

SELECT B.GEOM
INTO UNIAO_BUFF
FROM TAB_UNIAO_BUF B;

VAZIO:= SDO_GEOM.SDO_DIFFERENCE(LIMITE, UNIAO_BUFF, 0.005);

AREA_VAZIO := SDO_GEOM.SDO_AREA(VAZIO,0.005);
```

```

INSERT INTO TAB_VAZIOS VALUES(VAZIO,AREA_VAZIO);
COMMIT;

END;

END;

```

Resultado: Vazio de Atendimento (em branco)

Área Município ~ 1.524 Km²
 Área Buffers ~ 290 km²
 Área Vazio ~ 1.234 Km² (não considerando as interseções)

Solução SPRING (Caso 1/ Parte 2)

O SPRING oferece a função “Recortar Plano de Informação”, que permite efetuar a operação de diferença entre dois planos de informação. No entanto, utilizando-se a função através do Menu Ferramentas do SPRING, ao definir a máscara, não foi possível selecionar todos os “buffers” resultantes da união, num único passo, pois a função seleciona apenas o último polígono para fazer o recorte sobre o PI ativo. Desse modo, para se concluir a operação, sucessivos recortes teriam de ser efetuados.

Assim, optou-se por implementar um procedimento em LEGAL (INPE, 2002) que executasse o recorte, como segue:

```

{
// Este procedimento subtrai a área do Município pela área formada pela união dos "buffers"
// e guarda o resultado no PI "limSP_VazioAtendimento".

```

```

// Declaração de Variáveis
Tematico Plmascara("Buffers_Distancia"),
    Plentrada ("EstruturaUrbana_tem"),
    Plsaida("EstruturaUrbana_tem");

//Instanciação das Variáveis
Plmascara = Recupere(Nome="Buff");
Plentrada = Recupere(Nome="limite_SP");
Plsaida = Novo(Nome="limSP_VazioAtendimento", ResX=20, ResY=20, Escala=7500);

//Operação
Plsaida = Plmascara.Classe != "Uniao" ? Plentrada : Plsaida ;

}

```

Para a execução do procedimento acima, foi preciso converter o PI correspondente ao Limite do Município ('Limite-SP') para o formato matricial. O resultado gerado, também está no formato matricial, que pode posteriormente ser convertido para o formato vetorial, caso necessário.

Resultado: Área de Vazio de Atendimento (em azul)

Considerações do Caso 1/Parte 2

Para este caso, simplificou-se a solução fazendo a subtração do limite do Município pela União dos “buffers” de cada Unidade de Saúde.

A operação de subtração de camadas vetoriais no ORACLE SPATIAL é bem simples, mas foi necessário fazer a união dos “buffers” antes de executar a operação.

No SPRING, para usufruir da união dos “buffers” já existente, foi preciso utilizar os recursos da LEGAL e o PI correspondente ao Limite do Município foi necessariamente convertido para o formato matricial. O fato do SPRING trabalhar com dados matriciais, possibilita se chegar aos resultados de outra forma.

6.2.2 Identificação dos Distritos que são servidos pelo Metrô (Caso 2)

A localização dos Distritos que são servidos por meios de transporte de massa, regiões de mais fácil acesso à população, é importante, do ponto de vista administrativo e operacional, para, por exemplo instalar nestes Distritos, Unidades de Saúde especializadas. As várias especialidades médicas existentes, cada uma exigindo equipamentos e profissionais específicos, torna inviável economicamente a manutenção destas especialidades em todas as Unidades de Saúde. Assim, neste exemplo, a consulta envolve selecionar os Distritos que são cortados pela Linha Vermelha do Metrô. Esta consulta testa se há algum tipo de interação espacial entre a Linha Vermelha do Metrô e os Distritos.

Para o teste do Caso 2, serão utilizadas as Linhas do Metrô disponíveis, juntamente com os Distritos, conforme mostra a Figura 38.

Fig. 38 – Município de São Paulo: Linhas do Metrô e Distritos

Fonte: Fundação do Desenvolvimento Administrativo (FUNDAP)

Solução ORACLE SPATIAL (Caso2)

No ORACLE SPATIAL foi utilizado o relacionamento 'ANYINTERACT' para testar se há algum tipo de interação entre a Linha do Metrô e os Distritos, conforme as consultas SQL abaixo, sendo o segundo modo sem a utilização de índice espacial:

```
SELECT d.CodDist
FROM distrito d, metro m
WHERE m.Linha = 'Vermelha'
      AND MDSYS.SDO_RELATE (d.geom, m.geom,'mask=ANYINTERACT
      querytype=WINDOW') = 'TRUE';
```

ou

```
SELECT d.CodDist
FROM distrito d, metro m
WHERE m.Linha='Vermelha'
      AND SDO_GEOM.RELATE (d.geom, 'ANYINTERACT', m.geom,0.005)='TRUE';
```

Resultado:

