UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

Dissertação de Mestrado

DA MODELAGEM CONCEITUAL À REPRESENTAÇÃO LÓGICA DE TRAJETÓRIAS EM SGBDOR E SISTEMAS DE DW

BRUNO DE CARVALHO LEAL

FORTALEZA-CE

2011

UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

BRUNO DE CARVALHO LEAL

DA MODELAGEM CONCEITUAL À REPRESENTAÇÃO LÓGICA DE TRAJETÓRIAS EM SGBDOR E SISTEMAS DE DW

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Área de concentração:Banco de Dados

Orientador: Prof. Dr. José Antônio F. de Macêdo

Co-Orientador: Profa. Dra. Valéria C. Times

FORTALEZA-CE

2011

UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE CIÊNCIAS
DEPARTAMENTO DE COMPUTAÇÃO
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

BRUNO DE CARVALHO LEAL

DA MODELAGEM CONCEITUAL À REPRESENTAÇÃO LÓGICA DE TRAJETÓRIAS EM SGBDOR E SISTEMAS DE DW

Dissertação submetida à Coordenação do Curso de Pós-Graduação em Ciência da Computação da Universidade Federal do Ceará, como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Aprovada em: ___/___/_____

BANCA EXAMINADORA

Prof. Dr. José Antônio F. de Macêdo Universidade Federal do Ceará - UFC Orientador	Prof. Dr. Marco Antônio Casanova Pontifícia Universidade Católica do Rio de Janeiro - PUC-Rio
Profa. Dra. Valéria C. Times Universidade Federal de Pernambuco - UFPE Co-orientadora	Prof. Dr. Fábio André Machado Porto Laboratório Nacional de Computação Científica - LNCC
Profa. Dra. Vânia Maria Ponte Vidal Universidade Federal do Ceará - UFC	

RESUMO

Com o aumento do número de dispositivos móveis equipados com serviços de localização geográfica, tem se tornado cada vez mais economicamente e tecnicamente possível capturar os percursos (i.e. trajetórias) dos objetos móveis. Muitas aplicações interessantes têm sido desenvolvida com intuito de explorar análises de trajetórias de objetos móveis. Por exemplo, em sistemas de gerenciamento de veículos de entrega, pode ser realizado tanto o monitoramento dos veículos quanto análises para apoio a decisões estratégicas. De modo geral, as trajetórias podem ser analisadas em duas perspectivas: tempo real e histórica. Além disso, aplicações de trajetórias compartilham uma necessidade em comum que é o registro mais estruturado do movimento. Isso permite manipular trajetórias como objetos de primeira classe e adicionar qualquer semântica requerida pela aplicação e, também, a criação de métodos robustos e eficientes para agregar conjuntos de trajetórias de forma a permitir a realização de análises complexas. Este trabalho estende um trabalho anterior na modelagem conceitual de trajetórias pela generalização da ideia de paradas e movimentos e pela definição de um conjunto de funções de agregação para trajetórias. Neste trabalho é proposto, ainda, duas abordagens por modelagem, ambas baseadas em meta-esquemas, para elaboração de esquemas de trajetórias para ambiente transacional e multidimensional. Para demonstrar e provar nossas contribuições apresentamos um caso de estudo real sobre trajetórias de caminhões de entrega. Os resultados experimentais demonstram que as abordagens de modelagem oferecem a flexibilidade necessária para lidar com a complexidade da semântica das trajetórias em análises de tempo real e histórica.

Palavras-Chave: Dados de trajetória, Trajetória Semântica, Data warehouse

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Com o aumento do número de dispositivos móveis equipados com serviços de localização geográfica, tem se tornado cada vez mais economicamente e tecnicamente possível capturar os percursos (i.e. trajetórias) dos objetos móveis. Muitas aplicações interessantes têm sido desenvolvida com intuito de explorar análises de trajetórias de objetos móveis. Por exemplo, em sistemas de gerenciamento de tráfego, "engarrafamentos" podem ser descobertos realizando mineração sobre os padrões de movimento de grupos de veículos. Similarmente, na zoologia, a análise de grupos de trajetórias podem ajudar a explicar os padrões de migração. A partir de análises e descoberta de padrões de trajetórias é possível ofertar serviços personalizados aos usuários. Em um sistema de entregas, a aplicação pode se utilizar das trajetórias dos veículos desde o monitoramento em tempo real até a realização de análises que podem fornecer ajuda no processo de tomada de decisões estratégicas, tais como o planejamento das entregas.

Do ponto de vista do gerenciamento dos dados de trajetória, as aplicações requerem: (i) registro mais estruturado dos dados, o qual possibilite a manipulação de trajetórias como entidades de primeira classe e adicionar qualquer semântica necessária para a aplicação e (ii) métodos para modelar e organizar trajetórias multidimensionalmente. Já do ponto de vista da análise de trajetórias, é requerido (iii) análises de trajetórias em tempo real e (iv) análises históricas de trajetórias, para apoiar o processo de tomada de decisão através da computação de agregações e pela visualização delas em diferentes níveis de perspectivas e níveis de detalhes. Estes quatro requisitos são a chave para facilitar o desenvolvimento de aplicações que utilizam dados de trajetórias.

Uma quantidade substancial de pesquisas tem sido conduzidas para prover métodos e sistemas protótipos para armazenamento, consulta e análise de trajetórias de objetos móveis em modelos relacionais e multidimensionais, como por exemplo os trabalhos [15] [37] [30] [29] [43]. Entretanto, nenhuma das abordagens propostas fornece uma solução apropriada para o gerenciamento de trajetórias como objeto de primeira classe em ambientes transacional e multidimensional. Além disso, somente a abordagem de [43] oferece uma visão semântica de trajetória que permite às aplicações associar as semântica desejada às trajetórias. Porém, essa abordagem contempla, apenas, a apli-

cabilidade em esquema transacional. De fato, até onde temos conhecimento, nenhum trabalho foi publicado utilizando trajetórias como objeto semântico em modelagem de dados multidimensionais.

Para preencher a lacuna entre a visão transacional e a visão multidimensional sobre trajetórias, uma nova abordagem tem que ser desenvolvida para permitir que as aplicações realizem consultas tradicionais sobre trajetórias semânticas armazenadas em banco de dados transacionais e computar agregações sobre trajetórias semânticas armazenadas em data warehouses multidimensionais. Deste modo, um conjunto de funções de agregação para trajetórias semânticas é necessário. E, ainda, uma visão semântica de trajetórias é necessária, tanto para o esquema transacional, quanto para o multidimensional. Isso exige que trajetórias sejam tratadas como objetos de primeira classe em ambos os esquemas citados, o que está além da capacidade dos atuais protótipos espaço-temporais. Por esse motivo, nossas contribuições são:

- uma definição formal para trajetória semântica;
- um meta-esquema objeto-relacional para trajetórias semânticas;
- um meta-esquema multidimensional estrela para trajetórias semânticas;
- uma definição para assinatura de funções de agregação para trajetórias semânticas;
- um conjunto de funções de agregação para trajetórias semânticas;
- dois algoritmos de ETL: um para popular o esquema transacional a partir dos dados brutos e outro para popular o esquema multidimensional a partir do esquema objeto-relacional.

O restante do trabalho é organizado como a seguir. No capítulo seguinte apresentamos uma breve discussão sobre os conceitos preliminares e definições necessárias para contextualização e entendimento do restante do trabalho e, também, o estudo de caso real sobre o qual ilustramos as contribuições e aspectos do trabalho. Após isso, no capítulo 3, realizamos um estudo sobre a literatura com foco em três tópicos principais para nosso trabalho: trajetórias semânticas, data warehouse de trajetórias e funções de agregação para trajetórias. O capítulo 4 discute, de forma mais pontual, os requisitos necessários para gerenciamento e análise de dados de trajetórias e, após isso, descreve detalhadamente nossas contribuições (citadas acima), bem como a aplicação destas em nosso estudo de caso. O capítulo seguinte detalha os experimentos realizados em um ambiente de aplicação real para gerenciamento de uma frota de caminhões de entrega. Finalmente, no capítulo 6, expomos nossas conclusões sobre o trabalho e destacamos possibilidades de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Apresentamos, neste capítulo, uma introdução aos conceitos preliminares relacionados com o armazenamento e manipulação dos dados de trajetórias. Após isso destacamos as definições básicas para o modelo de trajetórias que são necessárias para o restante do trabalho e, em seguida, a descrição do estudo de caso de um cenário real de aplicação que utiliza dados de trajetórias e é utilizado ao longo do trabalho para ilustrar os requisitos necessários de análise.

2.1 Dados de trajetórias de objetos móveis

Nesta seção tratamos sobre os dados de mobilidade (i.e. *Movement Data*), que são oriundos, tipicamente, de objetos móveis como carros, pessoas, aeronaves ou animais equipados com GPS. Para isso iremos utilizar como guia quatro perguntas estabelecidas em [3] sobre movimento voltado para o contexto específico de trajetórias: (i) o que é? (ii) como pode ser refletido nos dados? (iii) como pode ser caracterizado? e (iv) do que ele depende?

O que é movimento? É a noção de deslocamento por mudança do posicionamento físico, que é tomada, geralmente, com base em algum sistema de referência para caracterizar suas posições (geralmente o posicionamento por coordenadas no espaço geográfico).

Como o movimento se reflete nos dados? O movimento é capturado por dispositivos de localização geográficas (e.g. dispositivos GPS), porém sem nenhuma estruturação para trajetórias. O que se tem é uma quantidade massiva de dados no formato bruto de tuplas. Por exemplo, um dispositivo GPS que coleta dados em intervalos fixos de 10 segundos, pode gerar ao final do dia dados de uma trajetória contendo 5.760 pontos, e ao longo de um mês 172.800 pontos. Considerando, ainda, que em cenários de mobilidade reais o número de objetos móveis também seja grande, tem-se um aumento grande no volume de dados. Estes dados brutos (referidos na literatura como *raw data*) são obtidos na forma de tuplas, que possuem, de forma geral, um par $P = (id, T)$, no qual id é o

identificador do objeto e $T =< x, y, t >$, onde x e y representam a latitude e longitude no instante de captura da posição do objeto, visto como ponto, e t representa o instante dessa captura. Uma trajetória de um objeto pode ser vista, do ponto de vista dos dados brutos, como uma sequência de tuplas de formato P ordenados por ordem crescente de t. Note, neste caso, que não há uma percepção de uma entidade identificável como trajetória, o que se enxerga é, apenas, uma sequência de dados não processados.

Como uma trajetória pode ser caracterizada a partir do movimento? De acordo com o exposto no parágrafo anterior, podemos caracterizar uma trajetória como a evolução da posição de algum objeto se deslocando em algum espaço durante um dado intervalo de tempo, sendo, portanto, um conceito espaço-temporal [43]. Para o nosso trabalho, consideramos movimento de trajetórias os movimentos que se caracterizam pelo deslocamento do objeto móvel de um ponto a outro para cumprir um objetivo, o que se diferencia de demais objetos espaço-temporais como, por exemplo, os destinados a representar o registro da evolução espacial de uma área (e.g. registro de áreas de desmatamento).

Qual a relação entre movimento e objetos do mundo real? As entidades móveis, assim como os demais objetos espaciais e temporais, têm suas próprias características que, consequentemente, influenciam diretamente sobre seus movimentos. O movimento de um carro, por exemplo, está restrito às vias urbanas e seus sentidos, enquanto o movimento dos pedestres não é influenciado diretamente pelo sentido das vias. Além disso, o movimento pode se relacionar com outras informações e objetos do contexto (e.g. pontos de interesse, região espacial de ocorrência, clima etc.), os quais depende para resposta a consultas analíticas.

Nesta seção, buscamos realizar uma descrição do contexto sobre o qual estão inseridos os dados de mobilidade que consideramos para o escopo de nosso trabalho. Com isso pretendemos deixar bem claro quais tipos de dados de mobilidade são considerados alvos das contribuições deste trabalho. Para isso definimos a origem e o formato dos dados de mobilidade e especificamos a caracterização de uma trajetória de objeto móvel a partir destes dados. Para completar, levantamos uma discussão sobre os aspectos do mundo real que influenciam a trajetória e devem ser considerados na semântica da trajetórias dentro do domínio da aplicação.

2.2 Banco de Dados de Objetos Móveis

Realizar manipulação e análises a partir do conjunto de dados brutos de trajetórias é uma tarefa muito complexa e custosa tendo a disposição apenas uma quantidade massiva de tuplas de dados brutos. Como não se tem estruturação por trajetórias, conseguir formular métodos ou consultas para obter informações como comprimento da trajetória, duração e velocidade média se torna extremamente custoso. Isso acontece, principalmente, devido à simplicidade da forma como os dados se apresentam. Para trabalhar com estes dados de forma simples e eficiente é preciso, então, uma forma de abstrair a movimentação dos objetos móveis (i.e. trajetórias) em entidades identificáveis de primeira classe que possam ser manipuladas em alto nível de forma eficiente. Essa abstração facilita, ainda, a criação de aplicações que manipulam trajetórias de forma integral. Dessa forma, percebe-se a necessidade de uma modelo capaz de representar trajetórias de objetos móveis como objetos de primeira ordem, bem como mecanismos de banco de dados para armazenamento e manipulação desses dados em alto nível usufruindo da expressividade e eficiência das linguagens de consulta e da capacidade para lidar com grandes volumes de dados.

Atualmente os dados dos objetos móveis são armazenados e manipulados em Banco de Dados de Objetos Móveis (*Moving Objects Database* - MOD) que são construídos por meio da extensão da tecnologia de SGBD de forma a oferecer suporte à representação de objetos móveis no banco de dados. Isso é necessário pois os SGBDs clássicos e o modelo relacional possuem limitações ao lidar com os aspectos espaço-temporais de trajetórias tais como a representação de suas formas geométricas ou a representação de sua evolução ao longo do tempo. Essa abordagem, geralmente, se dá em duas etapas: (*i*) criação de tipos de dados, os quais podem ser usados para representar tipos de colunas em um SGBD relacional tradicional, integrar um modelo orientado a objetos ou objeto-relacional; (*ii*) criação de operadores para representar, consultar e manipular os dados espaço-temporais dos objetos móveis [15]. Porém, os MODs atuais apresentam apenas a capacidade de armazenamento e mecanismos para manipular e consultar os dados, não apresentando estruturas e métodos para lidar com estes como objetos de primeira grandeza, dificultando a realização de análises mais complexas e desenvolvimento de aplicações com foco em trajetórias. No MOD proposto por Guting et. al [15] [16], por exemplo, uma trajetória é vista como o retorno de um operador aplicado a um tipo abstrato ponto móvel, o que dificulta a manipulação de medidas e propriedades de forma direta além de não viabilizar a criação de métodos de manipulação próprios para trajetórias. Além disso, não existe nenhuma semântica para representação de trajetórias, onde as mesmas são representadas por todo o registro de movimento do ponto móvel. Para obter o comprimento de uma trajetória, por exemplo, deve-se instanciar um elemento construído a partir de uma consulta

que retorna um tipo abstrato e sobre esse elemento deve ser aplicado o operador para construção de trajetória e, sobre o resultado desse operador deve ser chamada a operação que retorna o tamanho como mostrado para o esquema *flight* na Figura 1.

```
flight
  (airline: string, no:int, from:string, to:string, route:mpoint)
LET route =
  (SELECT r FROM flight WHERE airline = "LH" and no = 257);
length(trajectory(route));
```

Figura 1: Exemplo de consulta e manipulação em MOD [15]

É importante notarmos que, para a representação e consulta sobre os dados de trajetórias, os movimentos em si nem sempre são o foco principal do estudo para análises. Mas sim, a extração ou descoberta de conhecimento sobre as entidades móveis que os geram ou, ainda, o contexto de ambiente onde são realizados [3]. Por exemplo, a trajetória de um veículo e de um pedestre estão limitadas às vias públicas, porém, para o pedestre, o sentido da via não limita o seu movimento por ela. Com isso, percebe-se a importância de um banco de dados para trajetórias apresentar suporte para as características dos objetos móveis que têm influência direta sobre os seus movimentos. Além disso, a representação de características através de atributos variam de acordo com a natureza do objeto móvel e interesse da aplicação. Por exemplo, para a aplicação de uma empresa de entregas, pode ser importante a quantidade de combustível para cálculo da autonomia do veículo, porém essa característica não é interessante para uma aplicação de controle de tráfego. Portanto, é necessário para um MOD conseguir representar, também, as características peculiares de acordo com a natureza do objeto móvel e suas medidas e atributos de interesse.

Percebemos, então, que existe uma lacuna que é preciso ser preenchida para banco de dados de objetos móveis no que diz respeito a trajetórias: é preciso que as aplicações possam definir qualquer semântica necessária às trajetórias, seja através de atributos e métodos ou através de ligações entre as trajetórias e outros objetos da aplicação. Para isso, é necessário que as trajetórias sejam um conceito de primeira classe no modelo de dados, o que está além das capacidades dos atuais protótipos espaço-temporais [43].

2.3 Estudo de caso

A importância de análises sobre dados de trajetórias é decorrente dos seus domínios de aplicação que, geralmente, atuam sobre cenários de grande impacto social e/ou econômico e alguns dos principais interesses em analisar dados de trajetórias são oriundos do setor comercial, principalmente empresas que trabalham com veículos para distribuição

de seus produtos. A análise sobre as trajetórias de entregas permite aos estrategistas da empresa, por exemplo, avaliar percursos e tempos médios de entrega e, com isso, obter estimativas sobre prazos de entrega, garantindo confiabilidade para o cliente. Além disso, a análise pode fornecer informações úteis para controle sobre os caminhões e motoristas: se estão seguindo rotas planejadas, demorando mais que o esperado, realizando operações indevidas etc.

O gerenciamento logístico é uma importante classe de aplicação que pode se aproveitar da análise dos dados de trajetória, particularmente as aplicações que gerenciam a entrega de produtos devido aos altos valores econômicos envolvidos. Para esse tipo de aplicação, as viagens (i.e. trajetórias) podem ser usadas para conferir quando planos de entrega são corretamente executados (exemplo de análise de tempo real) ou para avaliar o desempenho das entregas (exemplo de análise sobre histórico dos dados). Estas análises são a chave para melhorar o conhecimento para muitas questões envolvidas com o gerenciamento logístico, como por exemplo:

- exibir no mapa as viagens que estão ocorrendo em tempo real para monitoramento;
- os planejamentos de entrega estão sendo cumpridos?
- com base no estado atual de uma viagem, qual a estimativa para atendimento de um determinado cliente do planejamento?
- qual a relação de custo-benefício para uma certa viagem em relação à quantidade de produto entregue pela duração?
- com base no histórico das viagens, a análise acima apresentou custo-benefício acima do padrão?

As respostas para questões como estas são fundamentais para suporte aos processos de tomada de decisão em aplicações de gerenciamento de veículos de entrega.

Nosso estudo de caso se dá com dados de mobilidade de caminhões de distribuição de gás. Trata-se de um caso que envolve a análise de rotas de diversos caminhões de abastecimento de gás combustível a vários clientes (cozinhas industriais, restaurantes, condomínios etc.). Neste domínio de aplicação, de acordo com os interesses definidos pela equipe de planejamento, iremos considerar que uma trajetória é dada pelo deslocamento da saída da base (i.e. pátio de abastecimento da empresa) até o retorno. Cada caminhão é equipado com um dispositivo GPS que possui, além da função de enviar os dados de localização e tempo para o sistema de monitoramento, funções adicionais para registro de eventos acionados pelo motorista ou detectados pelo próprio sistema. Esses eventos, tais

como saída da garagem, chegada no cliente, início de atendimento, final de atendimento, parada e retorno à garagem, devem ser diretamente associados às trajetórias.

Os dados brutos de mobilidade desse caso são fornecidos através de tuplas que possuem os seguintes atributos:

- identificador da tupla (PK);
- identificador numérico do veículo;
- marcação de tempo para o instante de captura da posição;
- valor numérico para a latitude;
- valor numérico para longitude;
- velocidade instantânea para o instante de captura da posição;

Enquanto os dados brutos dos eventos são tuplas fornecidas à parte que apresentam os atributos abaixo:

- identificador da tupla(PK);
- identificador numérico do veículo;
- descrição textual do tipo de evento;
- marcação de tempo para o instante do início do evento;
- marcação de tempo para o instante de término do evento;
- valor numérico para a latitude;
- valor numérico para longitude;
- descrição textual dos valores referentes ao evento (e.g. para o evento Excesso de velocidade apresenta a velocidade limite e a velocidade realizada e para evento Atendimento apresenta a quantidade de produto entregue)

Com o objetivo de analisar viagens dos caminhões, a aplicação deve registrar dois tipos de dados:

1. Dados sobre as trajetórias dos caminhões: dados sobre a posição do veículo e ocorrências relacionadas são enviadas em intervalos regulares de tempo para um servidor em formato bruto, os quais são armazenados em um banco de dados convencional (i.e. relacional). Por exemplo, quando o caminhão para os seguintes dados

são registrados: (i) o tipo de parada (atendimento ou outro tipo); (ii) quando não são atendimentos os tipos de paradas devem ser informados pelo motorista, tais como refeição, emergência, descanso, abastecimento etc. Quando o caminhão está em deslocamento, o dispositivo controlador informa a posição, velocidade e direção do veículo.