```
CODDIST
```

```
-----
54
8
82
60
93
5
36
6
70
67
80
10
```

12 rows selected.

Solução SPRING (Caso 2)

O Caso 2 utiliza a função INTERCEPTA da opção Consulta Espacial, em Visualização de Objetos, disponível para Planos de Informação Cadastrais.

Primeiramente foi feita uma consulta para selecionar a Linha Vermelha do Metrô, que foi armazenada na coleção Linha1. Assim, testa-se a interseção de todos os Distritos com a Linha Vermelha do Metrô, armazenada na coleção Linha1.

Resultado:

Uma variação do Caso 2 seria envolver dados alfanuméricos na consulta espacial, por exemplo efetuando a consulta “Selecionar os distritos com população maior que 50.000 habitantes que se interceptam com as Linhas do Metrô”. Foi visto no Caso1 que no ORACLE SPATIAL essa consulta é bem simples, bastando colocar mais uma condição na lista SELECT. No SPRING, que tem Arquitetura Dual, primeiro é necessário selecionar os distritos com população acima de 50.000, processo realizado na consulta não espacial (conforme Figura 39), e posteriormente selecionar, desse conjunto resultante, através da consulta espacial, aqueles distritos que se interceptam com as linhas do Metrô.

Fig. 39 – Seleção dos Distritos com população acima de 50.000

A seguir, procede-se com a consulta espacial sobre a seleção feita anteriormente. O resultado da seleção espacial é apresentado na Figura 40.

Fig. 40 – Seleção dos Distritos com população acima de 50.000 habitantes e que se interceptam com as linhas do Metrô

Considerações do Caso 2

A consulta foi efetuada em ambas as ferramentas com certa facilidade. Do mesmo modo que no caso anterior, o ORACLE SPATIAL não gerou uma visão espacial dos resultados, mas, devido ao pequeno número de Distritos resultantes da seleção, foi possível verificar que o resultado foi satisfatório, sem a necessidade de visualização espacial.

No SPRING, explorou-se a interface de Consulta Espacial, mais especificamente o relacionamento INTERCEPTA. É importante observar que qualquer consulta demanda a pré-seleção de uma coleção de objetos de um dos Planos a ser testado, para que posteriormente essa coleção possa ser comparada com o outro Plano de Informação (ou outra coleção), salvo em casos de utilização de todos os objetos do PI a ser consultado.

Aproveitou-se o exemplo para testar a consulta espacial no SPRING, em conjunto com os atributos. Nota-se que foi possível utilizar a mesma interface de consulta para executar a seleção não espacial e posteriormente a seleção espacial. A única observação é que, na Arquitetura Dual, esta seleção não é possível num único processo.

6.2.3 Análise de proximidade de Equipamentos Urbanos em relação aos meios de transporte (Caso 3)

Suponha que se queira determinar quais Unidades de Saúde podem ser transformadas em Unidades especializadas, levando-se em consideração a distância a ser percorrida pelos usuários, do meio de transporte até a Unidade de Atendimento. Assim, neste exemplo, deseja-se saber quais Unidades de Saúde estão a 2km da Linha Vermelha do Metrô.

Existem vários métodos para se medir a proximidade entre objetos espaciais. Neste caso, será utilizado o método de geração de “buffers” de distância ao redor do objeto testado, no caso, a Linha Vermelha do Metrô e, posteriormente, serão selecionadas as Unidades de Saúde que estiverem dentro desse “buffer”. Se as estações do Metrô estivessem mapeadas e disponíveis, esses “buffers” poderiam ser gerados ao redor de cada estação do Metrô, ao invés de utilizar a Linha do Metrô, o que resultaria numa melhor precisão.

Para este teste serão utilizadas as Linhas do Metrô disponíveis, juntamente com as Unidades de Saúde, conforme mostra a Figura 41.

Fig. 41 – Município de São Paulo: Linhas do Metrô e Unidades de Saúde.

Fonte: Fundação do Desenvolvimento Administrativo (FUNDAP)

Solução ORACLE SPATIAL (Caso 3)

A solução deste problema no ORACLE SPATIAL, pode ser efetuada utilizando-se a função `SDO_GEOM.SDO_BUFFER` ou utilizando-se o operador `SDO_WITHIN_DISTANCE` (ou ainda, a função `SDO_GEOM.WITHIN_DISTANCE`), sendo que no segundo caso, o “buffer” é gerado implicitamente na execução da operação.

Utilizando-se a função `SDO_GEOM.SDO_BUFFER`, inicialmente é necessário elaborar uma consulta para a geração do “buffer” de 2km ao redor da Linha Vermelha do Metrô. Este procedimento é efetuado através do seguinte comando SQL:

```
SELECT SDO_GEOM.SDO_BUFFER(m.geom,2000,0.05)
FROM metro m
WHERE m.Linha='Vermelha';
```

O “buffer” de 2km gerado para a Linha Vermelha do Metrô é apresentado na Figura 42.

Fig. 42 – “Buffer” de 2 Km gerado ao redor da Linha Vermelha do Metrô

Para calcular quais Unidades de Saúde estão dentro deste “buffer” de 2km foi utilizado o relacionamento ‘INSIDE’ do operador MDSYS.SDO_RELATE. Este procedimento está apresentado na consulta SQL abaixo, onde o parâmetro *geometry2* (em azul) corresponde ao resultado do SELECT anterior (ou seja, ao “buffer” de 2km):

```
SELECT s.Id11
FROM saudetp1 s
WHERE MDSYS.SDO_RELATE (s.geom,
  (SELECT SDO_GEOM.SDO_BUFFER(m.geom,2000,0.05)
   FROM metro m
   WHERE m.Linha='Vermelha'
  ),
  'mask=INSIDE querytype=WINDOW') = 'TRUE';
```

Resultado:

ID11	ID11	ID11	ID11
201	482	312	872
627	57	31	766
628	487	27	70
207	63	36	112
34	780	42	69
210	851	48	
53	892	52	
54	899	449	
506	227	451	
265	511	37	
73	307	454	

38 linhas selecionadas.

A solução do problema utilizando-se o operador `SDO_WITHIN_DISTANCE`, que produz o mesmo resultado anterior, é dada pelo seguinte comando SQL:

```
SELECT s.ID11
FROM saudetp1 s, metro m
WHERE m.linha = 'Vermelha' and
MDSYS.SDO_WITHIN_DISTANCE (s.geom,m.geom,
                             'distance=2000 layer_gtype=POINT') = 'TRUE';
```

O operador `SDO_WITHIN_DISTANCE` carece de um índice espacial para o parâmetro *geometry1*. Uma vez que as Unidades de Saúde estão representadas como pontos, e o índice *Quad-tree* é mais indicado neste caso, foi gerado o seguinte índice *Quad-tree* (com 8 níveis) para o layer 'saudetp1' :

```
/*
Criando um índice Quad-tree para as Unidades de Saúde
*/

CREATE INDEX saudetp1_idx_quad
ON saudetp1 (geom)
INDEXTYPE IS MDSYS.SPATIAL_INDEX
PARAMETERS ('SDO_LEVEL=8');
```

O parâmetro 'layer_gtype=POINT', no SELECT acima, reafirma a utilização do índice *Quad-tree*, através da especificação do tipo de geometria no *layer*.

Solução SPRING (Caso 3)

No SPRING, o “buffer” de 2 Km pode ser gerado através da função Mapa de Distância, conforme apresentado no Caso 1. Para verificar quais Unidades de Saúde estão dentro desse “buffer” gerado, isto é, num raio de 2km da Linha Vermelha do Metrô, utilizou-se o relacionamento topológico ‘DENTRO’ da Consulta Espacial, em Visualização de Objetos. O teste topológico, no entanto, só pode ser efetuado entre Planos de Informação do tipo Cadastral. O “buffer” gerado pertence a uma categoria Temática. O problema foi solucionado criando-se uma Categoria Cadastral para armazenar o “buffer” gerado. Prosseguiu-se com a geração da consulta, selecionando-se as Unidades de Saúde que estão dentro do “buffer”.

Resultado:

Como complemento desta solução no SPRING, visando explorar a função Mapa de Distância quando a saída é do tipo Numérico, foram geradas a grade de distância (de 200x200m) e a imagem de distância correspondente. A grade de distância e a imagem de distância foram geradas a partir da Linha Vermelha do Metrô, cujos resultados são apresentados a seguir, nas Figuras 43 e 44, respectivamente. Posteriormente, foi feito um fatiamento gerando-se um PI Temático com os “buffers”

de 2km (na cor rosa), 4km (na cor azul) e 6km (na cor verde), conforme mostra a Figura 45.

Fig. 43 – Grade de distâncias (200x200m) gerada a partir da Linha Vermelha do Metrô

Fig. 44 – Imagem correspondente à grade de distâncias (200x200m)

Fig. 45 – Fatiamento da grade de distâncias (200x200m) do PI Numérico

Tanto a grade de distância quanto a imagem podem ser utilizadas como mais um meio de suporte nesta análise de proximidade das Unidades de Saúde em relação aos meios de transporte.

Considerações do Caso 3

A consulta foi atendida por ambas as ferramentas. No entanto, no SPRING, a consulta foi mais trabalhosa, já que a ferramenta Consulta Espacial só está disponível para o PI Cadastral. A consulta exigiu a geração de um “buffer” ao redor da Linha Vermelha do Metrô, que no SPRING é efetuada pela função Mapa de Distância. Esta função gera um resultado Temático/Numérico, o que tornou necessário converter o “buffer” resultante para o PI Cadastral. Esta necessidade de conversão entre Planos de Informação no SPRING, pode confundir o usuário e implica em um aumento de processos na execução de uma consulta espacial. Tornar-se-ia salutar uma maior integração entre as funcionalidades da ferramenta e os Planos de Informação, bem como uma maior transparência em relação ao formato do dado espacial para com o usuário.

A geração da matriz de distância e da imagem de distância no SPRING, pode ser útil em vários tipos de aplicação, por exemplo, a utilização desta imagem para cruzamento de dados utilizando a LEGAL. Em geral, a combinação de informações

no formato matricial é mais fácil que no formato vetorial.

No ORACLE SPATIAL, explorou-se a utilização do operador SDO_WITHIN_DISTANCE como ferramenta de análise de proximidade. Observe que, em comparação com a função SDO_GEOM.SDO_BUFFER em conjunto com o operador SDO_RELATE, o operador SDO_WITHIN_DISTANCE elimina o passo de geração do “buffer” (que é gerado implicitamente), simplificando a consulta e também permitindo a utilização de índices. Uma vez que os dados eram pontuais, foi utilizado o índice *Quad-tree*.

6.2.4 Identificação dos Equipamentos pertencentes a um determinado Distrito Municipal (Caso 4)

Nos estudos sobre a localização de Unidades de Saúde, do ponto de vista administrativo, existe a necessidade de se conhecer a distribuição espacial e a quantidade de Unidades de Saúde por Distrito. Este conhecimento é importante, por exemplo, para a determinação do planejamento orçamentário, ou seja, através da quantidade de Unidades e suas características são determinados os gastos em manutenção e as necessidades de investimento.

Neste caso, deseja-se selecionar as Unidades de Saúde que pertençam ao Distrito cujo código é 53, conforme ilustra a Figura 46.

Fig. 46 – Unidades de Saúde do Distrito 53

Solução ORACLE SPATIAL (Caso 4)

No ORACLE SPATIAL foi utilizado o relacionamento 'INSIDE' para verificar quais Unidades de Saúde estão dentro do Distrito 53. A consulta SQL expressa abaixo é idêntica à consulta do Caso 3, alterando-se apenas os objetos analisados (a consulta anterior analisava as Unidades de Saúde que estivessem dentro do “buffer” de 2km, esta consulta analisa as Unidades de Saúde que estão dentro de um determinado Distrito):