2. Dados relacionados com as trajetórias: regiões de interesse (e.g. região, estado, cidade etc.), clientes visitados, condições de tráfego (que podem explicar atraso de entregas) etc.

A seguir ilustramos alguns exemplos de consultas executadas no cenário de aplicação descrito ate aqui:

- Qual a localização do caminhão 1103 e quais são os clientes já atendidos?
- Qual a quantidade de viagens realizadas na cidade de São Paulo no primeiro bimestre do ano e a velocidade média destas?
- Qual a estimativa de atendimento para o cliente A para uma dada viagem?
- Qual a média da quantidade de produto entregue por cliente para a base X em cada mês do ano?

No contexto de análises históricas para suporte a decisões, é importante a realização de consultas com resultados agregados de acordo com os diversos níveis já tradicionais nesse contexto, como organizacional e de tempo, além dos demais interesses da aplicação como, por exemplo, em relação a regiões. Em nosso estudo de caso a empresa é dividida em filiais, e as filiais em bases, que por sua vez possuem caminhões e é com base nesses níveis que são requeridas as análises com informações agregadas. Em relação ao tempo, são requeridas análises detalhadas por ano, por mês e por dia e, para as regiões, por região geográfica do país, estado e cidade. Normalmente as análises requeridas apresentam agregações combinando os aspectos de interesse citados. A seguir apresentamos alguns exemplos:

- exibir pelos níveis de base, anual, mensal e de caminhão a quantidade de eventos do tipo Excesso de velocidade;
- exibir a média de duração das viagens para os níveis de região, estado, cidade, anual e mensal;
- exibir para os níveis de estado e cidade, mensal e base para a média da quantidade de produto entregue em relação à duração das viagens;

A partir do formato dos dados brutos que estão à disposição, percebemos, então, a dificuldade de formulação de consultas ou procedimentos para as análises necessárias.

2.4 Trajetória semântica

Nesta seção iremos apresentar as definições necessárias para trajetórias e seus componentes que serão utilizadas no restante deste trabalho.

A partir de um ponto de vista conceitual, uma trajetória deve ser percebida pela aplicação ou pelo projetista do banco como um objeto identificável e gerenciável. Mesmo que a trajetória seja definida como um objeto de primeira classe, as aplicações podem requerer, ainda, que informações sejam relacionadas com segmentos da trajetória, como, por exemplo, para indicar que, ao longo de um intervalo de tempo da trajetória, o caminhão parou para entregar produto para um cliente. Por conseguinte, uma visão conceitual de trajetórias deve deve fornecer mecanismos para lidar com uma trajetória como um objeto de primeira classe, bem como para decompor a trajetória em eventos que apresentam o significado de acontecimentos decorridos durante a execução da trajetória. Com isso, temos, então, dois aspectos que precisam ser considerados:

- aspecto geométrico: é o registro espaço-temporal das posições do objeto móvel percebido como um ponto. É representado por um segmento espaço-temporal do caminho realizado pelo objeto móvel durante todo seu ciclo de vida e pode ser construído a partir do conjunto de pontos dos dados brutos;

- aspecto semântico: representa as características ou interesses da aplicação para a trajetória, ou sub-partes da mesma, voltadas especialmente para realização de análises. Pode ser, por exemplo, um segmento da trajetória onde um veículo de entrega realizava um abastecimento.

Um dispositivo de posicionamento geográfico coleta, em intervalos de tempo regular, a localização geográfica de um objeto móvel (e.g. latitude, longitude) associada com uma marcação de tempo. Considerando apenas um dispositivo ao longo do tempo, são acumuladas triplas (latitude, longitude, instante), as quais representam o movimento do objeto. Entretanto a sequência de triplas coletadas podem, na realidade, representar diferentes trajetórias do objeto. Por exemplo, suponha um caminhão que realizou várias viagens em um dia. Ao final, as triplas capturadas pelo dispositivo devem ser divididas em diferentes trajetórias, cada uma representando uma viagem diferente. Agora vamos supor que o mesmo conjunto de triplas usados pela aplicação perceba uma trajetória como sendo definida pela sequência de triplas capturadas ao longo de um dia. Desta maneira,

percebe-se que a definição do que uma trajetória deve ser depende da aplicação, ou seja, a definição do que é o ponto de início e final de uma trajetória depende de como o contexto de aplicação percebe o movimento dos objetos. Devido a isso o projetista deve fornecer uma função para segmentar o conjunto de dados brutos coletados pelos dispositivos de posicionamento e gerar as trajetórias as quais serão, então, armazenadas em banco de dados e manipuladas pela aplicação. Na literatura, esse processo de segmentação dos dados brutos de trajetórias é conhecido como construção semântica de trajetórias - daí a utilização do termo trajetórias semânticas adotado para o trabalho. Esta tarefa pode incluir outras tarefas, como a correção e limpeza dos dados "sujos".

Para representar os conceitos, iremos apresentar algumas definições.

Um *ponto espaço-temporal* é uma tripla $p = (l, g, t)$, onde l e g são números reais que representam latitude e longitude respectivamente, e t é um número real positivo que representa uma marcação de tempo.

Um *conjunto de dados brutos espaço-temporal* é uma sequência finita de pontos espaço-temporais $R = ((l_1, g_1, t_1), ..., (l_k, g_k, t_k))$ tal que, para todos $i \in [1, k)$, $t_i < t_{i+1}$. Perceba que um conjunto de dados brutos espaço-temporais não possui semântica associada.

Seja A um *atributo de evento*, com valores de um domínio D. Um *evento* sobre A é uma tripla $e = (b, d, a)$, onde $b = (l_b, g_b, t_b)$ e $d = (m_d, h_d, u_d)$ são pontos tais que $t_b \leq u_d$, representam o *begin* and *end points* de e, and a é um atributo com valor de D, representando a *informação do evento*.

Uma *trajetória semântica* sobre um atributo de evento A é uma quádrupla $s = (b, d, R, L)$, onde

- $b = (l_b, g_b, t_b)$ e $d = (m_d, h_d, u_d)$ são pontos tais que $t_b \leq u_d$, representam o *begin* e *end points* of s;
- $R = ((l_1, g_1, t_1), ..., (l_k, g_k, t_k))$ é um conjunto de dados brutos espaço-temporais tal que $t_b = t_1$ e $t_d = t_k$;
- $L = (e_1, ..., e_n)$ é uma sequência finita de eventos sobre A tal que:
  - para $i = [1, n]$, os pontos de início e fim de e_i são pontos em R;
  - para $i = [1, n)$, se $d_i = (m_i, h_i, u_i)$ é o ponto final de e_i e $b_{i+1} = (l_{i+1}, g_{i+1}, t_{i+1})$ é o ponto inicial de e_{i+1}, então $u_i \leq t_{i+1}$.

Note que b and d, os pontos de início e fim de s, são apenas uma definição de conveniência, pois estes podem ser trivialmente definidos a partir de R.

Resumidamente, uma trajetória semântica é uma representação definida pela aplicação a partir da evolução da posição de um objeto móvel (percebido como ponto) durante um dado intervalo de tempo. A aplicação pode extrair (definir) trajetórias semânticas a partir de um conjunto de dados brutos espaço-temporais utilizando uma função que incorpora a semântica do que uma trajetória deve ser para o domínio. A definição de uma trajetória deve incluir uma sequência de eventos, os quais podem ser referenciados e gerenciados individualmente, o que permite serem tratados, também, como objetos de primeira classe que captura semântica adicional.

3 TRABALHOS RELACIONADOS

Os fenômenos espaço-temporais têm sido estudados em várias comunidades de pesquisa. Na comunidade de banco de dados, o trabalho de Guting foi a primeira implementação completa de um banco de dados para lidar com dados de objetos móveis. E, baseado nos trabalhos de Guting [15] e Wolfson [45], Pelekis [37] desenvolveu um framework para representar e consultar objetos móveis sobre a plataforma SGBD Oracle. A comunidade de banco de dados tem focado, principalmente, em técnicas para gerenciamento de objetos móveis considerando os aspectos geométricos e temporais, mas negligenciando o suporte do conceito de trajetórias como objetos de primeira classe com capacidade de representar a semântica do ponto de vista da aplicação.

O principal trabalho de referência para banco de dados de objetos móveis, que teve como precursores alguns dos principais trabalhos nesta área [41] [9], é o trabalho de Guting [15], onde é proposto a definição de um framework completo para servir como fundamentação conceitual para representação e consulta de dados espaço-temporais através da definição de um modelo de dados e uma linguagem de consulta para lidar com geometrias espaço-temporais. O modelo de baseia na definição formal de tipos de dados abstratos com operações próprias onde a ideia é que os tipos possam ser usados como valores de colunas em extensões de SGBD tradicionais, integrados em SGBD-OR ou orientado a objetos, bem como integrar suas linguagens de consulta. Porém, o trabalho não contempla trajetórias como entidades identificáveis por seu significado, mas sim como um registro único sem semântica associada para todo o movimento do objeto visto como um ponto móvel, o que dificulta bastante o desenvolvimento de de aplicações para consultas e análises que requerem mais significado do que apenas comparações espaciais e temporais - conforme exemplificado na seção 2.2.

Duas das principais soluções para a banco de dados de objetos móveis têm como base a representação do modelo proposto Guting em [15]: SECONDO [8] [13] [6] [14] e HERMES [36] [37]. O primeiro diz respeito aos estudos e desenvolvimentos em relação aos tipos de dados abstratos de objetos móveis e aos algoritmos para as operações definidas em [15], como o trabalho de Forlizzi et al. [9], que fornece uma visão resumida sobre essa questão, o trabalho de Lema et al [22], que apresenta um estudo sobre algoritmos para

parte dos métodos propostos e o trabalho de Almeida et al. [6] que apresenta um protótipo. Em resumo, SECONDO é uma plataforma SGBD especialmente ajustada para extensões de módulos algébricos para aplicações não convencionais, que não oferece suporte para um modelo de dados pré-determinado, mas é próprio para a implementação de novos modelos [10]. HERMES, por sua vez, é um mecanismo de banco de dados proposto recentemente para lidar com objetos que alteram sua localização, forma e tamanho - de forma discreta ou contínua - no tempo através de funcionalidades espaço-temporais em SGDB Objeto-Relacional (SGBD-OR), cujo protótipo foi feito a partir da extensão do STAU [33] [38] sobre a plataforma Oracle SGBD-OR a partir da definição de tipos de dados para os objetos móveis.

Várias métodos têm sido propostos para proporcionar melhoria na análise de trajetórias, incluindo o uso de banco de dados espaço-temporais e técnicas de data mining. Estes métodos podem ser classificados em quatro grupos: (i) descoberta e geração de padrões de trajetórias (e.g. convergência, encontros, multidões e comboios) de acordo com as propriedades geométricas das trajetórias [20]; (ii) extração de clusters de pontos brutos das trajetórias, basicamente utilizando tempo e espaço para encontrar trajetórias que estão localizadas em regiões densas, trajetórias com formas ou distâncias similares e trajetórias que se movem entre regiões em tempos similares [28] [19] [11] [34]; (iii) análises de trajetórias de um ponto de vista semântico, buscando adicionar informações do contexto [2] [12] [43] [32] [40] [27]; (iv) desenvolvimento de arquiteturas, processo de ETL (Extraction, Transformation and Loading), modelos de dados e linguagens para suporte na construção de data warehouses de trajetórias (TDW). Os dois últimos grupos são o foco do trabalho descrito aqui e, portanto, o resto deste capítulo irá analisar trabalhos relacionados deste grupo em mais detalhes.

3.1 Trabalhos relacionados para trajetórias semânticas

A definição de um modelo conceitual para trajetórias é a base para o desenvolvimento de um modelo de dados para as mesmas. Por isso destacamos o trabalho de Spaccapietra et al. [43], utilizado como fundamentação teórica básica a ser estendida para definição dos tipos e meta-esquemas propostos em nosso trabalho. O trabalho apresenta uma definição semântica para trajetórias baseada em *stops* e *moves*, onde as paradas representam as partes importantes para a semântica e, além disso, analisa as necessidades, do ponto de vista da aplicação, em termos de modelagem de dados para agregar valor semântico às trajetórias dos objetos móveis. Isso se dá através de duas abordagens por modelagem conceitual para trajetórias sendo percebidas como elementos de primeira

ordem: uma através da proposta de um meta-esquema e a outra na definição de tipos próprios para trajetórias, onde ambas podem ser estendidos com informações semânticas do contexto da aplicação. As duas abordagens pretendem proporcionar construtores e regras para os projetistas de BD que utilizam dados de mobilidade. Essas modelagens buscam oferecer uma caracterização para trajetórias e seus componentes com atributos, restrições semânticas, restrições topológicas e links para os objetos da aplicação. As duas contribuições de modelagem são demonstradas através da apresentação de esquemas adaptados de cada uma para o contexto de migração de pássaros.

Os trabalhos [1] e [2] de Alvares et al. utilizam a definição do modelo conceitual de trajetórias com segmentação por *stops* e *moves* proposto no trabalho citado acima [43][1]. O primeiro realiza a descoberta de padrões de movimento para trajetórias com base em técnicas de data mining. É proposto um framework para modelar e realizar a mineração para descoberta de padrões de movimentos semânticos. O principal foco está na descoberta dos movimentos mais frequentes entre duas paradas (i.e. *stops*), onde cada *stop* é visto como um interesse da aplicação, o que proporciona melhor entendimento sobre comportamento dos movimentos em espaços geográficos e facilita escrever consultas sobre os padrões dos objetos móveis. O segundo, levando em consideração a complexidade de consultas e análises sobre dados de trajetórias em formato bruto, busca acrescentar informações semânticas às trajetórias com base no contexto geográfico no qual estão inseridas. Para isso é proposto um modelo genérico para representar as partes das trajetórias que são importantes para a aplicação bem como um algoritmo para computar essas partes, que no modelo adotado são paradas (i.e. *stops*) transformadas em objetos geográficos que possuem significado para a aplicação. As análises passam, então, a serem executadas sobre esses dados, minimizando o espaço de busca espacial e as junções espaciais e, com isso, consegue-se obter uma redução significativa na complexidade de consultas para análises semânticas de trajetórias.

Ainda com base no modelo conceitual de Spaccapietra et al. [43], temos, fugindo um pouco da linha de análise, os trabalhos de Palma et al. [32] e Rocha et al. [40] que seguem abordagens por clusterização para descoberta de conhecimentos. O primeiro propõe uma solução para descoberta de locais importantes na trajetória com base na velocidade, ou seja, a descoberta dos *stops*. O trabalho se baseia na ideia simples de que segmentos com menor velocidade podem representar locais de interesse em duas etapas: a primeira parte do processo atua na descoberta de potenciais *stops* e a segunda, com base no resultado da primeira, analisa-os em relação às informações geográficas. O segundo busca a descoberta de locais de interesse com base em variações de direção, considerando domínios onde esse aspecto é importante, como a descoberta dos locais onde os navios realizam pesca oceânica, evitando que os mesmos pratiquem a atividade em locais proibidos.

[1] sua versão publicada como relatório técnico [42]

Em outra linha de trabalhos, Guc et al. [12] defende a idéia de que os dados de trajetória podem ser utilizados para facilitar o processo manual de anotação semântica das trajetórias. Para isso propõe um modelo para anotação de trajetórias com base na noção de episódios que permite a separação entre a parte semântica e física e, também, uma arquitetura para programa para realização das anotações semânticas. O trabalho de Yan et al. [46] segue a mesma linha e propõe um framework (SeMiTri) de propósito geral para vários domínios de aplicação (i.e. para trajetórias heterogêneas) que permite gerenciar e enriquecer trajetórias com anotações semânticas, permitindo que a aplicação possa se beneficiar de uma representação semântica do movimento através de inferências realizadas a partir das propriedades espaço-temporais dos dados brutos de posição (e.g. extração de paradas e movimentos, rastreamento de sua direção ou padrão de movimento), das regiões geográficas percorridas pelas trajetórias (e.g. ruas e locais notáveis) e dos objetos de aplicação relacionados com as trajetórias (e.g. clientes, estacionamentos). O framework se utiliza do modelo para trajetórias semânticas proposto, onde uma trajetória é representada como uma sequência de episódios semânticos que correspondem a uma interpretação da aplicação e, ainda, apresenta três algoritmos para realização de anotações das trajetórias, um com base em regiões de interesse, outro com base no trajeto e o último com base nos pontos de interesse, sendo estes algoritmos os responsáveis por cobrir as peculiaridades da heterogeneidade das trajetórias, como trajetórias de veículos, de pessoas a pé, animais etc.

3.2 Trabalhos relacionados para data warehouse de trajetórias (TDW)

A tecnologia de data warehouse (DW) tem auxiliado muitas empresas a melhorar seus processos de tomada de decisão bem como sua performance. E, com isso, tem havido um número considerável de propostas na literatura que buscam integrar as funcionalidades e características relacionadas com os dados de trajetória com os sistemas de DW [30] [29] [35] [26] [24] [4] [31]. O principal objetivo é realizar o armazenamento de trajetórias em um data warehouse e fornecer processamento multidimensional com a capacidade de consultar essas trajetórias de acordo com várias perspectivas de análises com o objetivo de apoiar o processo de tomada de decisões relacionados com objetos móveis. Este tipo de ambiente tem sido referenciado na literatura como trajectory data warehouses.

Em [26], de Marketos et al., os dados originados a partir de sensores GPS foram carregados em um TDW utilizando uma série de procedimentos de ETL. E ainda, mecanismos para realização de consultas analíticas com agregações espaço-temporais sobre dados de trajetórias são detalhados em [24] [23]. Além disso, baseado no modelo

conceitual de trajetória proposto por Spaccapietra et al. [43], um esquema de modelo conceitual para TDW que objetiva o apoio para o monitoramento dos dados bioquímico de atletas é discutido por Porto et al. em [39]. Entretanto, estes dados bioquímicos não são organizados em estruturas de dados dimensionais, através do uso de tabelas fato e de dimensão e a representação de diferentes níveis de agregação. De fato, técnicas de data warehouse tradicionais e ferramentas analíticas atuais não satisfazem os requisitos de aplicações TDW porque a representação e agregação de trajetórias implica a necessidade de modelagem e agregação de vários tipos de dados inter-relacionados, tais como as geometrias, informações de contexto e objetos espaço-temporais. Consequentemente, ainda não existe consenso sobre como as trajetórias devem ser modeladas multidimensionalmente, organizadas em diferentes níveis de agregação e utilizadas para responder consultas com agregação de sistemas OLAP.

Mesmo que as vantagens de data warehouses sejam conhecidas a algum tempo, até onde vai nosso conhecimento, nenhuma solução comercial de business intelligence (BI) emprega TDW atualmente. Encontramos em [30], de Orlando et al., a primeira proposta de um DW para trajetórias que define um cubo de dados OLAP com ambas as dimensões espacial e temporal, e um conjunto de hierarquias espaço-temporais determinadas pelo uso de grids regulares. Tanto medidas numéricas como espaciais são propostas nesse trabalho. Alguns outros estudos têm sido conduzidos para auxiliar o desenvolvimento e difundir o uso de TDWs. Porém, escassos exemplos destes lidam com o aspecto semântico das trajetórias que são, raramente tratadas como objetos de primeira classe e, portanto, o resultado da análise multidimensional deixa a desejar em conhecimento sobre a descoberta de padrões multidimensionais.

No trabalho de Orlando et al. [29] são tratadas, de forma mais geral, as características, requisitos e problemas envolvidos no desenvolvimento de um data warehouse de trajetórias. São apresentados problemas, previamente introduzidos por Braz et al. em [5], com respeito ao projeto de um TDW, no qual é ressaltado o processamento e armazenamento de medidas obtidas por agregação holística. É proposto, ainda, um modelo para data warehouse de trajetórias implementado em plataforma Oracle e é realizada uma avaliação sobre os desafios de projeto - como viabilidade e desempenho - e do processo de ETL com experimentos realizados com dados reais e fictícios. Porém, neste trabalho, é importante ressaltar que, o modelo de TrDW proposto somente armazena informações agregadas sobre as trajetórias. Os dados de trajetórias são armazenados e representadas em um BD de objetos móveis proposto por Guting e Schneider em [16], o qual fornece extração de informações sobre as trajetórias para compor o DW.