```
SELECT s.id11
FROM saudetp1 s, distrito d
WHERE d.CodDist=53
      AND MDSYS.SDO_RELATE (s.geom, d.geom , 'MASK=INSIDE
                              QUERYTYPE=WINDOW')='TRUE';
```

Resultado:

```
ID11
-----
470
472
479
```

Solução SPRING (Caso 4)

O procedimento no SPRING, assim como no ORACLE SPATIAL, é parecido com o do Caso 3, onde foi utilizado o relacionamento DENTRO. A diferença está nos objetos analisados, neste caso utilizou-se o Distrito ao invés de um “buffer”. Assim, o primeiro passo é gerar uma coleção contendo o Distrito de código 53. Gerada a coleção, procedeu-se à consulta espacial, escolhendo-se as Unidades de Saúde que estavam dentro do Distrito 53.

Resultado:

Considerações do Caso 4

Este é um exemplo simples envolvendo uma consulta espacial que busca objetos com representações pontuais dentro de um determinado objeto representado por um polígono. Observou-se que as duas ferramentas executaram essa operação sem apresentarem problemas.

6.2.5 Identificação dos Distritos adjacentes a um Determinado Distrito (Caso 5)

Neste caso, deseja-se selecionar quais os distritos que fazem fronteira com o Distrito cujo código é 49, conforme ilustra Figura 47.

Fig. 47 – Unidades de Saúde do Distrito 53.

Fonte: Fundação do Desenvolvimento Administrativo (FUNDAP)

Solução ORACLE SPATIAL (Caso 5)

No ORACLE SPATIAL foi utilizado o relacionamento ‘TOUCH’ para verificar os Distritos adjacentes ao Distrito 49, conforme a consulta SQL abaixo:

```
SELECT d2.CodDist
FROM distrito d1, distrito d2
WHERE d1.CodDist = 49
and mdsys.sdo_relate (d1.geom, d2.geom,
'mask=TOUCH querytype=WINDOW') = 'TRUE';
```

Resultado:

```
CODDIST
-----
80
7
14
92
67
```

Solução SPRING (Caso 5)

No SPRING foi utilizado o relacionamento TOCA. Assim como no caso anterior, o primeiro passo é gerar uma coleção contendo o Distrito de código 49. Assim, tomou-se a coleção já gerada anteriormente e procede-se à consulta espacial, selecionando-se os Distritos que ‘tocam’ o Distrito 49.

Resultado:

Considerações do Caso 5

Este é um exemplo simples envolvendo uma consulta espacial que testa a adjacência entre objetos do mesmo *layer*. Observa-se que no SPRING, a consulta espacial é feita utilizando-se o relacionamento topológico TOCA, que verifica a adjacência topologicamente, sem a necessidade de executar operações de geometria computacional, uma vez que são consultas no mesmo plano de informação. No ORACLE SPATIAL, que não armazena a topologia, os relacionamentos são testados com operações de geometria computacional.

Observa-se que, no SPRING, para que o relacionamento TOCA possa ser testado, a base deve estar topologicamente correta, ou seja, não pode haver linhas duplicadas entre os polígonos vizinhos. Algumas vezes, quando o dado vem de outro sistema (por exemplo o formato 'shapefile' do ArcView), cada polígono armazena todas as suas coordenadas e, desta forma, as coordenadas das fronteiras entre os vizinhos são duplicadas. Portanto, foi necessário corrigir a topologia dos Distritos, através de edição vetorial da base, antes de efetuar este teste.

6.2.6 Consultas Direcionais (Caso 6)

A seguir, apresentaremos um exemplo de consulta direcional no SPRING, cuja

funcionalidade não está implementada no ORACLE SPATIAL.

O SPRING implementa o operador direcional para determinar os objetos que estão a Norte, Sul, Leste, Oeste, Nordeste, Noroeste, Sudeste, Sudoeste e Centro, sendo possível escolher mais de uma opção na mesma consulta.

A Figura 48 mostra um exemplo para cada operador do caso de seleccionar os Distritos que estão a norte da Linha Vermelha do Metrô.

Fig. 48 – Exemplo do Operador Direcional “Norte”

7 CONCLUSÕES E CONSTATAÇÕES

Este trabalho abordou os conceitos fundamentais acerca dos Sistemas Gerenciadores de Banco de Dados Geográficos, abordando principalmente os seguintes tópicos:

- Tipos de dados e operadores espaciais;
- Linguagens de consulta espacial; e
- Otimização das consultas espaciais através de índices espaciais.

Nos testes práticos, analisando o comportamento das aplicações (de algumas operações) nas Arquiteturas Dual e Integrada, nas ferramentas ORACLE SPATIAL e SPRING, observou-se algumas diferenças que serão apresentadas a seguir.

Em relação aos tipos de dados suportados, o ORACLE SPATIAL só manipula dados vetoriais. O SPRING manipula tanto dados vetoriais como matriciais. O que se percebe é que as extensões espaciais na maioria dos casos tratam de dados apenas vetoriais. No futuro estes bancos objeto-relacionais deverão suportar tipos de dados de imagens também.

A semântica dos operadores espaciais é semelhante, visto que ambos os softwares estão baseados no The Dimensionally Extended 9 Intersection Model (DE9IM). No entanto, algumas diferenças podem ocorrer, por exemplo, o operador INTERCEPTA do SPRING verifica e calcula a interseção entre dois objetos. No ORACLE SPATIAL, o parâmetro ANYINTERACT do operador SDO_RELATE (ou da função SDO_GEOM.SDO_RELATE) verifica a interseção entre duas geometrias, e a função SDO_GEOM.SDO_INTERSECTION calcula a interseção entre duas geometrias. Além dos operadores topológicos e métricos, diferentemente do ORACLE SPATIAL, o SPRING também possui o operador direcional.

Em relação às consultas, as diferenças vão desde a formulação até a execução das mesmas. A Arquitetura Integrada do ORACLE SPATIAL, juntamente com o aprimoramento da Linguagem SQL, possibilitando a interpretação das operações espaciais, resultou em grande flexibilidade na elaboração e execução das consultas. A flexibilidade significa que os dados estão armazenados no banco de dados e são facilmente acessíveis pelo administrador do banco ou pelo desenvolvedor da aplicação, através de uma Linguagem de Consulta padrão como a SQL. Os dados espaciais são tratados da mesma forma que os dados não espaciais. Assim, o usuário pode livremente elaborar as suas consultas, a partir das operações e funções pré-

definidas, sem estar atrelado a um conjunto de passos a serem executados via Menu. No SPRING, a LEGAL (Linguagem Espacial para Geoprocessamento Algébrico) permite ao usuário fazer operações com os dados espaciais, que vão além das funcionalidades disponíveis nos menus. No entanto, para o PI Cadastral, apesar dos operadores topológicos (direcionais e métricos) já estarem implementados, ainda não existem interpretadores de modo que esses operadores sejam utilizados num programa em LEGAL.

O ORACLE SPATIAL se diferencia dos softwares SIG tradicionais no sentido de que, nestes últimos, ao passo que as consultas espaciais vão sendo efetuadas (via Menus) os resultados já vão sendo produzidos e apresentados na tela do monitor. No ORACLE SPATIAL, as consultas são geradas via SQL*PLUS ou SQL*PLUS Worksheet, que são Interfaces de Linhas de comando do ORACLE e que não estão associadas a uma tela onde os resultados espaciais aparecem, pode-se ver apenas as coordenadas desses objetos. A Linguagem SQL não é uma linguagem de visualização de dados. Como não era objetivo deste trabalho explorar os visualizadores compatíveis com o ORACLE SPATIAL, os resultados das consultas foram armazenados em tabelas espaciais, como se pertencessem a uma outra camada espacial, e então podiam ser visualizados utilizando-se o Enterprise Manager Console, juntamente com os dados da base de dados. Mesmo assim, essa ferramenta de visualização é muito limitada e não apropriada para aplicações práticas de visualização de dados geográficos.