Atuando, ainda, no contexto das dificuldades do processo de trajectory warehou-

sing[2], o trabalho de Marketos et al. [26] propõe um framework que leva em consideração os problemas de construção das trajetórias para um Moving Objects Database (MOD) a partir dos dados brutos, o processo ETL para povoamento do TDW e posteriormente da agregação de medidas com propósito de aplicação de técnicas OLAP. Porém, apesar de apresentar um exemplo de TDW, a construção de trajetórias proposta ainda não apresenta estas como um elemento de primeira grandeza com valores semânticos associados e, além disso, não padroniza um modelo ou esquema para o DW de trajetórias, realizando apenas uma breve discussão sobre o que o mesmo deve apresentar como dimensões.

O estado da arte dos TDWs é discutido por Pelekis et al. em [35] e por Gomez et al [17]. O primeiro apresenta uma série de requisitos de modelagem e gerenciamento dos dados apresentando as principais características e problemas, e com isso, se forma um guia com o caminho completo para a construção de TDW. Dentre os principais requisitos e peculiaridades apresentados destacamos a necessidade de modelagem conceitual com tipos complexos, o relacionamento com hierarquias da dimensão espacial, que pode apresentar problema denominado de *partial containment relationships*, onde os dados de uma trajetória podem estar em mais de um nível hierárquico, e a relevância do desenvolvimento de um modelo de dados abrangente para especificação e implementação de funções de agregação espaço-temporais. O segundo trata mais especificamente sobre o problema da análise de trajetórias e realiza um apanhado dos esforços mais recentes no processo de data warehousing, passando por todas as preliminares que envolvem o contexto espaço-temporal específico de trajetórias como Temporal DW, SDW e OLAP, agregação espacial, MODs etc. Indica as futuras direções a serem tomadas na aplicação de OLAP em contexto espaço-temporal. Além disso, apresenta as abordagens para trajectory data warehouse de dois projetos: GeoPKDD[3] com o Hermes e o framework Piet[4].

A iniciativa do projeto GeoPKDD (*Geographic Privacy-aware Knowledge Discovery and Delivery*), que foi um dos pioneiros na descoberta de conhecimento a partir de dados de objetos móveis, resultou em alguns dos principais trabalhos utilizados como referências básicas para nossas contribuições, tais como [3], [7] e [35]), e tinha como objetivo o desenvolvimento de métodos para representar, armazenar e gerenciar objetos móveis, prevendo, ainda, a possibilidade que estes pudessem mudar sua geometria ao longo da trajetória. Para isso, o foco estava no desenvolvimento de metodologias para data warehousing e data mining que suportassem trajetórias de objetos móveis. Grande parte do esforço do projeto foi empregado na extração e transformação dos dados de trajetórias, a partir dos quais, eram realizados os processos de mineração de dados para descoberta e extração de conhecimento. No entanto, pouco esforço foi dispendido na pesquisa sobre

[2]utilizamos o termo warehousing para o processo de criação e povoamento do data warehouse
[3]http://www.geopkdd.eu
[4]http://piet.exp.dc.uba.ar/piet

a aplicação de técnicas de data warehousing e OLAP com essa finalidade. Outro projeto que, atualmente, realiza pesquisas para descoberta de conhecimento a partir de dados de mobilidade seguindo a mesma linha do GeoPKDD é o MODAP[5] (*Mobility, Data Mining and Privacy*). Este atua na coordenação e incentivo a atividades de pesquisa em temas que envolvem mobilidade, mineração de dados e privacidade.

3.3 Trabalhos relacionados para funções de agregação de trajetórias

Na realização de análises em meios onde a quantidade de dados é vasta, a obtenção de informações sumarizadas pode ser mais útil do que a obtida sobre entidades individualizadas. Isso ocorre em ambientes transacionais e, principalmente, em data warehouses, que são voltados para realização de análises com agregação das medidas da tabela fato que, no contexto do nosso trabalho, devem ser informações de trajetórias. E, conforme mencionado na seção anterior, ainda não existe consenso sobre como as trajetórias devem ser modeladas multidimensionalmente para realização de agregação em sistemas OLAP. Como trajetória é um tipo específico de entidade espaço-temporal, recaímos na área de estudo de funções de agregação para dados espaço-temporais.

Vega Lopez et al. realiza uma pesquisa detalhada em [44] sobre as principais técnicas para avaliação de consultas com agregação sobre dados espaciais, temporais e espaço-temporais. Da mesma forma, realiza uma caracterização de todos os tipos distintos de consultas de agregação espaço-temporais que podem ser do interesse do usuário apresentando definições formais para cada uma, o que permite classificar todas as abordagens da literatura em um mesmo framework, que por sua vez permite a análise e comparação das diferentes técnicas existentes para avaliação de consultas com agregação. Além de poder ser considerado como um guia para os trabalhos que envolvem a agregação de dados espaciais, temporais e espaço-temporais, o trabalho aponta as lacunas da área que precisam de estudos mais efetivos e, também, oportunidades de pesquisa.

Alguns trabalhos da literatura voltada para agregação de dados de trajetórias propõem soluções para agregação de informações de trajetórias em contexto operacional [21] [11] [4] [18], enquanto outros, mais especificamente, discutem e propõe soluções para como a tecnologia de DW pode ser usada para obter informações agregadas de trajetórias e realizar operações OLAP sobre trajetórias [30] [29] [5].

Os trabalhos de Lee et al. [21] e Baltzer et al. [4] apresentam propostas para a agregação de dados de trajetórias de objetos móveis. O primeiro apresenta algoritmos

[5]http://www.modap.org

para geração de clusters de trajetórias a partir dos dados de objetos móveis e, para cada cluster, encontrar as trajetórias representativas. Os clusters são gerados por segmentos independentes das trajetórias obtidos em uma primeira fase de segmentação, e não pelas trajetórias como um todo. E, após essa fase de segmentação, na fase de agrupamento é que ocorre a identificação dos segmentos semelhantes e geração das trajetórias representativas. Essa abordagem por segmentação permite a identificação de regiões de interesse - informação esta que pode se perder quando o cluster é computado com base na trajetória como um todo. O segundo propõe um operador com três versões para agregação de trajetórias por operações *group by*:

- *group by overlap* agrega subconjuntos de trajetórias que correspondem a sequências de movimentos com sobreposição suficiente entre trajetórias subsequentes;
- *group by intersection* agrega subconjuntos de trajetórias que possuem movimentos similares ou sincronizados, assim como movimentos paralelos;
- *group by overlap and intersection* combinação entre os dois operadores anteriores para agregar subconjuntos de trajetórias que apresentam uma combinação de sequências de movimentos e movimentos similares ou sincronizados.

Para cada operador são apresentados os algoritmos e, com base nesses operadores, é proposto, ainda, um ambiente interativo para análise de trajetórias que permite a realização das operações *roll-up* (operação para visão mais geral das informações) e a operação inversa *drill-down*. Os dois trabalhos, no entanto, não apresenta nenhum modelo ou estrutura para a representação de trajetórias e considera apenas a sequência de dados espaço-temporais no formato bruto ($< x, y, t >$). Outros trabalhos nesse contexto são apresentados por Jeung et al. [18] para abordar a descoberta de grupos de objetos móveis que seguem juntos (*convoy*) e por Giannoti et al [11], onde é introduzido e desenvolvido um padrão espaço-temporal para agregação de movimentos de trajetória. Esse padrão analisa os dados de trajetórias e descreve os padrões de comportamento em termos de espaço e tempo, apontando áreas de interesse (regiões do espaço visitadas durante os movimentos) e os tempos de duração do movimento entre as transições.

Os trabalhos [30] e [29] de Orlando et al. mencionados na seção anterior estendem o trabalho de Braz et al. [5] e discutem, ainda, problemas para os modelos de data warehouses propostos em relação ao armazenamento de medidas agregadas, focando principalmente no problema da computação da medida de presença. Esse problema é detalhado através do problema da contagem para a medida de presença, a qual retorna o número de trajetórias distintas em uma região durante um intervalo de tempo nas operações de *roll-up*. Nestes trabalhos, a tabela fato não representa um objeto de alto nível para trajetórias, o que limita a capacidade de formular novas consultas com agregação, pois, como a tabela

fato armazena valores agregados e não medidas que podem sofrer agregação da forma que for desejado, as agregações devem ser pré-determinadas. Além do mais, para contornar o problema da contagem, no qual uma trajetória pode ser contada mais de uma vez, as abordagens se utilizam de cálculos complexos para calcular os valores da agregação, nos quais são levados em conta valores probabilísticos, o que, consequentemente, apresenta margens de erro.

3.4 Conclusão

Neste capítulo introduzimos uma visão geral dos trabalhos que estão envolvidos com o armazenamento e manipulação de dados de trajetórias. Em seguida realizamos um apanhado sistemático dos trabalhos que apresentam contribuições para representação e análise de trajetórias semânticas de objetos móveis em ambiente operacional e, após isso, em ambiente multidimensional. Por último discutimos os trabalhos que propõem soluções para agregação de informações de trajetórias, também, em ambiente transacional e multidimensional.

Para suprir as necessidades e integrar os três contextos de trabalhos mencionados acima de análise de dados de trajetórias, incluindo as características semânticas da aplicação e considerando as análises multidimensionais através de data warehouse, propomos neste trabalhos duas modelagens, ambas baseadas em meta-esquemas, para o desenvolvimento de esquemas de dados para trajetórias, tanto em ambiente transacional quanto multidimensional, nos quais as trajetórias são consideradas objetos de primeira classe. Como complemento para fechar os requisitos necessários para realização de análises sobre trajetórias semânticas de objetos móveis, propomos uma assinatura genérica para guiar a construção de funções de agregação específicas para trajetórias e, ainda, um conjunto de funções de agregação. Com isso tempos uma solução completa para realização de análises de trajetórias tanto em contexto transacional quanto histórico dos dados, o quer permite a criação de relatórios e outros produtos que tendem a ser muito mais ricos do que aqueles cujos dados espaço-temporais são tratados como um conjunto de pontos sem semântica associada. Também, um caso de estudo com trajetórias reais de veículos de distribuição foi desenvolvido para ilustrar como, tanto trajetórias de tempo real quanto históricas, foram modeladas e consultadas para realização de análises e geração de relatórios.

4 MODELAGEM DE DADOS DE TRAJETÓRIAS

Neste capítulo iremos apresentar duas soluções de modelagem para a representação e estruturação de trajetórias de acordo com os problemas e requisitos mencionados anteriormente no capítulo de introdução e detalhados na seção seguinte (seção 4.1). Cada uma foi projetada para um objetivo distinto de acordo com o contexto de aplicação. O primeiro contexto se refere à manipulação de trajetórias para monitoramento e obtenção de informações em tempo real sobre dados recentes e sua interação com outros objetos (e.g. veículos e regiões). O segundo contexto se refere à realização de análises com resultados agregados em diversos níveis de detalhes sobre as trajetórias e suas informações temáticas (e.g. regiões, pontos de interesse e origem dos objetos móveis) com a finalidade de gerar informações e relatórios para suporte a decisões estratégicas gerenciais. A solução para o primeiro contexto (seção 4.2) tem como fundamentação o encapsulamento dos dados de trajetórias em um objeto do tipo complexo *TrajectoryType* específico, o qual provê atributos e métodos que permitem acesso e manipulação de seus componentes - medidas e eventos - e interação com outros objetos ou elementos do domínio. Para o contexto seguinte, a segunda solução (seção 4.3) apresenta uma modelagem multidimensional para uso de técnicas de data warehouse e ferramentas OLAP, contendo como fato o tipo trajetória e como dimensões suas informações temáticas. Além disso, na seção 4.4, oferecemos algoritmos para guiar a construção dos objetos e povoamento para esquemas de ambos os modelos propostos e, ainda, na seção 4.5 uma assinatura padrão para guiar a construção de funções de agregação para trajetórias bem e um conjunto de funções de agregação proposto para nosso estudo de caso.

4.1 Requisitos

Para os diversos domínios de aplicação que atuam com dados de objetos móveis e possuem interesse em analisar fenômenos espaço-temporais de trajetórias, a obtenção de informações e realização de análises é essencial, porém não é uma tarefa trivial e não se trata, apenas, da manipulação e avaliação de um conjunto de dados sequenciais e gerenciamento de imenso volume de dados. Não é possível capturar esses objetos através

de atributos distintos para as características espaciais e temporais como é realizado para o registro da evolução de objetos espaço-temporais menos complexos (como regiões) [15]. Para isso é necessário um registro mais estruturado dos movimentos que permita a manipulação de trajetórias como entidades de primeira ordem com a possibilidade de adicionar os aspectos semânticos requeridos pelo domínio da aplicação e, além disso, modelos e métodos que permitam a realização de análises tanto em tempo real quanto análises sobre o histórico dos dados.

A manipulação e obtenção de qualquer informação sobre os dados brutos de trajetórias obtidos dos dispositivos de localização geográfica é trabalhosa e não possui qualquer estruturação ou significado que possa ajudar a abstrair essa complexidade. Se torna complicado, até mesmo, obter simples informações como o comprimento do percurso realizado por um objeto móvel (e.g. automóvel). Seria, então, praticamente inviável a realização de consultas em tempo real e analíticas sobre as medidas de interesse das trajetórias, como por exemplo o comprimento, duração e ocorrência de eventos associados a elas. A principal dificuldade decorre do fato que as trajetórias devem ser vistas como uma entidade única que represente a sequência de observações que descrevem o comportamento do objeto móvel. Para isso, devemos poder perceber as trajetórias como tipos complexos específicos de objetos espaço-temporais. Portanto, o requisito mais básico para realização de qualquer tipo operação em alto nível envolvendo trajetória, é sua estruturação como um objeto complexo de primeira ordem capaz de oferecer manipulação em alto nível e com possibilidade de associar a semântica do domínio.

É necessário, também, uma forma de representação com foco no objeto trajetória que permita a realização de análises sobre todo o histórico dos dados com informações agregadas em diversos níveis de detalhes eficientemente. Ao mesmo tempo é importante preservar as características de objeto complexo em alto nível e suas relações com outros objetos temáticos do contexto permitindo, ainda, o emprego de técnicas de datawarehouse e aplicações OLAP para suporte aos processos de tomada de decisão.

Estender modelos de dados existentes com novos tipos de dados é a chave para prover uma adequada representação e manipulação de objetos complexos do mundo real. Na área de banco de dados esta prática tem sido frequentemente aplicada com objetivo de reduzir a complexidade de projeto e desenvolvimento de aplicações como acontece, por exemplo, com a criação de novos tipos complexos compostos por atributos de tipos já existentes no SGBD - inclusive tipos para geometrias espaciais. Isto permite, não só facilitar o desenvolvimento, mas também a manipulação eficiente e otimizada dos novos tipos realizada pelo próprio banco.

Além da estruturação como objeto complexo de primeira ordem, em geral, os usuários precisam de informações que associam a semântica da aplicação à trajetória como

um todo ou com partes da trajetória. Então, além da representação de trajetória com um tipo dedicado, esse tipo deve permitir que haja a decomposição do objeto trajetória em partes semânticas. Isso enfatiza a necessidade de representação por tipos complexos. Então, para atacar este e os demais requisitos apresentados acima, nós propomos dois meta-esquemas para esquemas em banco de dados para análises em tempo real e históricas. Um para ambiente transacional e outro para ambiente multidimensional, onde consideramos o primeiro apropriado para tempo real e o segundo para histórico.

Como complemento dos requisitos tratados acima, a partir de uma perspectiva analítica, é essencial, ainda, fornecer informações sumarizadas a partir de um conjunto de trajetórias. Funções de agregação têm sido amplamente utilizadas em banco de dados e aplicações de suporte a decisão com essa finalidade. Entretanto, as funções de agregação proporcionadas pelos bancos não são apropriadas para agregar objetos complexos como as trajetórias. É necessário, então, para melhorar a capacidade de processamento de análise de trajetórias, um novo conjunto de funções de agregação para dados de trajetórias.

4.2 Meta-esquema transacional

Padrões de projetos são artefatos bastante utilizados na engenharia de software para facilitar e guiar o processo de desenvolvimento através de soluções pré-definidas para problemas recorrentes no projeto de softwares. Porém, em nossas contribuições nós utilizamos os mesmos para guiar o desenvolvimento de esquemas de banco para dados de trajetórias de objetos móveis. A idéia consiste em oferecer um meta-esquema padrão de esquema de dados utilizando os conceitos de trajetória semântica e pontos para extensão do esquema (*hooks*), os quais podem ser utilizados para adicionar informações semânticas relacionadas com as trajetórias.

O meta-esquema transacional para representação semântica de trajetórias proposto tem como principal característica o encapsulamento dos dados de trajetórias em tipos estruturados. No modelo de dados são consideradas, além da entidade trajetória, entidades para representar os objetos móveis e os eventos decorrentes de cada trajetória. O modelo permite uma maior legibilidade e facilidade de manipulação dos dados do que em seu estado bruto e, além disso, o encapsulamento de cada entidade em um tipo estruturado permite propor, ainda, métodos para manipulação de cada entidade e seus atributos. Com isso, tem-se uma maneira eficiente de lidar com os aspectos geométricos dos objetos, enquanto o aspecto semântico fica a cargo dos atributos exigidos pelo domínio de aplicação acrescentados como extensão do modelo e dos relacionamentos do objeto trajetória com os demais objetos do contexto da aplicação.

O tipo proposto para trajetórias é apresentado na Figura 2, o qual possui duas

listas de objetos encapsuladas. Uma para objetos que representam os dados oriundos dos dispositivos de localização geográfica e outra para objetos que representam a segmentação semântica da trajetória com base nos eventos decorrentes da mesma. Essas listas são responsáveis por capturar os aspectos geométricos e semânticos das trajetórias em um único tipo de dado. Cada evento possui dois pontos espaço-temporais que correspondem ao início e fim do mesmo e uma descrição textual sobre o mesmo (caracterização do tipo de evento), a qual é própria de cada aplicação. Mais ainda, possui métodos recuperar e manipular os atributos geométricos e computar medidas da trajetória como, por exemplo, o comprimento, duração, velocidade média e a linha espacial do seu percurso. Além dos métodos já propostos no encapsulamento, o projetista da aplicação pode estender o tipo com novos métodos e atributos de acordo com suas necessidades. Além da semântica associada ao tipo trajetória, qualquer outra deve ser acrescentada através de novos atributos em tipos definidos pelo projetista (e.g. objeto móvel contendo uma lista de trajetórias) ou através do relacionamento de trajetórias com outros objetos (e.g. locais de interesse).

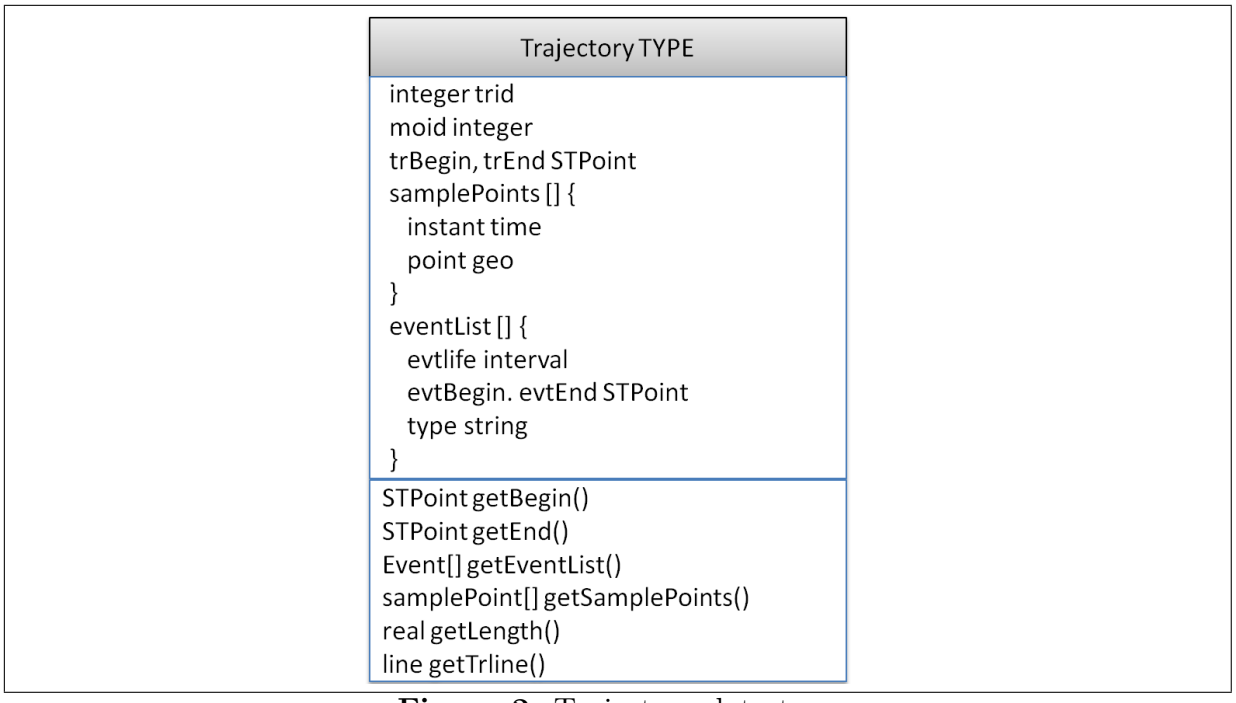


Figura 2: Trajectory data type

Exibimos na Figura 3 um exemplo de extensão do tipo *TrajectoryType* do padrão proposto de acordo com o nosso estudo de caso para ilustrar. As extensões são realçadas por um sinal de "+". Na definição dos métodos, foi identificada a necessidade de obter relações de velocidade média para a trajetória como um todo, sua duração total e, principalmente, métodos para manipulação dos eventos. Dentre estes métodos identificados, destacamos a necessidade de um método para contagem de eventos de tipos específicos da trajetória (e.g. quantidade de eventos do tipo Atendimento realizado em uma trajetória), para computar a soma da duração de tipos específicos de eventos (e.g. a soma da

duração dos eventos do tipo atendimento da trajetória) e para computar a soma do valor associado aos eventos (e.g. soma da quantidade de produto entregue por trajetória). Na definição das propriedades, criamos atributos para representar como variável o resultado do processamento dos métodos para evitar reprocessamento em trajetórias que já foram finalizadas.

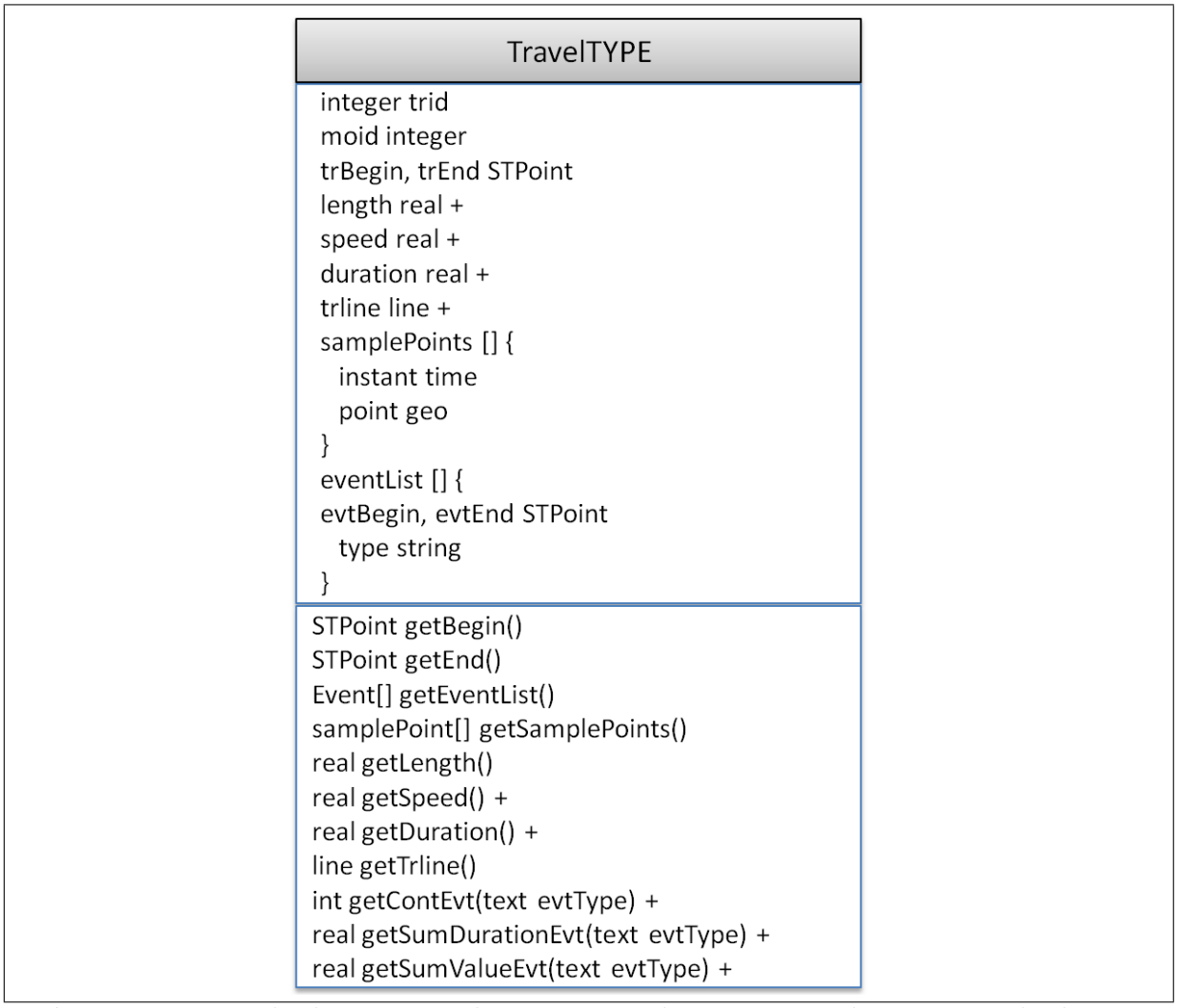


Figura 3: Exemplo de extensão do Trajectory data type com base no estudo de caso

Como exposto, grande parte das informações das trajetórias é dependente da aplicação. É impossível fixar um tipo trajetória adequado a todos os domínios. A solução, então, é decompor os atributos do tipo em componentes na forma de um meta-esquema na qual outros objetos ou informações semânticas da aplicação possam ser ligados. Para o atributo que identifica o objeto móvel, um novo tipo (*MovingObjectType*) foi criado, já que uma trajetória é sempre resultado de um deste.

A Figura 4 exibe o meta-esquema proposto para o esquema transacional de trajetórias com tipos necessários para representar as trajetórias, seus eventos e os objetos móveis, pois consideramos que estes são os componentes comuns para todas as aplicações

de gerenciamento de trajetórias. O projetista pode acrescentar atributos para os tipos e, através dos pontos de extensão representados pelas caixas quadriculadas, acoplar os tipos da aplicação. Esses tipos podem ser especializações para os tipos do padrão. Podemos ter especializações do tipo *MovingObjectType* para representar os diferentes tipos de objetos móveis como, por exemplo, pessoas, veículos, animais, navios etc. Raciocínio análogo pode ser aplicado para trajetórias, onde além de estender o tipo *TrajectoryType* com os atributos necessários para cada domínio, podemos especializá-lo para diferentes interpretações de trajetória. Podemos ter, para trajetórias de pessoas se deslocando a pé em uma cidade, um tipo específico como trajetórias de turistas. As caixas tracejadas podem representar, ainda, objetos da aplicação com os quais os objetos do padrão se relacionam, como regiões, clientes, locais de interesse (e.g. local de trabalho, shoppings), planejamentos de entrega (nos casos de trajetórias de veículos de entrega) etc. Essas extensões, juntamente com a segmentação da trajetória por eventos, são responsáveis por representar a semântica do domínio da aplicação.

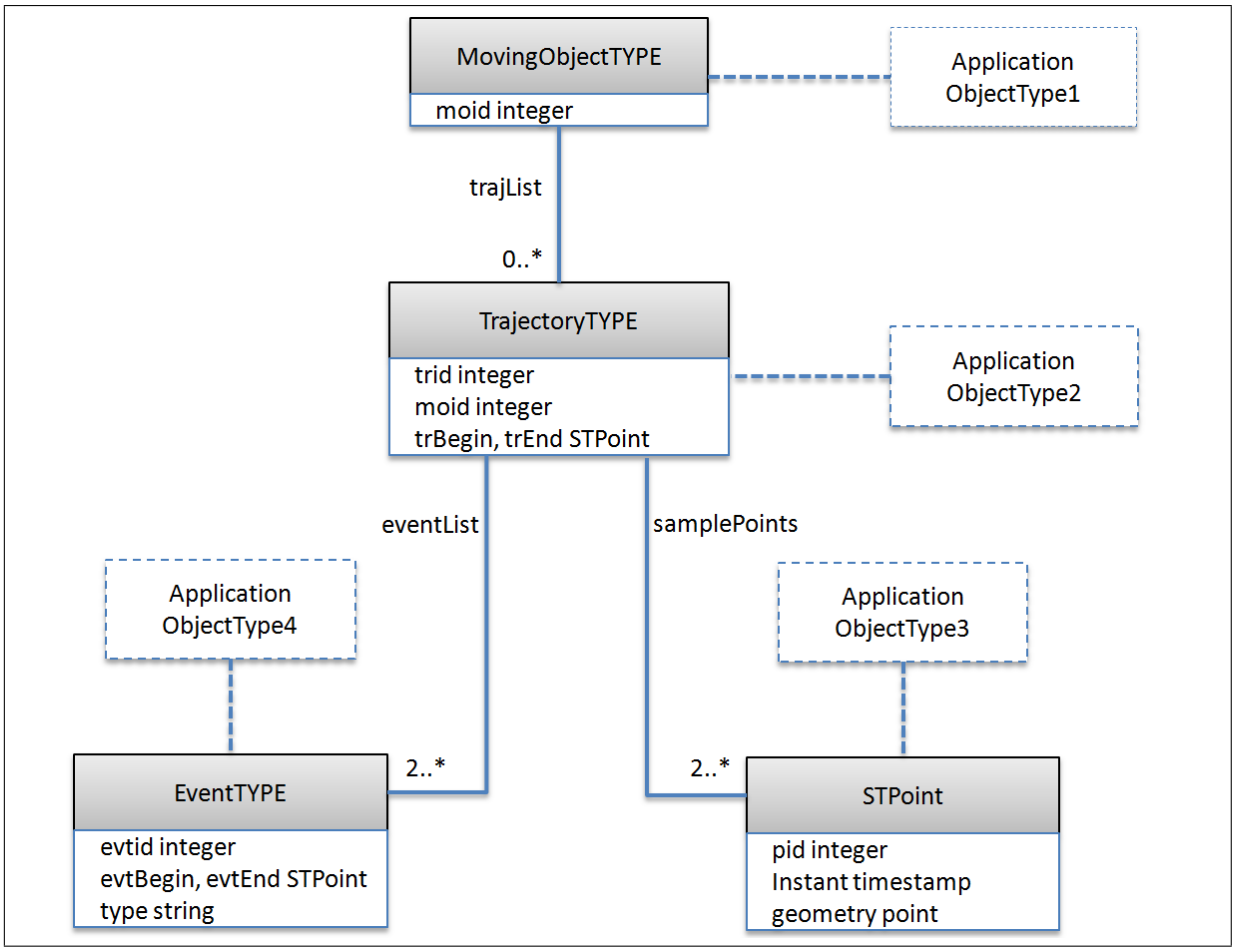


Figura 4: Meta-esquema para ambiente transacional

Ilustramos a extensibilidade do meta-esquema transacional proposto através da Figura 5, na qual apresentamos o padrão adaptado de acordo com o estudo de caso. Estendemos o tipo *MovingObjectType* e criamos a classe Truck para representar os veículos

da aplicação. A trajetória é representada pelo tipo *TravelType* previamente estendido na Figura 3 e, a partir deste tipo, outros dois foram especializados: *DeliveryTravel* e *RefuelingTravel*. O primeiro para representar as características específicas de viagens de entrega de produto, enquanto o segundo para as de viagens de coleta de produto. Ainda, para os eventos, foi necessário a criação de uma propriedade para representar um valor associado ao evento (e.g. a quantidade entregue em um evento do tipo Atendimento, velocidade instantânea em um evento do tipo Excesso de velocidade). Além dos tipos especializados, podemos ter objetos do domínio da aplicação, como os do tipo Área e Planejamento, os quais podem ser usados para estabelecimento de relações. Por exemplo, podemos usar objetos do tipo Área para verificar relações espaciais (e.g. inside, cross) entre estes e trajetórias ou segmentos de trajetória definidos por eventos.

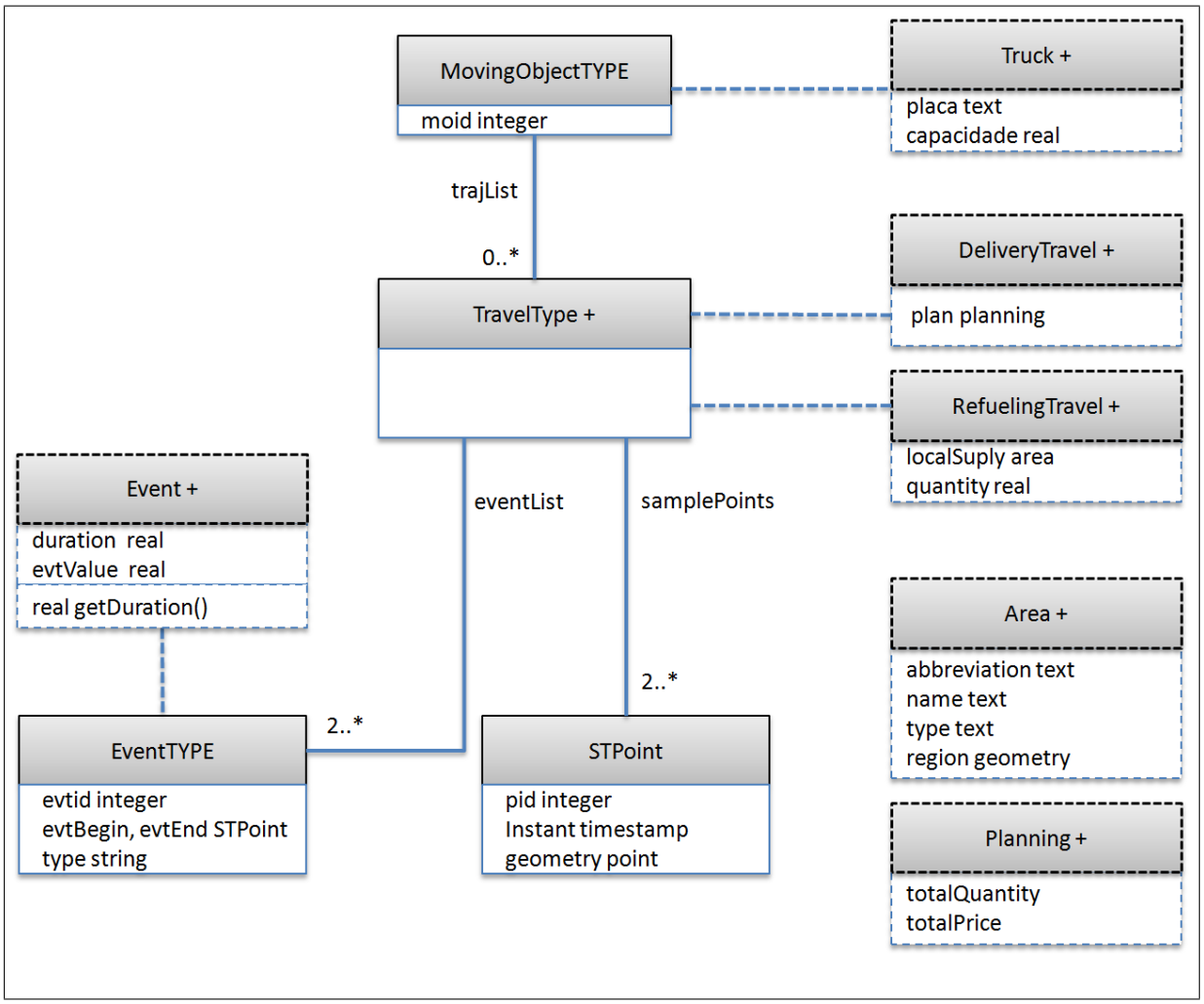


Figura 5: Exemplo de extensão do meta-esquema para ambiente transacional com base no estudo de caso

4.3 Meta-esquema multidimensional

Modelagem multidimensional diz respeito a técnicas e conceitos usados no projeto de datawarehouses, que são compostos, principalmente de dois elementos: fatos e dimensões, onde, basicamente, as medidas dos fatos sofrem agregação de acordo as relações e níveis hierárquicos estabelecidos com as dimensões. O meta-esquema multidimensional proposto busca aliar o encapsulamento dos objetos de trajetórias apresentado no meta-esquema da seção anterior com modelagem multidimensional para aplicações de técnicas de datawarehouse e OLAP. Com isso nos beneficiamos das vantagens já citadas pela abstração e manipulação dos tipos encapsulados junto com as otimizações de desempenho e facilidade de escrita de consultas para análises com informações agregadas de contexto histórico oferecidas pelos sistemas DW.

A Figura 6 apresenta o meta-esquema para ambiente multidimensional. Foi usada notação UML e esteriótipos são usados para ressaltar o fato e as dimensões e, além disso, as dimensões são representadas por caixas tracejadas indicando que estas são pontos de extensões, as quais podem ser estendidas com adição de novos níveis hierárquicos e propriedades. A tabela fato aceita adição de medidas e métodos, porém, todos os componentes originais definidos no *TrajectoryType* devem ser preservados para manutenção do padrão a fim de que todos os métodos e estruturas (e.g. funções para cálculo de medidas e funções de agregação) que atuem sobre os objetos desse tipo possam continuar atuando sobre o fato, visto que este utiliza *TrajectoryType* como medida para agregação. Podemos recuperar, por exemplo, a duração médias das trajetórias da tabela fato que possuem um determinado padrão de eventos ocorridos em um intervalo de tempo especificado e, para isso, é necessário a manutenção dos métodos de acesso e manipulação do tipo.

Como uma trajetória é um tipo específico de objeto espaço-temporal e é sempre relacionada a um objeto móvel, é natural que a proposta do modelo apresente dimensões genéricas para essas características: dimensão de objetos móveis, de espaço (região) e de tempo. As dimensões apresentadas no modelo podem ser definidas de acordo com as informações que podem ser extraídas a partir das instâncias de *TrajectoryType* (e.g. dimensão de tempo e objeto móvel) e de acordo com os objetos da aplicação com as quais as trajetórias podem se relacionar (e.g. regiões e categorias de trajetória). Com respeito às ligações com as dimensões é importante ressaltar que pode haver mais de uma ligação entre o fato e cada dimensão, onde cada ligação representa um interesse de análise do domínio da aplicação. Para exemplificar, apresentamos na Figura 7 uma extensão do padrão com base no estudo de caso apresentado na seção 2.3 que apresenta duas ligações com a dimensão de tempo - uma para início e outra para o fim da trajetória -, uma ligação com a dimensão de objetos móveis, a qual foi estendida com níveis hierárquicos organizacionais da empresa, e uma ligação com a dimensão de regiões representando a

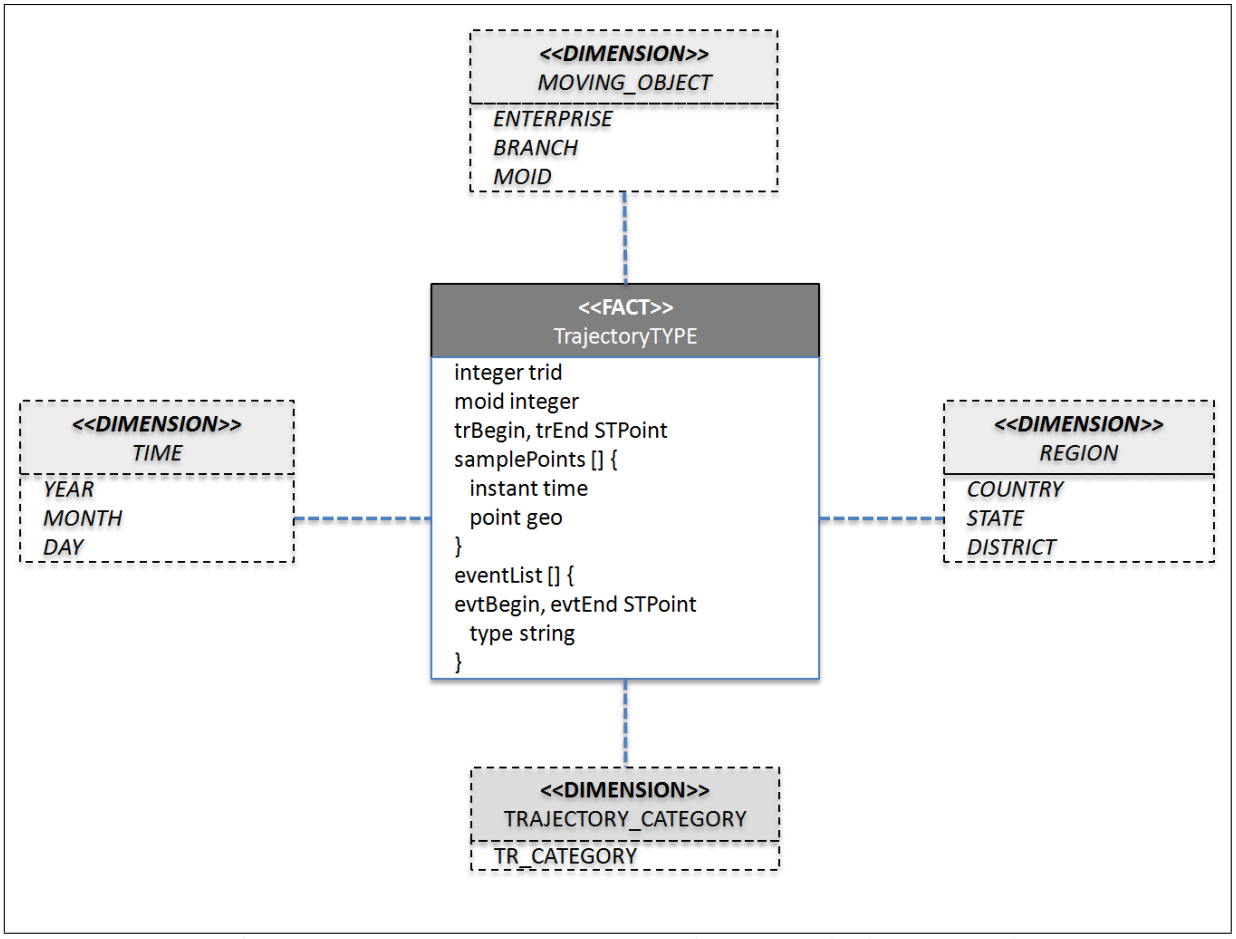


Figura 6: Meta-esquema para ambiente multidimensional

relação *inside*. Perceba que se for do interesse do projetista ele pode criar outras ligações com a dimensão de região utilizando outras relações (e.g. *near by*, *touch*, *cross* etc.). Além das citadas, as demais extensões feitas aparecem com um sinal "+" a frente.