Nota-se também que não se consegue fazer muita coisa apenas digitando-se um comando SQL na linha de comando do SQL*PLUS, por exemplo. Para se produzir resultados ou combinar comandos SQLs, foi necessário utilizar os recursos de implementação da PL/SQL, a Linguagem de Programação nativa do ORACLE. No entanto, a PL/SQL também tem as suas limitações. A PL/SQL não é voltada para o desenvolvimento de aplicações, e sim para acesso a banco de dados (tabelas), não possuindo, por exemplo, estruturas de dados pré-definidas. A PL/SQL também não permite a criação de classes e não trata problemas recursivos com a facilidade de outras Linguagens de Programação, tal como a Linguagem C. Problemas mais complexos demandam a utilização de Linguagens de Programação mais completas em conjunto com a PL/SQL e com a SQL. O ORACLE SPATIAL é compatível com

o C ou C++.

Observa-se ainda que as consultas são facilmente construídas quando apenas um par de geometrias está envolvido na consulta. No entanto, quando mais de duas geometrias precisam ser analisadas, a solução fica mais complexa. Um exemplo seria resolver o problema de “gerar grupos de geometrias que se interceptam no espaço”. Tomamos esse problema como bastante complexo e recursivo, sendo que em um grupo, pode haver interseção entre mais de dois objetos espaciais. Se fôssemos dar prosseguimento ao Caso 1, depararíamos com este problema, ao tentar agrupar os “buffers” que se interceptam no espaço. Devido às interseções entre os “buffers”, significando redundância de áreas de atendimento, a cada grupo deveria ser inserida uma medida corretiva diferenciada, para recalcular a área de atendimento dos equipamentos a partir dos quais os “buffers” foram gerados. O ORACLE SPATIAL permite fazer a união dos “buffers”, mas quando se quer fazer a união apenas daqueles que se interceptam e associar a cada grupo um identificador, bem como todos os elementos (“buffers”) pertencentes ao grupo, a solução necessita dos recursos de uma Linguagem de Programação mais completa que a PL/SQL, que permita utilização de recursividade e estruturas de dados.

O ORACLE SPATIAL possui um conjunto de operadores e funções para análises espaciais, sendo os operadores mais eficientes, devido à utilização dos índices. Algumas operações como cálculo de área, por exemplo, só são possíveis com a utilização das funções, e assim, não utilizam índices. No SPRING, as operações espaciais topológicas são possíveis para os PIs Cadastral e Rede, mas neste trabalho foram explorados apenas elementos mapeados para o PI Cadastral.

Ambas as ferramentas suportam as consultas utilizadas nos testes, porém, com as restrições apresentadas no Capítulo 6. Observa-se que, no SPRING, não é possível fazer uma consulta envolvendo o objeto espacial e o atributo ao mesmo tempo, devido à Arquitetura Dual. Mas, apesar disso, no Caso 2, vimos que foi possível utilizar a mesma interface de consulta para executar a seleção espacial e posteriormente a seleção não espacial. Ainda no SPRING, as sucessivas conversões entre Planos de Informação, efetuadas em algumas consultas, podem confundir o usuário e implicam em aumento de processos na execução de uma consulta espacial. Isto se deve a uma maior complexidade do Modelo Conceitual do SPRING, que

trabalha com dados de Categoria Temática e Cadastral. Além disso, seria conveniente que os processos de consulta estritamente espacial pudessem ser aplicados a dados de Categoria Temática, o que evitaria processos de conversão entre planos. Nas operações envolvendo teste de adjacência entre polígonos, diferentemente do ORACLE SPATIAL, o SPRING testa a adjacência topologicamente, o que implica numa base de dados topologicamente correta, isto é, seguindo o Modelo Topológico, não podendo haver linhas duplicadas entre os polígonos.

Quanto ao processo de otimização das consultas, o trabalho abordou apenas algumas considerações em relação à utilização ou não de índices. No ORACLE SPATIAL, é possível escolher o índice a ser criado para as geometrias (*R-tree* ou *Quad-tree*). O SPRING cria o índice automaticamente para os objetos espaciais, de acordo com o tipo de representação dos mesmos (*Kd-tree* para pontos e *R-tree* para linhas e polígonos). O índice é criado internamente, de modo que o usuário não percebe este processo.

Ainda sobre os índices, é importante notar que, quando se têm muitas atualizações na base de dados, a performance dos mesmos começa a degradar (sobretudo para o índice *R-tree*) e às vezes é conveniente reconstruí-los. Em ambos os *softwares* analisados, essa operação é possível.