4.4 Algoritmos

Nesta seção apresentamos descrições e algoritmos como contribuição para guiar a criação dos objetos e povoamento dos esquemas dos modelos propostos, primeiramente para o esquema transacional e, atuando a partir deste, o algoritmo para o esquema multidimensional.

4.4.1 Algoritmo de construção dos objetos

Apresentamos, também, como contribuição, o algoritmo *Populate Transactional Schema* (Algorithm 4.4.1), que atua sobre os dados brutos, para guiar a construção dos objetos e povoamento de esquemas instanciados a partir do meta-esquema transacional. A partir do conjunto bruto dos pontos espaço-temporais capturados pelos dispositivos de

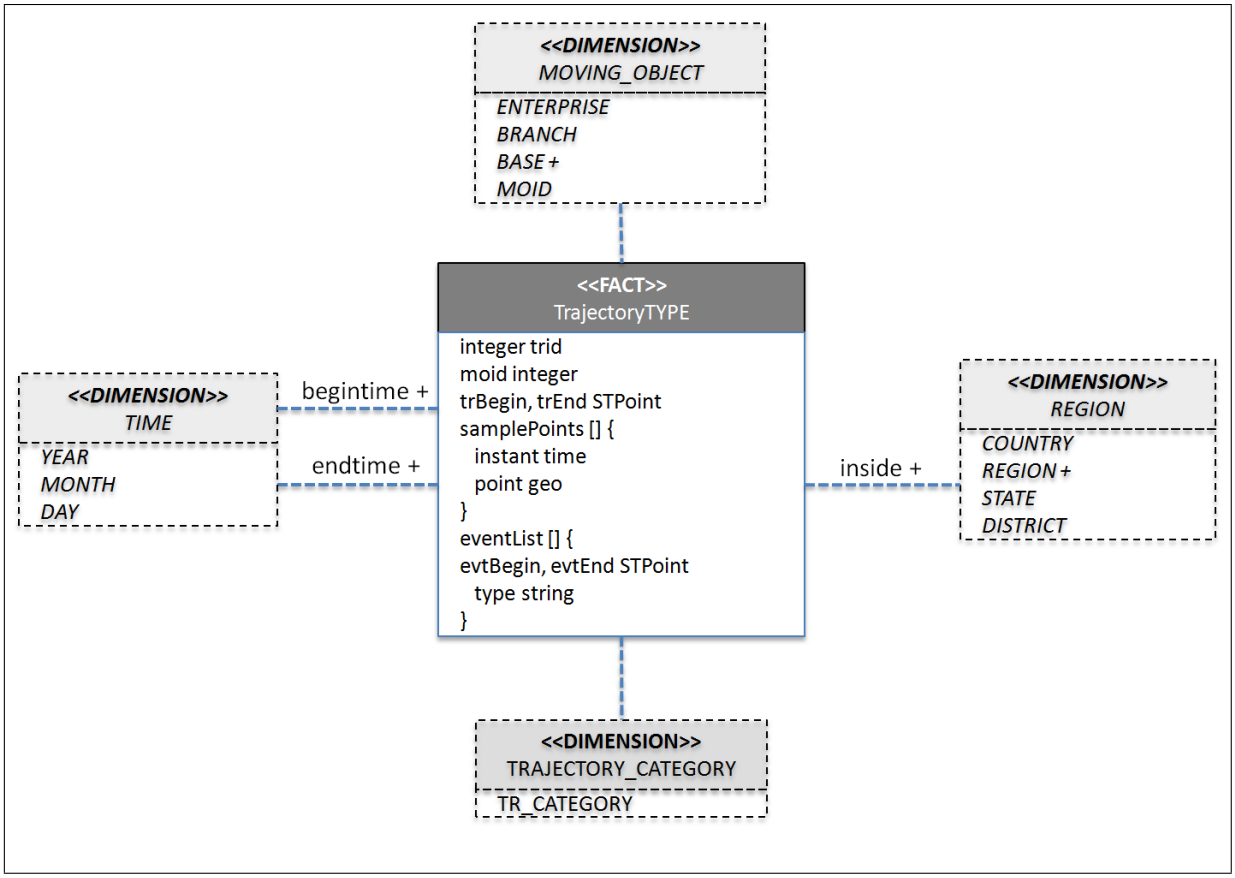


Figura 7: Exemplo de extensão do meta-esquema para ambiente multidimensional com base no estudo de caso (seção 2.3)

localização e, para cada objeto móvel identificado (linha 1), é criado uma instância do tipo *MovingObjectType* (linha 2). Após isso, é chamada uma função para construção das trajetórias (linha 3) que deve ser desenvolvida pelo projetista de acordo com o significado de trajetória para a aplicação. Para cada trajetória criada nessa etapa (linha 4) uma instância do tipo *TrajectoryType* é criada (linha 5) e, por fim, através de outra função que deve ser fornecida pelo projetista de acordo com os interesses da aplicação, são criados as instâncias de *EventType* através da segmentação da trajetória em sua lista de eventos (linha 6).

Algorithm 4.4.1 Populate Transactional Schema

1: **for** each distinct *moid* in the Raw Spatio-Temporal Dataset **do**
2: creates an instance *mo* of *MovingObjectType*
3: $mo.trajList$ = trajectorySegmentation(lp)
4: **for** each trajectory $ti \in trajList$ **do**
5: creates an instance *tr* of *TrajectoryType*
6: *tr*.eventList = eventSegmentation(ti)
7: **end for**
8: **end for**

4.4.2 Algoritmo de ETL

A utilização de modelagem multidimensional exige um processo de extração, conversão e carga (ETL) a partir dos dados dos esquemas de transações de tempo real (OLTP) para povoamento do esquema do data warehouse. Para o nosso modelo proposto, primeiramente deve haver o povoamento das dimensões (seus níveis hierárquicos) com todas os valores extraídos a partir dos atributos dos objetos *TrajectoryType* e suas relações com os demais objetos do esquema transacional. Por exemplo, a dimensão de tempo deve conter, pelo menos, os níveis referentes ao "tempo de vida" das trajetórias, assim como a dimensão de objetos móveis deve conter todos os níveis hierárquicos da empresa. A dimensão de região é um caso a parte, onde é importante observar que as relações nem sempre irão atender a todos os níveis da hierarquia, ou seja, uma trajetória pode não estar associada somente a partir de um determinado nível da hierarquia mas não estar associado com seus níveis inferiores. Podemos ilustrar com o seguinte caso usando a relação *inside*: uma trajetória que ocorreu no estado de São Paulo mas não completamente dentro de nenhuma de suas cidades. Então, para o povoamento dessa dimensão, além de todas as possibilidades normais, temos que considerar um valor nulo como possibilidade para cada nível. Por exemplo, considerando apenas os níveis de estado e cidade para simplificação, podemos ter [São Paulo - Osasco; São Paulo - null; null - null], onde o primeiro seria ligado com as trajetórias que estão contidas em Osasco e consequentemente no estado de São Paulo, o segundo que estaria contida no estado mas em nenhuma de suas cidades completamente e, o último, que não estaria contida no estado.

Após a realização do povoamento das dimensões, é possível, então realizar o povoamento do fato trajetória e estabelecimento de suas ligações com as dimensões. Para guiar esse processo nós propomos o algoritmo *Populate Multidimensional Schema* (Algorithm 4.4.2) que atua sobre os dados do esquema transacional proposto. O algoritmo é executado sobre as instâncias de *Trajectorytype* (linha 1) estabelecendo para cada uma as respectivas ligações com as dimensões (linhas 3 a 13). Quatro dos cinco links com as dimensões são estabelecidos de maneira similar por comparações: a linha 3, cria dois links com a dimensão de tempo, um para sinalizar o tempo de início da trajetória e o outro o término, os quais são calculados por comparações entre os atributos de instante de trBegin e trEnd da trajetória com os valores dos níveis da dimensão; na linha 4 é criado o link entre o fato e a dimensão de objetos móveis por simples comparação entre os identificadores dos objetos móveis; o link estabelecido na linha 13 se dá pela comparação entre a categoria da dimensão e a categoria da trajetória obtida por uma função que deve ser criada pelo projetista para identificar a categoria da trajetória de acordo com o interesse da aplicação. As linhas de 5 a 12 criam a ligação com a hierarquia assimétrica [25] da dimensão de regiões conforme o problema citado no parágrafo anterior. Para isso

começamos a verificação da relação percorrendo os níveis da hierarquia começando pelo mais baixo e, no caso de não haver uma relação com o nível, este irá ser considerado como valor null. Repete-se o processo até que a relação com um dos níveis seja estabelecida ou até que seja conferido o nível mais alto.

Algorithm 4.4.2 Populate Multidimensional Schema

```
1: for each trajectory ti of TrajectoryType table in transactional schema do
2:    copy ti to the trajectory fact in the multidimensional schema
3:    create two links to the lowest level of Time dimension using ti.trBegin and ti.trtEnd
      properties
4:    create one link to the lowest level of Moving Object dimension using ti.oid;
5:    for each hierarchical level rl of Region dimension ordered in ascending order do
6:       if SpatialTopologicalFunction(ti.geometry,rl.geometry)=true then
7:          create a link to the level rl of Region dimension
8:          break
9:       else
10:         set null to link to the level rl of Region dimension
11:      end if
12:   end for
13:   create one link to the lowest level of Trajectory Category dimension using
      ti.category() function;
14: end for
```

4.5 Funções de agregação para trajetórias

Ao realizar análises sobre dados de trajetórias temos, quase sempre, a necessidade de obter informações sumarizadas de seus atributos, tais como média, soma ou máximo de uma medida. Pode ser requerido, por exemplo, obter a soma dos comprimentos de todas as trajetórias e a velocidade média dos caminhões de uma determinada empresa que ocorreram em um intervalo de tempo especificado e que cruzaram uma dada região. Percebe-se, então, a necessidade de um conjunto de funções de agregação específicas para trajetórias. Isso envolve a dificuldade de trabalhar-se com tipos de dados não convencionais. Estes dados são medidas de trajetória que podem ser atributos numéricos, temporais ou espaciais, tais como o comprimento da trajetória, a posição espacial de início ou a geometria espacial representativa da trajetória.

4.5.1 Assinatura padrão

Nós propomos um conjunto de funções de agregação para trajetórias, que reduzem um conjunto de objetos do tipo trajetória a um elemento de retorno que pode ser um objeto ou conjunto de objetos do tipo trajetória, valor escalar, intervalo de tempo ou

linha espacial. Porém, nós não especificaremos todo o conjunto de funções possíveis pois acreditamos que é responsabilidade do projetista identificar as agregações necessárias e, além disso, seria exaustivo e pouco acrescentaria. Ao invés disso, nós propomos uma assinatura genérica que ajuda a especificar a criação de funções de agregação para trajetórias definida a seguir:

$$OP : T \text{ x } F \rightarrow D \text{ onde}$$

- OP representa operações de agregação. Por exemplo: SUM, COUNT, AVERAGE, MIN, and MAX;
- T é um conjunto de dados de trajetória do tipo proposto *TrajectoryType*;
- F define qual medida da trajetória é utilizada pela operação de agregação OP. Uma medida de trajetória é uma informação que pode ser computada ou obtida a partir de uma trajetória (por exemplo, geometry, length, duration, conjunto de sample points/events, etc). Os possíveis tipos de medidas de trajetória são: TRAJETÓRIA, ESPACIAL, TEMPORAL, ESCALAR e CONJUNTO;
- D é o tipo de dado do retorno.

A tabela 1 apresenta alguns exemplos de funções compostas a partir da assinatura proposta. Nos exemplos, a variável t representa o conjunto de trajetórias do tipo *TrajectoryType* que irão passar pelo processo de agregação. A medida utilizada pela agregação deve ser fornecida por funções implementadas no tipo *TrajectoryType*. Por exemplo, a função t.events("Lunch") retorna um conjunto de objetos do tipo *EventType* que possuem a descrição (i.e. tipo) "Lunch".

4.5.2 Conjunto de funções de agregação para trajetórias

Com base na assinatura para funções de agregação proposta e nosso estudo de caso, nós apresentamos o conjunto de funções de agregação a seguir:

1. TRAGG_COUNT_TR
   - assinatura: COUNT(T,T):ESCALAR;
   - significado: retorna a quantidade de trajetórias;
2. TRAGG_SUM_LENGTH
   - assinatura: SUM(T,T.LENGTH):ESCALAR;

Tabela 1: Exemplos de funções de agregação para trajetórias

SIGNATURE	EXAMPLE	SIGNIFICADO
COUNT(T,EVENTS):ESCALAR	tr_agg_count(t,t.events("Lunch"))	retorna o número de eventos do tipo "Lunch"
AVG(T,DURATION): TEMPORAL	tr_agg_avg(t,t.duration)	retorna a média dos intervalos de duração
MAX(T,LINE): TRAJECTORY	tr_agg_max(t,t.trline)	retorna a trajetória com maior comprimento
MAX(T,SPEED): TRAJECTORY	tr_agg_max(t,t.speed)	retorna a trajetória com maior velocidade média
SUM(T,EVENTS): TEMPORAL	tr_agg_sum_duration (t,t.events("Atendimento"))	retorna a soma da duração dos eventos do tipo "Atendimento"
MIN(T,LINE): ESPACIAL	tr_agg_min(t,t.trline)	retorna a linha espacial com menor comprimento

- significado: retorna a soma do comprimento das trajetórias;

3. TRAGG_SUM_DURATION

   - assinatura: SUM(T,T.DURATION):ESCALAR;
   - significado: retorna a soma da duração das trajetórias;

4. TRAGG_AVG_SPEED

   - assinatura: AVG(T,T.SPEED):ESCALAR
   - significado: retorna a velocidade média das trajetórias;

5. TRAGG_COUNT_ATENDIMENTO

   - assinatura: COUNT(T,T.EVENTS("Atendimento")):ESCALAR;
   - significado: retorna a quantidade de eventos do tipo Atendimento;

6. TRAGG_COUNT_EXCESSOVELOCIDADE

   - assinatura: COUNT(T,T.EVENTS("Excesso de velocidade")):ESCALAR;
   - significado: retorna a quantidade de eventos do tipo Excesso de velocidade;

7. TRAGG_AVG_ATENDIMENTO

   - assinatura: AVG(T,T.EVENTS("Atendimento")):ESCALAR
   - significado: retorna a média da quantidade de eventos do tipo Atendimento por trajetória;

8. TRAGG_SUM_PARADA-DURATION
    - assinatura: SUM(T,T.EVENTS(“Parada”).DURATION):ESCALAR;
    - significado: retorna a soma da duração dos eventos do tipo Parada;
9. TRAGG_AVG_PARADA-DURATION
    - assinatura: AVG(T,T.EVENTS(“Parada”).DURATION):ESCALAR
    - significado: retorna a média da duração de tempo parado por trajetória;
10. TRAGG_SUM_ATENDIMENTO-VALUE
    - assinatura: SUM(T,T.EVENTS(“Atendimento”).VALUE1):ESCALAR;
    - significado: retorna a soma do valor associado aos eventos do tipo Atendimento;
11. TRAGG_AVG_ATENDIMENTO-VALUE
    - assinatura: AVG(T,T.EVENTS(“Atendimento”).VALUE1):ESCALAR
    - significado: retorna a média do valor associado aos eventos do tipo Atendimento por trajetória;
12. TRAGG_AVG_ATENDIMENTO-VALUE_CLI
    - assinatura: AVG(T,$\frac{SUM(T,T.EVENTS(\text{“}Atendimento''\text{).}VALUE1):ESCALAR}{COUNT(T,T.EVENTS(\text{“}Atendimento''\text{)):}ESCALAR}$):ESCALAR;
    - significado: retorna a média do valor associado aos eventos do tipo Atendimento por cliente;
13. TRAGG_AVG_ATENDIMENTO-VALUE_LENGTH
    - assinatura: AVG(T,$\frac{SUM(T,T.EVENTS(\text{“}Atendimento''\text{).}VALUE1):ESCALAR}{SUM(T,T.LENGTH):ESCALAR}$):ESCALAR;
    - significado: retorna a média do valor associado aos eventos do tipo Atendimento por distância;
14. TRAGG_AVG_ATENDIMENTO-VALUE_DURATION
    - assinatura: AVG(T,$\frac{SUM(T,T.EVENTS(\text{“}Atendimento''\text{).}VALUE1):ESCALAR}{SUM(T,T.DURATION):ESCALAR}$):ESCALAR;
    - significado: retorna a média do valor associado aos eventos do tipo Atendimento por duração;
15. TRAGG_COUNT_RESTRICTENTRY
    - assinatura: COUNT(T,T.EVENTS(“Entrada restrita”)):ESCALAR;
    - significado: retorna a quantidade de eventos do tipo Entrada restrita;

5 EXPERIMENTOS

Neste capítulo iremos apresentar experimentos realizados em um cenário real de aplicação conforme descrito na seção 2.3. Implementamos um esquema transacional e um multidimensional instanciados a partir dos meta-esquemas apresentados como contribuição (seções 4.2 e 4.3) em banco de dados relacional com extensão de recursos OR. Em seguida implementamos os algoritmos propostos (seção 4.4) para construção e povoamento das instâncias dos esquemas. E, após isso, desenvolvemos algumas funções de agregação para atuar nos dois ambientes. Além disso, nós fizemos uso de uma ferramenta open source de BI para OLAP para atuar na construção e navegação em cubos de dados a partir do esquema multidimensional implementado. E, finalmente, nós apresentamos algumas consultas e análises obtidas, a partir de requisitos reais levantados na empresa X, sobre os dois ambientes implementados para demonstrar a importância de nossas contribuições. Ilustramos os dois esquemas trabalhando em paralelo, onde teremos o transacional obtendo informações em tempo real e o multidimensional obtendo as mesmas informações agregadas em diferentes níveis sobre o histórico da empresa para obtermos algum padrão de comparação para suporte a decisões estratégicas e estimativas de previsão.

Para realização dos experimentos de forma fiel aos ambientes corporativos reais, é importante ressaltar que, para os experimentos feitos no ambiente transacional, consideramos que cada base - neste caso o nível mais baixo da hierarquia organizacional - irá possuir seu próprio ambiente transacional e os exemplos da seção 5.2.4 são realizados nesse nível. Somente no ambiente multidimensional, sobre o qual são realizados os experimentos da seção 5.3.4, consideramos a união dos dados de todos os ambientes transacionais da empresa. Além disso, consideramos que, para manter eficiência nos ambientes transacionais, os dados do mesmo são carregados para o ambiente multidimensional, o qual é otimizado para lidar com grandes volumes de dados e realização de análises envolvendo agregações.

5.1 Conjunto de dados

O conjunto de dados brutos utilizado em nossos experimentos é oriundo de uma empresa real e foram coletados ao longo do período de pouco mais de um ano a partir de 251 veículos de distribuição, apresentando mais de 21 milhões e quinhentas mil tuplas. O formato destes dados segue a descrição apresentada na seção 2.3. À parte destes dados, utilizamos um outro conjunto de dados brutos que apresenta marcações de acontecimentos que foram assinaladas pelos motoristas em equipamentos acoplados ao veículo ou detectados pelo sistema da empresa, tais como atendimento, excesso de velocidade e parada. Este outro conjunto de dados apresenta dois atributos para coordenada (latitude e longitude), duas marcações de tempo para e início e término, tipo do evento e o código do veículo que originou o evento. Estes dados são a base para a obtenção de informações temáticas das trajetórias e seus eventos, conforme prevê o meta-esquema, possuindo mais de um milhão e trezentas mil tuplas.