Os bancos de dados convencionais atuais estão provendo capacidade de armazenamento e representação de dados geográficos no mesmo ambiente, utilização de índices espaciais nas consultas, e ainda estenderam a Linguagem SQL para suportar operadores e funções espaciais. Tudo isso cria um ambiente integrado que facilita o processo de manipulação dos dados geográficos. A tendência atual das Arquiteturas de Banco de Dados Geográficos é a passagem para esse ambiente objeto-relacional, o que permite o uso dos recursos de gerenciamento de banco de dados, tais como: controle de transação, segurança, acesso multi-usuário, dentre outros.

REFERÊNCIAS BIBLIOGRÁFICAS

ARAÚJO JR, N.S. Bancos de dados: passado, presente e futuro. **Developers' Cio Magazine**, n. 47, p. 12-14, jul. 2000.

BARBOSA, E.J.Q. **Estudo metodológico para a localização de equipamentos sociais utilizando técnicas de geoprocessamento**. São Paulo, 1999. 217p. Dissertação (Mestrado) – Escola Politécnica, Universidade de São Paulo.

BORGES, C. **Modelos de dados geográficos: uma extensão do modelo OMT para aplicações geográficas**. Belo Horizonte, 1997. 139 p. Dissertação (Mestrado) - Escola do Governo, Fundação João Pinheiro.

CÂMARA, G. **Modelos, linguagens e arquiteturas para bancos de dados geográficos**. São José dos Campos, 1995. 264p. Tese (Doutorado) - Instituto de Pesquisas Espaciais.

CIFERRI, R. R.; SALGADO, A. C. Análise da eficiência de métodos de acesso espaciais em termos da distribuição espacial dos dados. In: WORKSHOP BRASILEIRO DE GEOINFORMÁTICA (GEOINFO 2001), 3., Rio de Janeiro, 2001. **Anais**. Local: Instituto Militar de Engenharia, 2001. Disponível em: <<http://www.dcc.unicamp.br/~cmbm/geoinfo/index.html>>

CLEMENTINI, E.; FELICE, P.; OSSTROM, P. A small set of formal topological relationships suitable for end-user interaction. In: INTERNATIONAL SYMPOSIUM (SSD'93), 3., Singapore, Jun. 1993. **Proceedings**. Singapore: Springer, 1993. Lecture Notes in Computer Science, v. 692, ISBN 3-540-56869-7, p. 277-295.

CODD, E.F. **The relational model for database management**. Version 2. New York: Addison-Wesley Publishing, 1990.

DATE, C.J. **An introduction to database systems**. 6. ed. Massachusetts: Addison Wesley, 1995.

DOLTON, L.M. **Prospecting spatial database offerings**. Geospatial Solutions,

Recent Issues link. Disponível em:

<<http://www.geoinfosystems.com/1001/1001net.html>>. Acesso em: Nov. 2001.

EASTMAN, J.R. **Guide to GIS and Image Processing**: Volume 1. Worcester, MA:

Clark University. Aug. 1999.

EGENHOFER, M. J.; et al. Progress in computational methods for representing geographic concepts. **International Journal of Geographical Information Science**, v. 13, n. 8, p. 775-796, 1999.

EGENHOFER, M.F. Object-oriented GISs: the principles. In: SIMPÓSIO

BRASILEIRO DE GEOPROCESSAMENTO, 3., São Paulo, 1995. **Anais**. Local: São Paulo, jul. 1995, p. 175-203.

EGENHOFER, M.J.; HERRING, J. **Categorizing binary topological relationships between regions, lines, and points in geographic databases**. Orono, ME:

Department of Surveying Engineering, University of Maine, 1991. (Technical Report).

ELMASRI, R.; NAVATHE, S.B. **Fundamentals of database systems**. 3. ed.

Massachusetts: Addison-Wesley, 2000.

ENVIRONMENTAL SYSTEMS RESEARCH INSTITUTE (ESRI). **ARC/INFO**

data management: concepts, data models, database design, and storage. United States of America: ESRI, 1994.

FILGUEIRAS, E.Q. Híbridos: ORDBMS são alternativas viáveis para OODBMS?

Developers' Cio Magazine, n. 47, p. 16-18, jul. 2000.

FRANK, A.U., GOODCHILD, M.F. **Two perspectives on geographical data modeling**. Santa Barbara, CA: National Center for Geographic Information and

Analysis, 1990. (Technical Report 90-11).

Geographic information systems – office of research – pathbreakers – published November 1996 – University of Washington. Disponível em:
 <<http://washington.edu/research/pathbreakers/1960.htm>>. Acesso em Outubro 2001.

GOMES, J. M.; VELHO, L. **Computação visual: imagens**. Rio de Janeiro, SBM, 1995.

GOODCHILD, M.F.; KEMP, K.K. **NCGIA core curriculum in GIS**. Santa Barbara, CA: National Center for Geographic Information and Analysis, University of California. Unite 12, Unit43 and Unit44. 1990. Compiled with assistance from Gerald White, California State University, Sacramento. Disponível em:
 <<http://www.ncgia.ucsb.edu/education/curricula/cctp/Welcome.html>>.

GUIDARA JR., P. **Aspectos peculiares em projetos de dados geográficos com ênfase no desenvolvimento de metadados**. São Paulo, 2000. 163 p. Dissertação (Mestrado) - Escola Politécnica, Universidade de São Paulo.

IBM. DB2 Spatial Extender: **User's guide and reference**. Version 7, 2. ed., June 2001. Disponível em: <<http://www.ibm.com/support/>>. Acesso em: Nov. 2001.

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **Fundamentos de geoprocessamento: tutorial do SPRING**. São José dos Campos: Departamento de Processamento de Imagens, 2000. /no prelo/

INSTITUTO NACIONAL DE PESQUISAS ESPACIAIS (INPE). **Manual do SPRING**. São José dos Campos: Departamento de Processamento de Imagens. Disponível em: <<http://www.dpi.inpe.br/spring/usuario/indice.htm>>. Acesso em Janeiro 2002.

InterSystems do Brasil. Disponível em: <www.intersystems.com.br>. Acesso em: Fev. 2002.

KANTH, K.V.R.; RAVADA, S.;ABUGOV, D. **Quadtree and r-tree indexes in Oracle Spatial: a comparison using GIS data**. ACM SIGMOD 2002, Madison: Wisconsin, Jun. 2002.