5.2 Implementação do esquema transacional de trajetórias

Para projeto do esquema transacional proposto, primeiramente foi necessário definir uma maneira de representação do mesmo em banco de dados e, devido à natureza complexa dos dados de trajetórias e à necessidade de alto nível de abstração capaz de preservar a visão de trajetória como objeto de primeira ordem, a solução adotada deveria possuir característica do paradigma de orientação a objetos. Essa orientação fornece, ainda, uma boa compreensão do esquema e facilidade para elaboração de consultas complexas. Outro fato muito importante considerado no desenvolvimento do projeto foi a imensa quantidade de dados a serem armazenados e gerenciados de forma eficiente e, para isso, a melhor solução ainda são os banco de dados relacionais. Levando em consideração, principalmente, esses dois argumentos e embasamento em alguns trabalhos na literatura da área (e.g. HERMES e SECONDO) foi decidido a adoção de extensão objeto-relacional presente em SGBDs para usufruir de seus mecanismos otimizados de linguagens de consultas bem como de suas extensões para estruturas espaciais e índices eficientes.

5.2.1 Esquema

Com base no meta-esquema para ambiente transacional proposto na seção 4.2 e sua extensão com entidades especificas referentes ao estudo de caso, a Figura 8 exibe o esquema lógico do ambiente transacional projetado e a Figura 9 detalha os principais tipos utilizados na implementação. Foi, então, criada uma implementação em SGBD OR

(Oracle 11g[1]). A Figura 10 apresenta o esquema implementado[2], para o qual os recursos OR e espaciais oferecidos pelo SGBD facilitaram a criação e implementação das estruturas de dados para os tipos complexos propostos bem como seus métodos e funções.

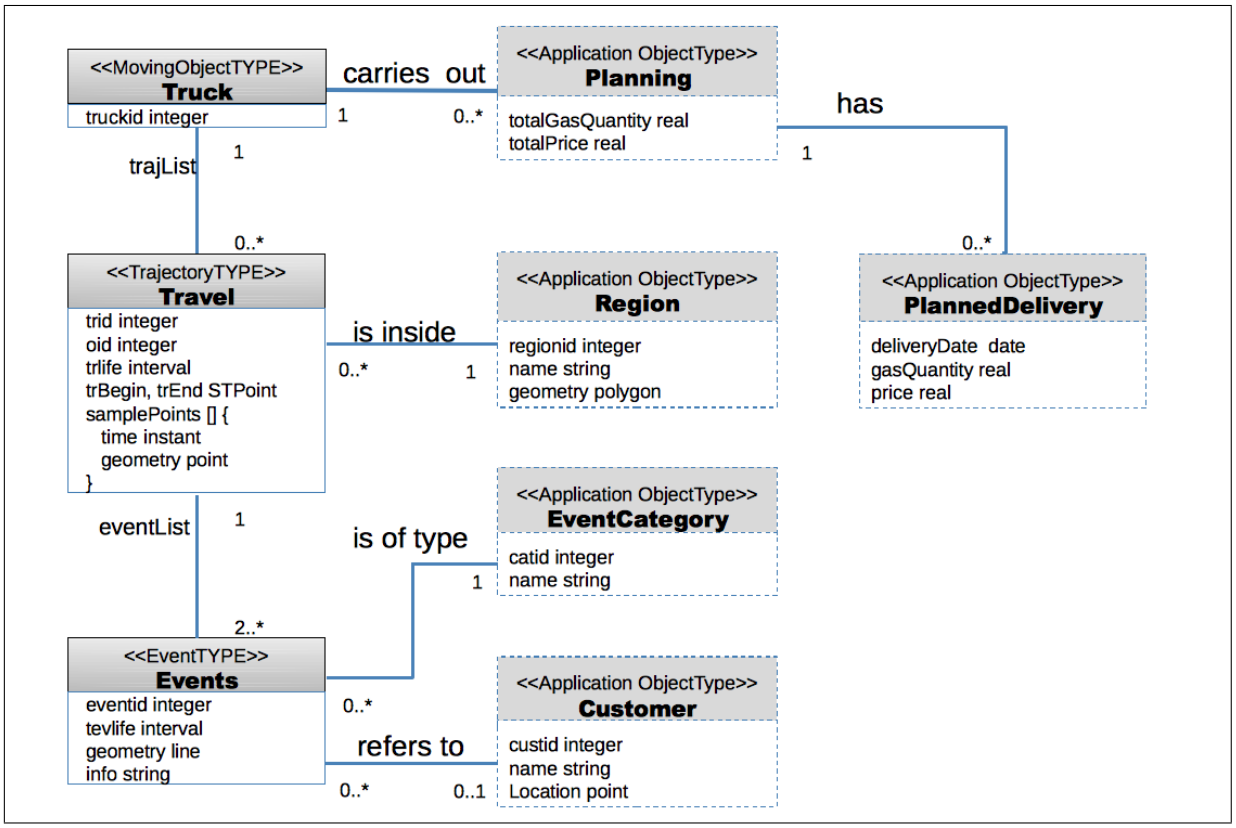


Figura 8: Projeto do esquema transacional para trajetórias em SGBD

5.2.2 Considerações sobre a implementação

Com o objetivo de evitar reprocessamento, foram criados atributos nos tipos para materialização dos resultados de alguns métodos no tipo trajetória (e.g. length, average speed, duration) e no tipo evento (e.g. duration), evitando, assim, reprocessamento dos mesmos uma vez que a trajetória esteja concluída e não mais irá sofrer atualizações em sua estrutura. Um outro aspecto de implementação considerado foi a representação dos relacionamentos, principalmente as coleções de objetos, para as quais foi adotado o uso de tabelas aninhadas (*nested tables*) objetivando obter uma maior fidelidade com a visão do paradigma de orientação a objetos, obtendo uma relação de entendimento mais direto e intuitivo com o mundo real do que o modelo relacional puro e que, além disso, permite maior abstração para criação de consultas. Ainda, para melhor desempenho e evitar carga desnecessária de objetos, as tabelas aninhadas possuem apenas as referências para os objetos da coleção que representa, ou seja, são tabelas aninhadas de referências.

[1] referenciar extensao OR e espacial

[2] algumas entidades (e.g. área) foram omitidos para melhor clareza da imagem

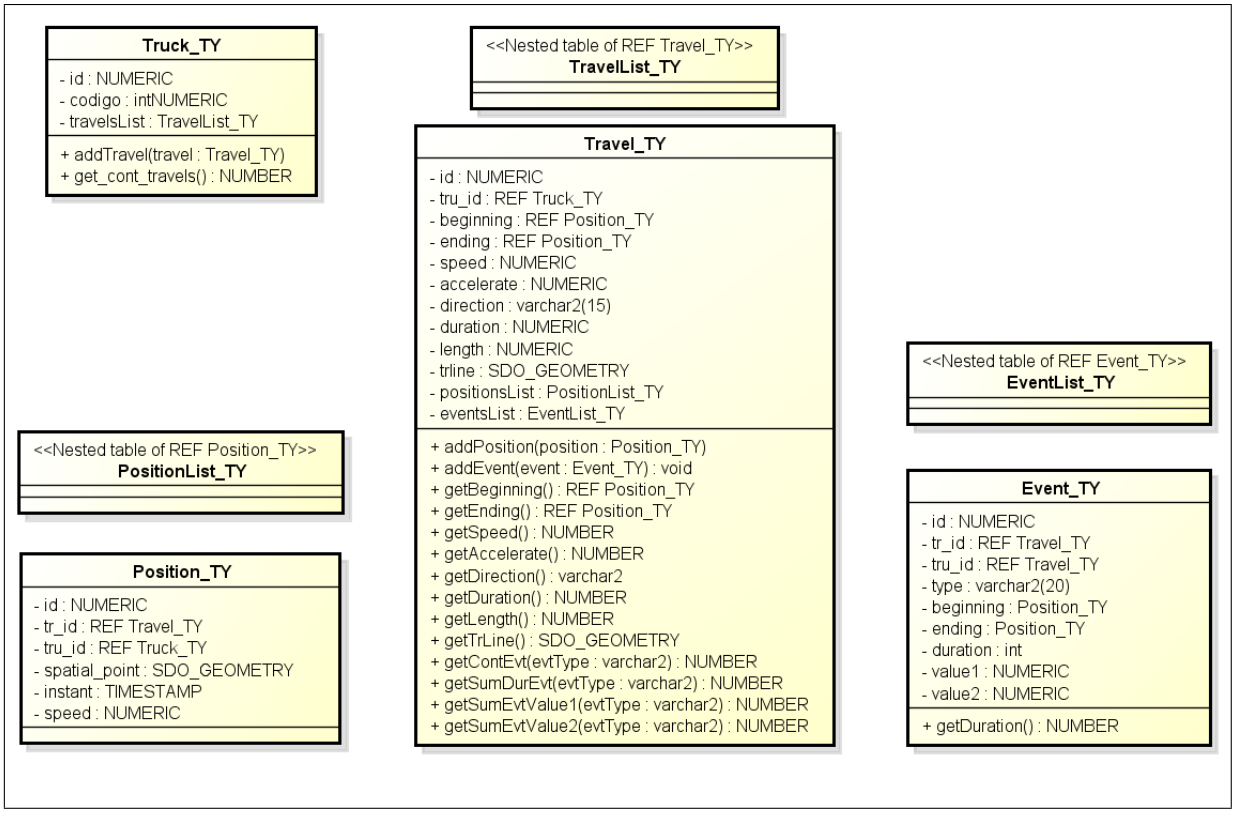


Figura 9: Principais tipos implementados para o ambiente transacional em SGBD OR Oracle

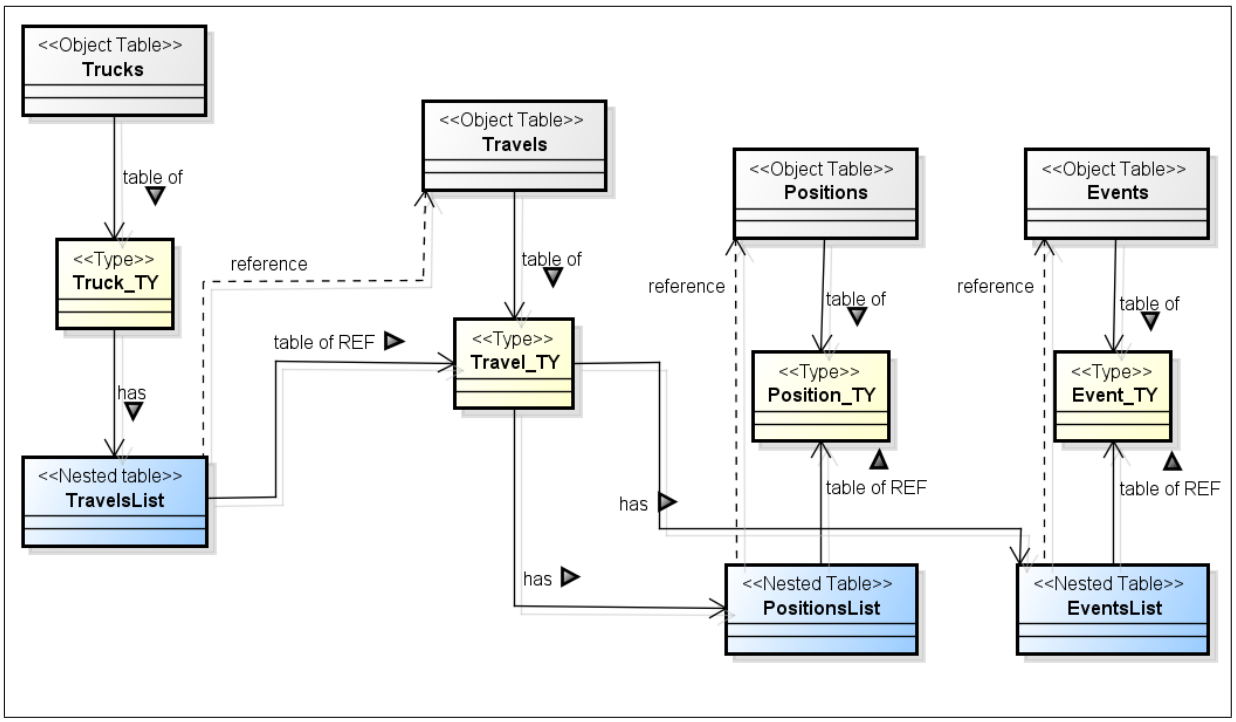


Figura 10: Esquema lógico da implementação em SGBD OR Oracle

5.2.3 Povoamento do esquema

A criação dos objetos e o respectivo povoamento das tabelas se deu através da implementação em PL/SQL do algoritmo 4.4.1 proposto na seção 4.4.1 através de proce-

dimentos (i.e. Oracle procedures) que representam as etapas do algoritmo. Os códigos para os procedimentos são apresentados no Apêndice A.

5.2.4 Exemplos de análises

De acordo com os requisitos levantados em nosso estudo de caso, iremos apresentar alguns exemplos de consultas que utilizam o ambiente transacional para obtenção de informações para acompanhamento e monitoramento dos veículos em tempo real e consultas para obtenção de informações para apoio à decisão. Em ambos os casos é importante ressaltar que as informações obtidas podem ser comparadas com o resultado quando aplicado sobre um conjunto mais geral como base de referência para a comparação.

Como mencionado anteriormente na descrição do cenário de nosso estudo de caso, a empresa X paga pelas entregas de acordo com as horas de duração das viagens (atendimento aos clientes do cronograma). Devido a isso o tempo que o veículo permanece parado é importante em dois casos: *(i)* com a finalidade de monitoramento do veículo em tempo real, pois é importante obter o tempo que o caminhão permanece parado na viagem corrente para controle de segurança, imprevistos e, até mesmo, realização de atividades ilícitas como desvio de produto; *(ii)* para fim de análise de custo é importante conseguir obter para os caminhões a média do tempo que permanecem parados por viagem. Apresentamos, então, na Figura 11 uma consulta para o primeiro caso e, na Figura 12 uma consulta para o segundo.

```
select  t.get_sum_dur_evt('Parada')
from  trucks tru, table(tru.travels) t
where tru.codigo = 2 and
 (t.beginning.instant,'DD-MM-YYYY')=(current_date,  'DD-MM-YYYY');
```

Figura 11: Consulta para obter o tempo que um dado veículo permaneceu parado em uma viagem

```
select tru.codigo, TRAGG_SUM_PARADA_DURATION(t)
from trucks tru, table(tru.travels) t
group by tru.codigo;
```

Figura 12: Consulta para obter a média de tempo que cada veículo permaneceu parado por viagem

A empresa tem interesse em acompanhar as atividades do veículo de entrega em tempo real e, para isso, é necessário obter o percurso do mesmo em forma de linha espacial bem como determinados tipos de eventos decorridos para plotagem em mapa obtendo, assim, visualização para monitoramento. Exibimos na consulta da Figura 13 *a*

uma consulta para obter o percurso do caminhão e na Figura 13 *b* a consulta que obtém os locais dos eventos do tipo atendimento. Em seguida é exibido a plotagem do resultado dessas consultas no Google Maps[3] na Figura 14.

```
(a)
select   t.get_trline()
from   trucks tru, table(tru.travels) t
where tru.codigo = 11 and
   (t.beginning.instant, 'DD-MM-YYYY')=(current_date, 'DD-MM-YYYY');

(b)
select   e.beginning.spatial_point
from   trucks tru, table(tru.travels) t, table(t.events) e
where tru.codigo = 11 and
   (t.beginning.instant, 'DD-MM-YYYY')=(current_date, 'DD-MM-YYYY')
   and e.type = 'Atendimento';
```

Figura 13: Consultas para obter o percurso de um veículo em sua viagem corrente (Consulta *a*) e os locais dos eventos de atendimento (Consulta *b*)

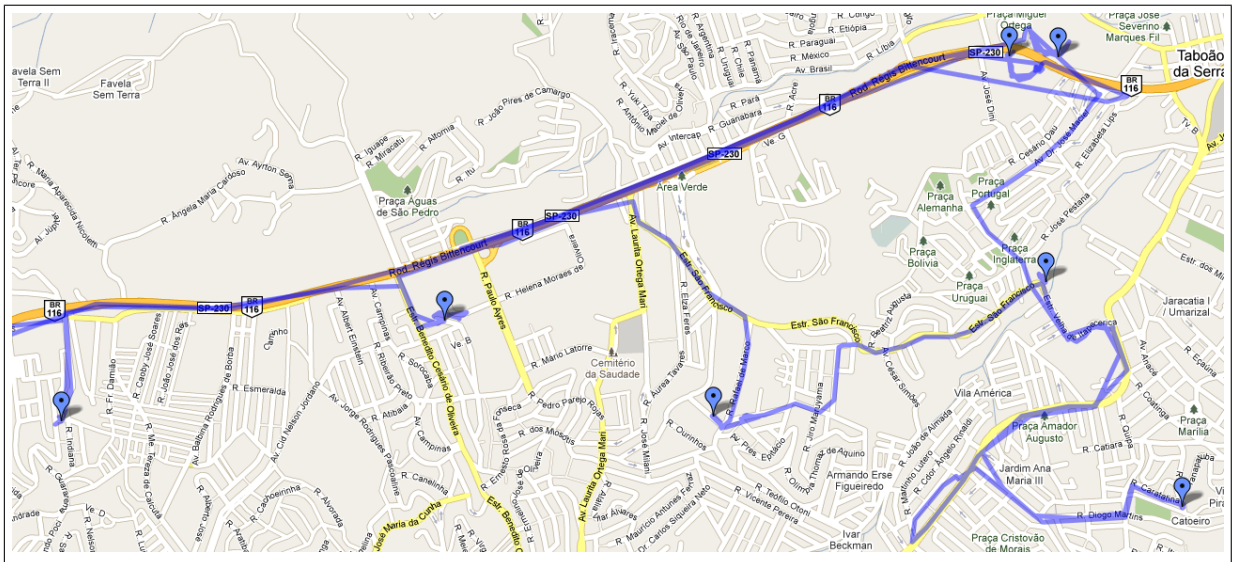


Figura 14: Visualização do resultados das consultas da Figura 13 no Google Maps

Uma outra informação de interesse levantada em nosso cenário de aplicação - no contexto de análise - é obter a média da quantidade de atendimentos de cada veículo por viagens realizadas em uma determinada região (e.g. estado de São Paulo) e, ainda, a média da quantidade entregue por cliente. A consulta que retorna essas informações é apresentada na Figura 15.

Para análise de custo benefício, um outro requisito levantado foi a necessidade em obter a relação entre a quantidade entregue em uma determinada viagem e sua extensão

[3]referencia

```
select tru.codigo, TRAGG_CONT_ATENDIMENTO(t),
       TRAGG_AVG_ATENDIMENTO_VALUE_CLI(t)
from trucks tru, table(tru.travels) t, areas a
where a.name = 'SAO_PAULO' and a.type = 'STATE' and
      sdo_inside(t.trline, a.region) = 'TRUE'
group by tru.codigo;
```

Figura 15: Consulta para obter de cada veículo e suas viagens realizadas em uma dada região a quantidade média de atendimentos por viagem e a média da quantidade entregue por cliente

ou a relação entre a quantidade entregue e sua duração. Esses dois requisitos são ilustrados na Figura 16. Em um contexto de análise mais geral, é necessário obter a média dessas mesmas relações para cada caminhão em um determinado período e em uma determinada região, conforme exibido na Figura 17.

```
select TRAGG_AVG_ATENDIMENTO_VALUE_LENGTH(t),
       TRAGG_AVG_ATENDIMENTO_VALUE_DURATION(t)
from trucks tru, table(tru.travels) t
where tru.codigo = 2 and
      (t.beginning.instant, 'DD-MM-YY') = '14-06-10';
```

Figura 16: Consulta para obter as médias entre quantidade entregue e comprimento do percurso e entre quantidade entregue e duração para uma viagem

```
select tru.codigo, TRAGG_AVG_ATENDIMENTO_VALUE_LENGTH(t),
       TRAGG_AVG_ATENDIMENTO_VALUE_DURATION(t)
from trucks tru, table(tru.travels) t, areas a
where a.name = 'OSASCO' and a.type = 'CITY' and
  sdo_inside(t.trline, a.region) = 'TRUE' and
  t.beginning.instant >= to_timestamp('01-01-10', 'DD-MM-YY') and
  t.ending.instant <= to_timestamp('30-06-10', 'DD-MM-YY')
group by tru.codigo;
```

Figura 17: Consulta para obter as médias da quantidade entregue pela distância percorrida e a da quantidade entregue pela duração para cada caminhão em um dado período e em uma dada área

Para monitoramento e análise sobre os veículos, é importante saber em uma viagem quanto tempo os mesmos permaneceram dentro de uma determinada área como, por exemplo, uma base de distribuição e, assim, ter conhecimento de quanto tempo foi gasto na mesma (Figura 18 *a*) e não efetivamente realizando a viagem. Para realizar isso em nossa implementação, criamos uma função de corte que recebe como parâmetros uma trajetória e uma área. Esta função retorna uma nova instância de trajetória a partir dos pontos da trajetória contidos na região e, a partir desta nova instância, obtemos as

informações desejadas em alto nível - neste caso a duração. Partindo desse ponto para uma situação mais ampla de análise, podemos descobrir o total de tempo gasto por cada caminhão enquanto estes permanecem nas bases em um período de tempo especificado (Figura 18 *b*), o que pode auxiliar, por exemplo, nas decisões sobre a viabilidade de investimentos em melhorias e ampliação.

```
(a)
select  trajectory_recort_area(t, b.region).get_duration()
from  trucks tru, table(tru.travels) t, bases b
where tru.codigo = 11 and
      to_char(t.beginning.instant) = '15-07-10' and
      b.codigo = 1003;

(b)
select  tru.codigo,
        TRAGG_SUM_DURATION(trajectory_recort_area(t, b.region))
from  trucks tru, table(tru.travels) t, bases b
where t.beginning.instant >= to_timestamp('01-07-10') and
      t.ending.instant >= to_timestamp('31-12-10')
group by tru.codigo;
```

Figura 18: Consultas para obter de uma viagem a duração que o veículo passou dentro de uma área dada (Consulta *a*) e para obter o tempo gasto por cada caminhão dentro das bases em um dado período (Consulta *b*)

As relações entre as viagens (trajetórias) e áreas é fundamental não só para análises mais detalhadas por locais específicos, mas também para verificação de restrições que podem ser impostas aos veículos devido à riscos para o mesmo ou de áreas que devem ser evitadas pois em caso de acidente é maior o risco para a população. Pode ser verificado, por exemplo, qual determinada área de uma cidade possui ruas muito estreitas para tráfego de veículos de grande porte ou que seja uma área densamente povoada por escolas. Nestes casos, é requerido que se verifique se, em algum momento, alguma das viagens correntes possui qualquer interseção com essas áreas explicitando o veículo, a viagem e a área de ocorrência. Um exemplo de consulta realizada para requisitos como estes pode ser visto na Figura 19 *a*. Utilizando, novamente, funções de agregação para trajetórias podemos obter informações analíticas para esse requisito: recuperar para cada caminhão, a quantidade de percursos que entraram em áreas restritas durante um intervalo de tempo especificado, conforme ilustrado na Figura 19 *b*. Para essa agregação foi necessário, em nossa implementação, criar uma função auxiliar que recebe como entrada uma trajetória e retorna um valor numérico que representa a quantidade de áreas restritas com a qual o percurso possui interação. Função esta que é chamada em cada iteração da agregação para a trajetória de contexto.

```
(a)
select tru.codigo, t.tr_id, r.name
from trucks tru, table(tru.travels) t, restrictedAreas r
where SDO_ANYINTERACT(t.trline, r.region) = 'TRUE' and
      t.beginning.instant >= to_timestamp('15-11-10');

(b)
select tru.codigo, tr_count_restrictEntry(t)
from trucks tru, table(tru.travels) t
where t.beginning.instant >= to_timestamp('01-01-10') and
      t.ending.instant <= to_timestamp('30-02-10')
group by tru.codigo;
```

Figura 19: Consultas para obter o veículo, sua trajetória e a área de ocorrência de violação de espaço restrito ocorridos em viagens correntes (Consulta *a*) e para obter a quantidade de viagens de cada caminhão que entraram em áreas restritas

Um dos aspectos mais interessantes levantados com relação ao monitoramento das viagens de entrega, diz respeito a sequências específicas de eventos decorridos durante a mesma para descobrir em quais estão ocorrendo certos padrões dados ou à quantidade de ocorrências de certos eventos. A sequência com um evento do tipo Parada entre dois eventos do tipo Atendimento deve ser possível de ser descoberta em tempo real para checagem do motivo dessa parada (lembrando que é de grande importância para a empresa a segurança do produto estocado no veículo e também a eficiência das viagens). A consulta da Figura 20 mostra como essas informações podem ser obtidas. Podemos, ainda, combinar a detecção de sequência de eventos com funções de agregação para comparar resultados de análises onde o padrão ocorre com o resultados onde o mesmo não acontece e, assim, ter melhor embasamento para verificação da relevância do padrão analisado. Uma outra possibilidade de análise é combinar função de agregação para trajetórias com padrão de eventos que se baseiam na quantidade de ocorrências dos mesmos. Nesse contexto, duas análises necessárias foram marcadas como importantes pela empresa: *(i)* exibir para cada veículo e suas viagens que possuem a sequência mencionada, a média de tempo parado por viagem (Figura 21 *a*); *(ii)* exibir para cada caminhão a velocidade média de suas viagens que possuem mais de cinquenta eventos do tipo Excesso de velocidade cada (Figura 21 *b*).

```
select tru.codigo, t.tr_id
from trucks tru, table(tru.travels) t
where t.beginning.instant >= to_timestamp('15-07-10') and
  traj_evt_seq(t,['Atendimento','Parada','Atendimento'])='TRUE';
```

Figura 20: Consulta para obter os veículos e suas trajetórias correntes nas quais ocorrem um determinado padrão de sequência de eventos

```
(a)
select tru.codigo, TRAGG_AVG_PARADA_DURATION(t)
from trucks tru, table(tru.travels) t
where
  traj_evt_seq(t,['Atendimento','Parada','Atendimento'])='TRUE'
group by tru.codigo;

(b)
select tru.codigo, TRAGG_AVG_SPEED(t)
from trucks tru, table(tru.travels) t
where t.get_cont_evt('Atendimento') >= 50
group by tru.codigo;
```

Figura 21: Consultas para obter para cada veículo a média de tempo parado por viagem para as viagens que possuem uma determinado padrão de sequência de eventos (Consulta *a*) e para obter para cada caminhão a velocidade média de suas trajetórias que possuem um determinado padrão de quantidade de eventos (Consulta *b*)

5.3 Trajectory DW implementation

Para demonstrar e validar o modelo multidimensional proposto, foi projetado um esquema de data warehouse de trajetórias através da extensão do padrão proposto na seção 4.3, o qual, assim como o transacional, foi feito buscando alto nível de abstração para preservar a visão de trajetória como entidade única de maior importância.

5.3.1 Esquema

O projeto para o esquema aparece na Figura 22, no qual os níveis hierárquicos foram definidos de acordo os requisitos levantados da aplicação para nosso estudo de caso. O link entre os fatos evento e trajetória representa trajetória como uma dimensão degenerada de evento. Os fatos possuem, cada um, duas ligações com a dimensão de tempo que representam as marcações temporais de início e término, uma ligação com a dimensão de objetos móveis e uma com a dimensão de regiões. A relação entre os fatos e a dimensão de região é uma decisão que deve ser tomada de acordo com os requisitos do domínio de aplicação e, para nossos experimentos, foi usada a relação *inside* - várias relações podem ser usadas simultaneamente, sendo necessário criar para cada uma uma ligação com a dimensão. Cada fato possui, também, uma ligação com uma dimensão temática que o classifica de acordo com o tipo.

O esquema de implementação foi instanciado, também, em banco relacional com recursos objeto-relacional, mas poderia ter sido realizado em qualquer modelo escolhido - é uma questão que cabe ao projetista. Essa decisão se deu para alcançar maior fidelidade

e facilidade de mapeamento entre modelo e implementação e, também, no mapeamento durante o processo de ETL a partir do ambiente transacional. Os tipos implementados bem como o esquema lógico de implementação realizado são detalhados nas Figuras 23 e 24 respectivamente.

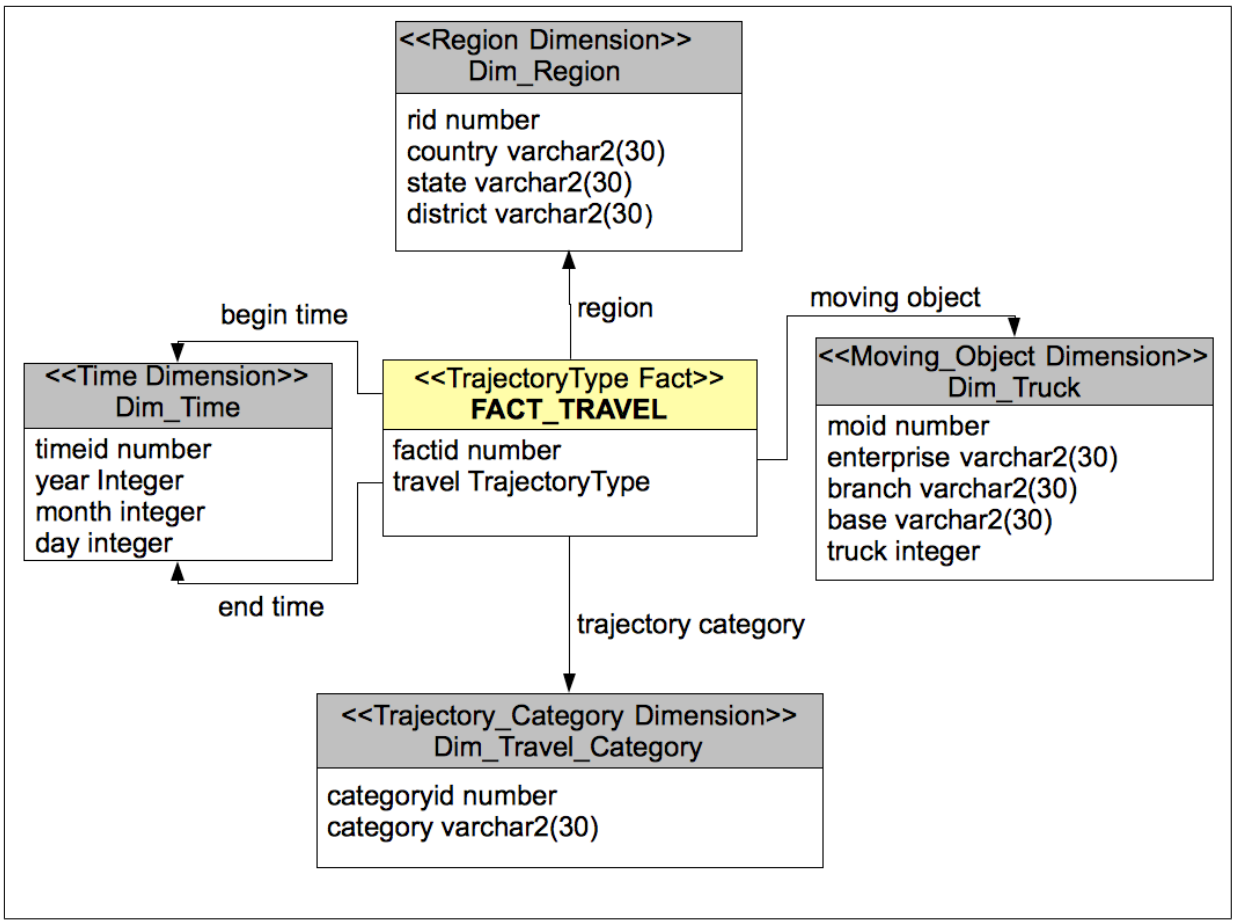


Figura 22: Projeto do DW de trajetórias em SGBD

5.3.2 Considerações sobre a implementação

Conforme mencionado na seção 4.4.2, a relação dos fatos com a dimensão de região nem sempre vai atender todos os níveis hierárquicos definidos. Exemplificando com a relação *inside* usada em nossos experimentos, uma viagem pode estar contida na região Sudeste mas não estar totalmente contida em um de seus estados e, para contornar essa limitação, a dimensão foi povoada com todas as possibilidades considerando a existência de níveis hierárquicos nulos (e.g. SUDESTE > null ; SUDESTE > SAO PAULO).

Os tipos *DW_Travel_TY* e *DW_Event_TY* são simplificações dos tipos *Travel_TY* e *Event_TY* na qual as medidas dos atributos complexos são decompostas diretamente como atributos. Esta decomposição é realizada para permitir a manipulação dos atributos

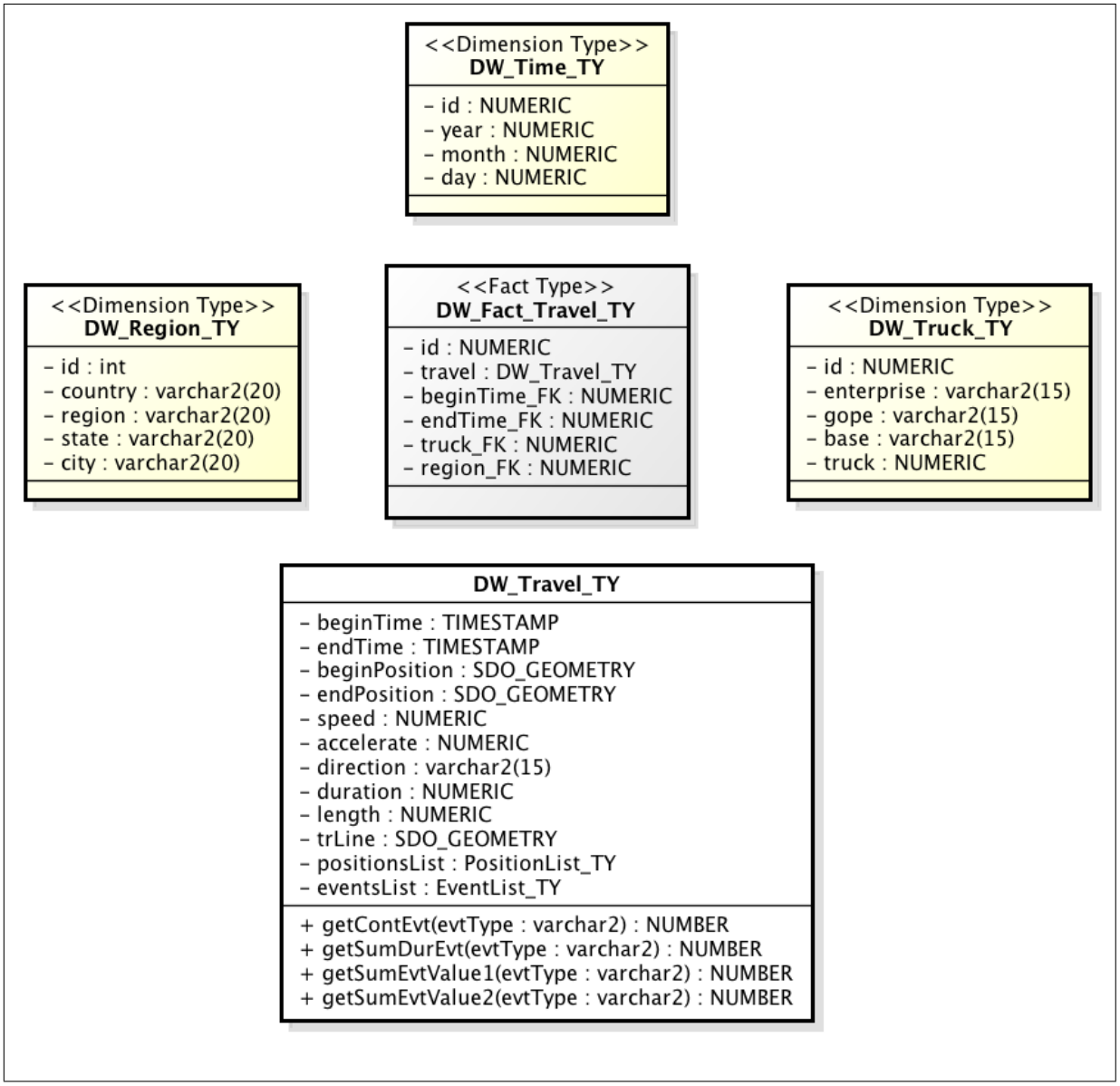


Figura 23: Tipos implementados para o ambiente multidimensional em SGBD OR Oracle

de forma direta por ferramentas OLAP (e.g. Mondrian[4]) pois as mesmas ainda não possuem suporte para manipulação de atributos de tipos complexos definidos pelo usuário;

5.3.3 Processo ETL

De acordo com o algoritmo proposto na seção 4.4.2 para povoamento do esquema multidimensional a partir do esquema transacional, criamos uma versão correspondente para o mesmo através de linguagem PL/SQL dentro do próprio SGBD. Os códigos para este encontram-se no apêndice B.

[4]link de referencia

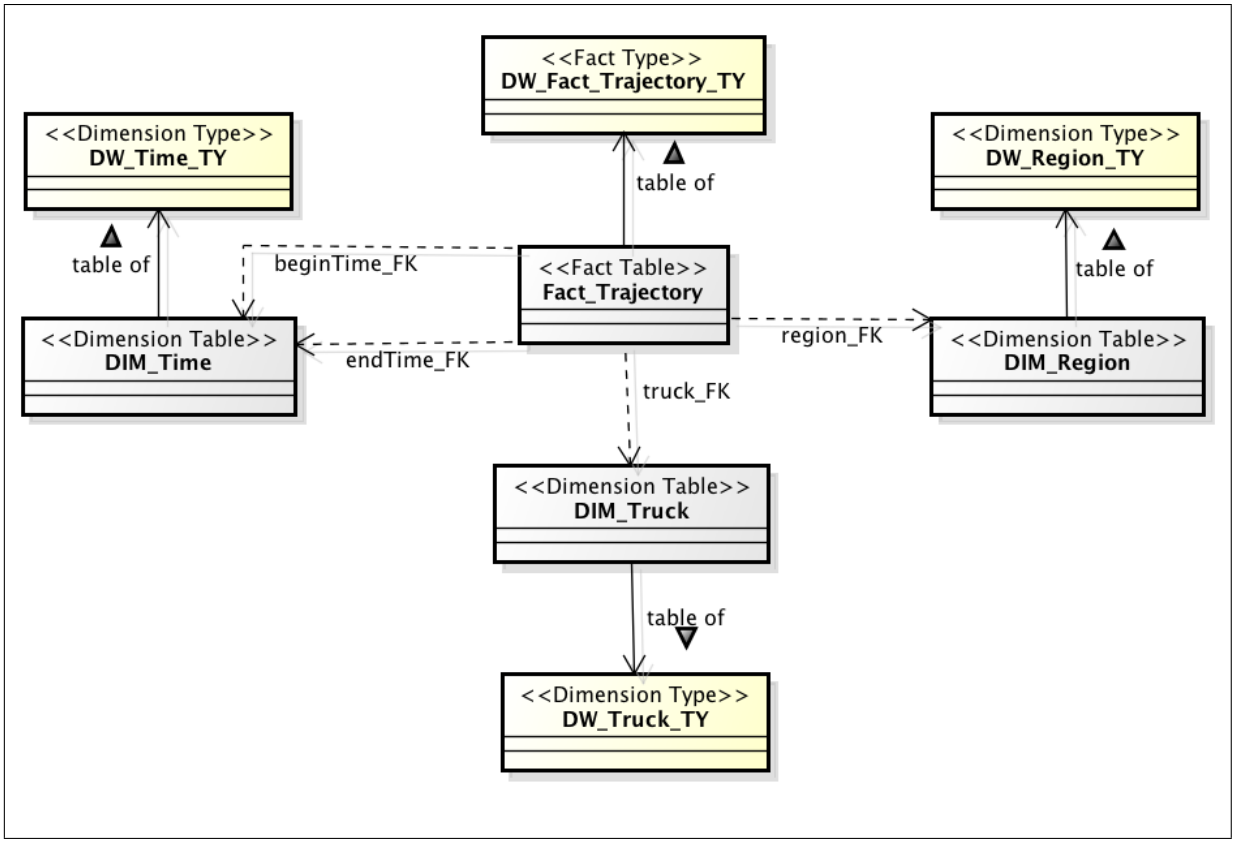


Figura 24: Esquema lógico da implementação do DW em SGBD OR Oracle

5.3.4 Exemplos de análises

Com base nos requisitos e exemplos executados na seção 5.2.4, iremos ilustrar a execução dos mesmos no esquema multidimensional sobre todos os dados do histórico para serem utilizados como base de comparação com análises feitas em tempo real. Iremos exibir, também, consultas para obter análises que exibem as informações desejadas em qualquer nível de detalhe que podem ser usadas para suporte a decisões estratégicas ou geração de relatórios. E, ainda, iremos mostrar a criação de cubos para navegação através de uma ferramenta de BI para OLAP.