LAURINI, R.; THOMPSON, D. **Fundamentals of spatial information systems**. London: Academic Press, 1992.

ORACLE COORPORATION. **Oracle spatial 8.1.6 performance -related**: an Oracle technical white paper, May 2000. Disponível em:
<http://technet.oracle.com/products/spatial/htdocs/Spatial_perf_twp/spatial_perf_twp.htm>. Acesso em: Nov. 2001.

ORACLE COORPORATION. **Oracle spatial discussion forum**. Disponível em:
<<http://otn.oracle.com/forums/database.html> (Spatial)> Acesso em: Jun. 2002.

ORACLE COORPORATION. **Oracle spatial training guide** : release 8.1.7, Jan. 2001. Disponível em <<http://www.oracle.com>>. Acesso em: Nov. 2001.

ORACLE COORPORATION. **Oracle spatial user's guide and reference**: release 9.0.1, June 2001. Part. No. A88805-01. Disponível em: <<http://technet.oracle.com>> e <<http://www.oracle.com>>. Acesso em: Jan. 2002.

PICTURE representation using quad-trees – McGill University: School of Computer Science – Winter 1999 Project 32 for 308-251B. Disponível em:
<<http://www.cs.mcgill.ca/~pcarbo/cs251/>>. Acesso em: Jan. 2002.

PL/SQL user's guide and reference. Release 9.0.1, June 2001. Part. No. A89856-01. Oracle Corporation. Disponível em: <<http://technet.oracle.com>> e <<http://www.oracle.com>>. Acesso em: Marc. 2002.

PROGRAMA de transferência de tecnologia. **Base de Dados**. São Paulo: Laboratório de Geoprocessamento da Escola Politécnica da USP / SABESP. 1998. 1 CD-ROM.

PROGRAMA de transferência de tecnologia. **Carta-imagem**. São Paulo: Laboratório de Geoprocessamento da Escola Politécnica da USP / SABESP. 1998. 2 CD-ROM.

PROGRAMA de transferência de tecnologia. **Material do Curso de Arcview**. São Paulo: Laboratório de Geoprocessamento da Escola Politécnica da USP / SABESP. 1998. /no prelo/.

QUINTANILHA, J.A; et. alii. The use of remote sensing in the monitoring of the metropolitan area in São Bernardo do Campo, Brasil. In: UDMS - 22nd URBAN and REGIONAL DATA MANAGEMENT SYMPOSIUM, Delft , Holanda, 2000. **Anais**. Delft , Holanda, 2000. 1 CD-ROM.

RAMIREZ, M.R.; SOUZA, J. M. **Processamento distribuído da junção espacial**. In: WORKSHOP BRASILEIRO DE GEOINFORMÁTICA (GEOINFO 2001), 3., Rio de Janeiro, 2001. **Anais**. Local: Instituto Militar de Engenharia, Out. 2001. Disponível em: <www.ic.unicamp.br/~cmbm/geoinfo/papers/131milton.pdf>

REEVE, D.E. **apostila on line**. University of Salford. Disponível em: <<http://www.els.salford.ac.uk/geog/staff/nmt/fungis/hsd/hsd5.htm>>. Acesso em: Out. 2001.

RIGAUX, P.; SCHOLL, M.; VOISARD, A. **Spatial databases with application to GIS**. San Francisco: Morgan Kaufman Publishers, 2002.

RUMBAUGH, J.; et al. **Modelagem e projetos baseados em objetos**. 8. ed., Rio de Janeiro: Campus, 1994.

SAMET, H. **Applications of spatial data structures: computer graphics, image processing and GIS**. Massachusetts: Addison-Wesley Publishing, 1989.

SHARMA, J. **Oracle Spatial: an Oracle technical white paper**. May 2001. Disponível em: <<http://www.oracle.com>>. Acesso em Nov. 2001.

SILBERSCHATZ, A.; KORTH, H.F.; SUDARSHAN, S. **Sistema de banco de dados**. 3. ed., São Paulo: Makron Books, 1999.

THOMÉ, R. **Interoperabilidade em geoprocessamento: conversão entre modelos conceituais de sistemas de informação geográfica e comparação com o padrão Open GIS**. São José dos Campos, 1998. 193 p. Dissertação (Mestrado) - Instituto de Pesquisas Espaciais.

TOMLINSON, R.F. Reflections on the revolution: the transition from analogue to digital representations of space, 1958-1988. **Journal of American Congress on Surveying and Mapping**, v. 15, n. 3, p. 249, 1988.

ULLMAN, J.D.; WIDOM, J. **A first course in database systems**. Rio de Janeiro: Prentice-Hall do Brasil, 1997.

WOODSFORD, P.A. **The significance of object-orientation for GIS**. Laser-Scan, Science Park, Milton Road, Cambridge, CB4, 0FY, UK. Disponível em: <<http://www.laser-scan.com/papers/ooforgis.htm>>. Acesso em Out. 2001.

ZHOU, Q. **Online GIS course** – University of New South Wales – Geo3142 – Lecture 4, Revised:07/10/98. Disponível em: <<http://geog.hkbu.edu.hk/Qzone/Teaching>>. Acesso em Out. 2001.

APÊNDICE A - DADOS DISPONÍVEIS

1 - Estrutura Urbana

Órgão Responsável: Prefeitura Municipal de São Paulo

Restrições de acesso e de uso: Disponível para venda em CD-ROM

Extensão Geográfica: Município de São Paulo

Tipo de Documento : MOC - Mapa Oficial da Cidade de São Paulo

Projeção utilizada: UTM

Sistema de Coordenadas: Coordenadas Planas

Datum: Córrego Alegre

Entidades representadas: Linhas do Metrô, Distritos, Equipamentos Urbanos (Unidades de Saúde)

Entidade: Linhas do Metrô

(Geometria: linha)

Entidade: Distrito

Distritos Municipais é a menor unidade territorial de referência para a Prefeitura do Município de São Paulo. O Município está dividido em 96 Distritos, cuja criação e delimitação foram estabelecidas pela Lei nº 11.220 de 20/05/92.

(Geometria: polígono)

Entidade: Equipamentos Urbanos (Unidades de Saúde - Hospitais)

(Geometria: ponto)

2 - Setores Censitários

Órgãos Responsáveis: IBGE

Restrições de acesso e de uso: Disponível na Internet no endereço
<http://www2.ibge.gov.br/pub/Censos/Censo_Demografico_2000/Dados_do_Universo/>

Extensão Geográfica: Município de São Paulo

Formato do Arquivo: XLS (PLANILHA DO EXCEL)

Ano: 1996