Considerando o requisito da consulta da Figura 12 para obter a média do tempo parado por viagem para cada caminhão, é possível obter essa informação sobre todo o histórico dos dados e, consequentemente, com maior precisão de forma que podemos comparar com os dados obtidos em consultas de tempo real (e.g. Figura 11). Esse requisito executado sobre o esquema multidimensional aparece na Figura 25. Utilizando outras dimensões e operadores OLAP do banco em cláusulas *group by* conseguimos obter maior capacidade analítica dos resultados, como ilustrado na consulta da Figura 26 e o seu resultado na Tabela 2 onde, além da média de tempo parado por viagem, apresenta-se a informação agregada por estados e bases com operador *roll-up* que agrega os dados dos níveis mais baixos para compor os níveis superiores. No resultado da tabela consideramos

somente os estados e bases da região Sudeste para clareza do resultado. As linhas 11, 16 e 19 apresentam a média para cada estado (segundo nível de agregação) considerando as viagens de todas as suas bases; a linha 23 considera todas as viagens realizadas na região sudeste considerando todos os seus estados e bases (nível mais alto da agregação); as demais linhas apresentam as informações no nível mais baixo de agregação considerando cada estado e cada uma de suas bases.

```
select dt.codigo, TRAGG_SUMPARADA_DURATION(ft.dw_trajectory)
from fact_travel ft, dim_truck dt
where ft.truck_fk = dt.id
group by dt.codigo;
```

Figura 25: Consulta para obter a média do tempo de parada por viagem de cada caminhão

```
select   dr.state, dt.base, count(ft),
         TRAGG_SUM_PARADA_DURATION(ft.travel)
from   fact_travel ft, dim_truck dt, dim_region dr
where  ft.truck_fk = dt.id and
       ft.region_fk = dr.id and
       dr.region = 'SUDESTE'
group by rollup(dr.state, dt.base);
```

Figura 26: Consulta para obter a média do tempo de parada por viagem utilizando operadores OLAP do SGBD

As informações obtidas com a consulta da Figura 15 que recupera a quantidade média de atendimentos e a quantidade média de produto entregue por cliente em uma determinada região sobre o conjunto de dados do histórico são recuperadas pela consulta realizada na Figura 27.

```
select   tru.codigo, TRAGG_CONT_ATENDIMENTO(ft.travel),
         TRAGG_AVG_ATENDIMENTO_VALUE_CLI(ft.travel)
from   fact_travel ft, dim_truck dt, dim_region dr
where  ft.truck_fk = dt.id and
       ft.region_fk = dr.id and
       dr.state = 'SAO_PAULO'
group by tru.codigo;
```

Figura 27: Consulta para obter para cada veículo a média da quantidade de atendimentos por viagem e da quantidade de produto entregue por cliente para uma determinada região

Podemos exibir uma visão mais analítica dos resultados através da agregação das informações obtidas e aplicação de operadores OLAP. A consulta da Figura 28, que faz uso das dimensões de tempo e de veículos, é um exemplo de requisito para suporte analítico.

Tabela 2: Resultado da consulta da Figura 26

row	**STATE**	**BASE**	**QTD_TRAVEL**	**AVG_STOP**
1	SAO PAULO	25	155	253
2	SAO PAULO	27	639	940
3	SAO PAULO	28	14	396
4	SAO PAULO	1002	4238	276
5	SAO PAULO	1003	7204	290
6	SAO PAULO	1007	629	622
7	SAO PAULO	1017	122	213
8	SAO PAULO	2010	71	417
9	SAO PAULO	2012	1	0
10	SAO PAULO	3002	10	115
11	SAO PAULO	**TOTAL**	13083	333
12	MINAS GERAIS	1002	1	66
13	MINAS GERAIS	1003	11	723
14	MINAS GERAIS	1015	266	445
15	MINAS GERAIS	1020	3	603
16	MINAS GERAIS	**TOTAL**	281	456
17	ESPIRITO SANTO	1015	3	882
18	ESPIRITO SANTO	1019	11	349
19	ESPIRITO SANTO	**TOTAL**	14	463
20	RIO DE JANEIRO	1003	56	614
21	RIO DE JANEIRO	1017	417	484
22	RIO DE JANEIRO		473	500
23	**TOTAL**	**TOTAL**	14218	339

Nela conseguimos as informações desejadas em todos os níveis especificados - neste caso em relação aos gopes (termo utilizado como sinônimo de *branch* no esquema de implementação multidimensional conforme visto na Figura 23 de acordo com os requisitos levantados no estudo de caso), bases e tempo. O resultado para essa consulta é apresentado na Tabela 3. As linhas 2, 5, 7, 9, 12, 15, 18, 21, 25, 28, 31 e 34 apresentam o resultado agregado por tempo para gope e cada base; as linhas 22 e 35 apresentam o resultado agregado por tempo e por base para cada gope; por fim, a linha 36 apresenta o resultado de mais alto nível que agrega todos os níveis.

Como ficou claro nos exemplos até aqui a forma de realização de consultas para obter resultados comparativos com os conseguidos em tempo real do esquema transacional, iremos, então, a partir desse ponto, detalhar somente as execuções que envolvem contextos de análises em diferentes níveis de agregação sobre o modelo multidimensional. A partir daqui iremos considerar, também, que a partir dos exemplos anteriores o leitor já está familiarizado com a leitura das tabelas de resultados obtidas a partir dos operadores OLAP do SGBD.

```
select  dt.gope, dt.base, dti.month,
        TRAGG_CONT_ATENDIMENTO(ft.travel),
        TRAGG_AVG_ATENDIMENTO_VALUE_CLI(ft.travel)
from    fact_travel ft, dim_truck dt, dim_time dti
where   ft.truck_fk = dt.id and
        ft.begintime_fk = dti.id and
        dti.year = 2010 and dt.month >= 7 and dt.month <= 8
group by rollup(dt.gope, dt.base, dti.month);
```

Figura 28: Consulta para obter a média da quantidade de atendimentos por viagem e da quantidade de produto entregue por cliente agregados para os níveis de gope, base e mês para um dado bimestre

Para o mesmo requisito das Figuras 16 e 17 de recuperar a relação entre a quantidade entregue em uma determinada viagem e sua extensão e a relação entre a quantidade entregue e sua duração (extensão em KM e duração em horas), apresentamos a consulta da Figura 29. Nesta apresenta-se a relação nos níveis de gope e base para o primeiro semestre de um dado ano. Esse tipo de análise é importante para geração de relatórios ou cubos de navegação a serem utilizados pelos níveis gerencias da empresa em análises de desempenho e custo-benefício. A Tabela 4 apresenta o resultado para esta consulta.

```
select  dt.gope, dt.base,
        TRAGG_AVG_ATENDIMENTO_VALUE_LENGTH(ft.travel),
        TRAGG_AVG_ATENDIMENTO_VALUE_DURATION(ft.travel)
from    fact_travel ft, dim_truck dt, dim_time dti
where   ft.truck_fk = dt.id and
        ft.begintime_fk = dti.id and
        dti.year = 2011 and dt.month >= 1 and dt.month <= 6
group by rollup(dt.gope, dt.base);
```

Figura 29: Consulta para obter a média da quantidade de produto entregue pela extensão e entre a quantidade entregue e sua duração com resultado agregado para os níveis de gope e base

As informações obtidas nas consultas da Figura 18 não representam análises típicas para execução em data warehouses pois, apesar das relações serem pré-processadas durante a fase de ETL para estabelecimento do link entre a tabela fato e as dimensões, as informações são obtidas a partir de objetos que não estão armazenados no esquema e existem apenas em tempo de execução. Um exemplo que combina as relações pré-definidas das dimensões do esquema com as informações obtidas dos objetos criados em tempo de execução a partir da função de recorte de viagens é mostrado na Figura 30. Nesta obtemos, para um período de tempo especificado, a duração dos trechos de viagens que os veículos permaneceram dentro de bases agregadas por nível de base e caminhão.

Tabela 3: Resultado da consulta da Figura 28

row	GOPE	BASE	MONTH	COUNT_TR	AVG_ATD	AVG_QTD
1	1	17	8	19	4,89	2066,47
2	1	17	**TOTAL**	19	4,89	2066,47
3	1	18	7	14	2,71	2855,85
4	1	18	8	128	2,62	2034,85
5	1	18	**TOTAL**	142	2,63	2115,79
6	1	19	8	66	3,45	2301,66
7	1	19	**TOTAL**	66	3,45	2301,66
8	1	21	8	26	4,73	1661,69
9	1	21	**TOTAL**	26	4,73	1661,69
10	1	2002	7	254	7,38	3427,73
11	1	2002	8	285	7,4	3905,1
12	1	2002	**TOTAL**	539	7,39	3680,14
13	1	2004	7	35	6,02	3223,65
14	1	2004	8	126	8,13	4501,91
15	1	2004	**TOTAL**	161	7,67	4224,03
16	1	2010	7	34	9	4541,08
17	1	2010	8	89	7,76	4082,2
18	1	2010	**TOTAL**	123	8,1	4209,04
19	1	2012	7	323	7,86	4388,61
20	1	2012	8	302	9,82	5127,13
21	1	2012	**TOTAL**	625	8,81	4745,46
22	1	**TOTAL**	**TOTAL**	1701	7,37	3928,34
23	2	1002	7	329	4,98	4046,33
24	2	1002	8	334	4,64	3661,79
25	2	1002	**TOTAL**	663	4,81	3852,61
26	2	1003	7	428	5,29	4197,43
27	2	1003	8	435	5,11	4091,23
28	2	1003	**TOTAL**	863	5,2	4143,9
29	2	1007	7	49	4,18	2952,24
30	2	1007	8	57	3,89	2912,03
31	2	1007	**TOTAL**	106	4,02	2930,62
32	2	1017	7	21	7,28	6833
33	2	1017	8	226	5,57	2966
34	2	1017	**TOTAL**	247	5,72	3294,77
35	2	**TOTAL**	**TOTAL**	1879	5,06	3861,05
36	**TOTAL**	**TOTAL**	**TOTAL**	3580	6,16	3893,02

Em relação às áreas restritas e os requisitos explicitados para as consultas da Figura 19, criamos a consulta da Figura 31 para ilustrar como obter uma visão mais abrangente sobre os dados de acordo com o número de ocorrências agregadas por tempo e por caminhão.

Tabela 4: Resultado da consulta da Figura 29

row	**GOPE**	**BASE**	**COUNT_TR**	**QTD_LENGTH**	**QTD_TIME**
1	1	17	138	18,06	405,74
2	1	18	408	19,11	352,72
3	1	19	196	62,08	764,44
4	1	20	70	0	0
5	1	21	114	14,34	301,03
6	1	2002	1019	22,19	313,3
7	1	2004	656	25,88	383,79
8	1	2010	323	15,99	328,25
9	1	2012	1062	17,32	273,38
10	1	**TOTAL**	3986	19,85	320,97
11	2	25	193	35,98	610,3
12	2	27	269	19,11	350,73
13	2	1002	1148	27,78	392,77
14	2	1003	3231	17,08	318,01
15	2	1007	170	15,3	411,92
16	2	1017	541	27,55	493,01
17	2	1019	13	30,42	354,3
18	2	**TOTAL**	5565	21	372,48
19	3	28	168	24,44	344,37
20	3	1015	232	13,09	177,82
21	3	1020	3	0	0
22	3	3002	119	20,61	359,11
23	3	3003	103	2,89	42,02
24	3	**TOTAL**	625	16,71	244,55
25	4	4002	11	44,19	272,7
26	4	5002	71	15,2	170,31
27	4	**TOTAL**	82	18,05	187,25
28	5	4004	30	37,22	197,35
29	5	4005	17	0	0
30	5	4007	61	1,69	23,08
31	5	**TOTAL**	108	2,32	30,04
32	**TOTAL**	**TOTAL**	10366	20,2	339,41

```
select   dt.base, dt.truck,
         TRAGG_SUM_DURATION(travel_recort_area(ft.travel, b))
from   fact_travel ft, dim_truck dt, dim_time dti, bases b
where  ft.truck_fk = dt.id and
       ft.begintime_fk = dti.id and
       dti.year = 2011
group by rollup(dt.base, dt.truck);
```

Figura 30: Consulta para obter a soma da duração dos trechos de viagens nos quais os veículos permaneceram dentro das bases apresentados pelos níveis de base e caminhão

```
select   dti.month, dt.truck,
         tr_count_restricAreas(ft.travel)
from   fact_travel ft, dim_truck dt, dim_time dti
where  ft.truck_fk = dt.id and
       ft.begintime_fk = dti.id and
       dti.year = 2011
group by rollup(dti.month, dt.truck);
```

Figura 31: Consulta para obter a quantidade de vezes que viagens entraram em áreas restritas apresentadas por níveis de mês e caminhão

Para as análises envolvendo padrão de eventos, seja por sequência ou por número de ocorrências como na Figura 21, iremos necessitar, agora, empregar um misto entre as relações pré-definidas e relações criadas por funções cujos resultados são obtidos em tempo de execução. Exemplos são vistos na Figura 32, onde queremos a média de duração dos tempos de parada para as viagens do primeiro bimestre do ano que possuem a sequência de eventos do exemplo citado (Atendimento,Parada,Atendimento) agregadas por gopes e bases (Consulta a) e onde obtemos a velocidade média e a quantidade de eventos do tipo excesso de velocidade das viagens agregadas por tempo, e caminhões que possuem um determinado padrão de repetição de eventos (Consulta b). O resultado para esta segunda aparece na Tabela 5.

5.3.5 Exemplos de cubos de análise utilizando ferramenta OLAP

Além das análises realizadas através de consultas, como demonstrado até aqui, nossa contribuição do ambiente multidimensional permite a aplicação das técnicas e ferramentas OLAP para a criação de cubos de navegação capazes de fornecer de forma intuitiva navegação em qualquer nível de detalhe desejado de acordo com as dimensões especificadas para o mesmo. Por exemplo, pode ser do interesse dos executivos da empresa obter uma visão sobre o comprimento médio das viagens realizadas pelos veículos de entrega e, ainda, a duração média das viagens para avaliação de custos e projeções futuras para

```
(a)
select  dt.gope, dt.base, TRAGG_AVG_PARADA_DURATION(ft.travel)
from  fact_travel ft, dim_truck dt, dim_time dti
where ft.truck_fk = dt.id and
    ft.begintime_fk = dti.id and
    dti.year = 2011 and dti.month >=1 and dti.month <= 2 and
    traj_evt_seq(t,['Atendimento','Parada','Atendimento'])='TRUE'
group by rollup(gope, base);

(b)
select  dti.month, dt.truck,
        TRAGG_COUNT_EXCESSOVELOCIDADE(ft.dw_trajectory),
        TRAGG_AVG_SPEED(ft.dw_trajectory)
from  fact_travel ft, dim_truck dt, dim_time dti
where ft.truck_fk = dt.id and
    ft.begintime_fk = dti.id and
    dti.year = 2011 and dti.month >=1 and dti.month <= 3 and
    ft.dw_trajectory.get_cont_evt('Excesso de velocidade') >= 50
group by rollup(gope, base);
```

Figura 32: a) Consulta para obter a soma da duração dos eventos do tipo Parada para as trajetórias que possuem um determinado padrão de sequência de eventos apresentados por níveis de gope e base; b) consulta para obter a quantidade de eventos do tipo Excesso de velocidade e a velocidade média para as viagens que possuem um determinado padrão de repetição de eventos para os níveis de mês e caminhão

orçamentos. Um exemplo de cubo para navegação sobre essas informações é apresentado na Figura 33. Nesta podemos observar uma navegação para visualizar as informações das viagens realizadas em 2010 e mês de novembro, aquelas que ocorreram na região Sudeste detalhadas por nível de gope e base. Para melhor relação de entendimento, apresentamos, também, a quantidade de viagens.

No exemplo da Figura 34, é exibido um cubo criado para navegação sobre os dados da quantidade total de produto entregue de acordo com as dimensões de tempo de início e término, de região e de hierarquia corporativa do veículo, o que pode fornecer para os analistas uma real noção sobre a demanda de produto por período, região e filiais de distribuição da empresa em qualquer nível necessário para a tomada de decisão.

5.4 Implementação das funções de agregação

Na realização das análises exibidas nas seções anteriores deste capítulo, utilizamos algumas funções de agregação desenvolvidas de acordo com os requisitos levantados em nosso estudo de caso e o conjunto de funções de agregação para trajetórias proposto

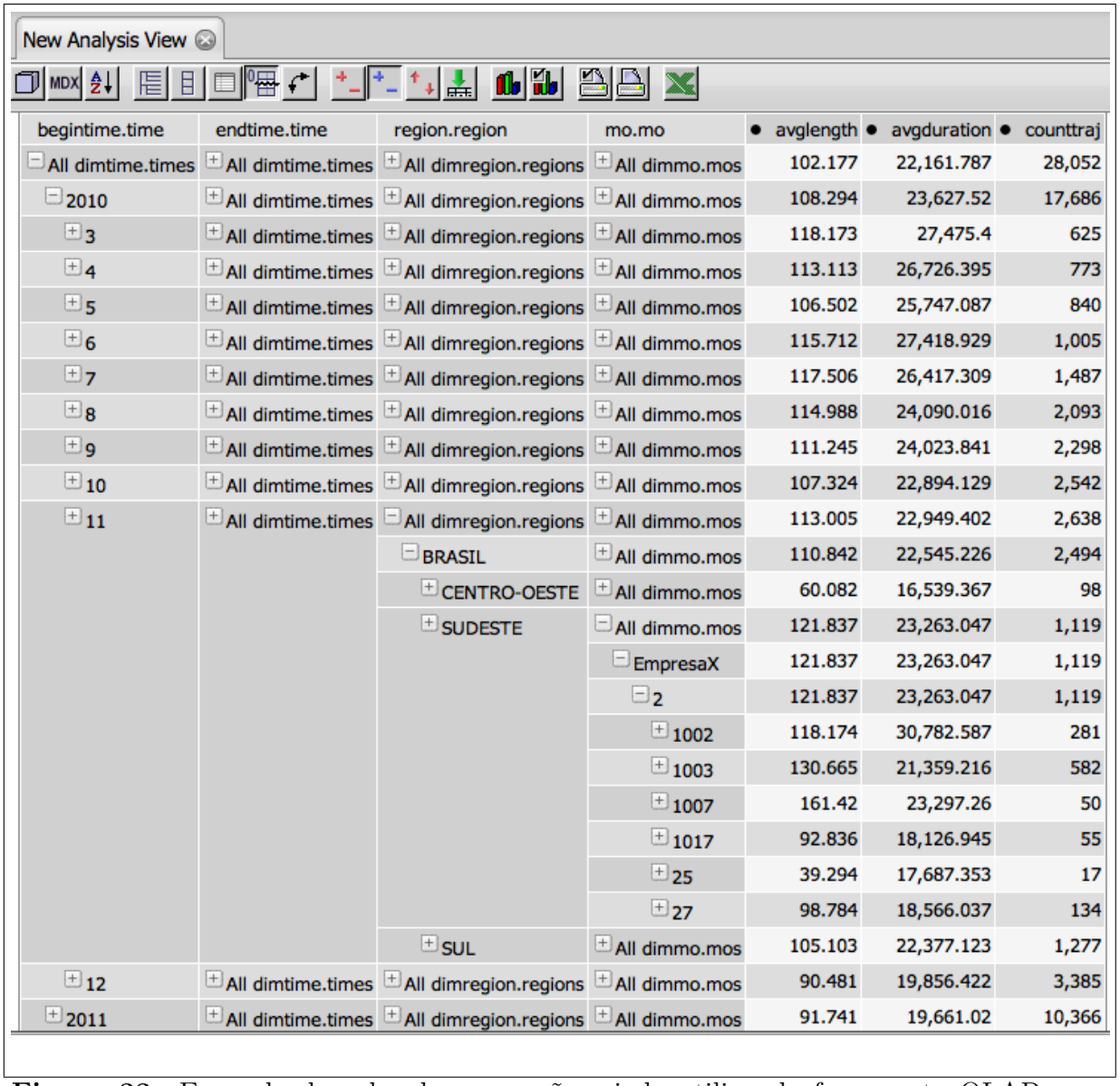
Tabela 5: Resultado para a consulta da Figura 32 *b*

row	**MONTH**	**TRUCK**	**COUNT_EVT**	**AVG_SPEED**
1	1	16	187	50,5
2	1	23	246	42
3	1	128	718	66,5
4	1	195	50	342
5	1	**TOTAL**	1201	80,66
6	2	16	53	29
7	2	128	914	54,5
8	2	196	53	365
9	2	243	110	54,5
10	2	263	56	60
11	2	264	577	54,9
12	2	**TOTAL**	1763	66,28
13	3	128	679	98,6
14	3	178	64	57
15	3	211	337	216
16	3	264	508	58,37
17	3	273	183	50,33
18	3	**TOTAL**	1771	107,13
19	**TOTAL**	**TOTAL**	4735	84,81

na seção 4.5.2. Nesta seção, demonstramos o desenvolvimento de uma das funções de agregação para trajetórias, a partir da qual, por processo similar, foram implementadas as demais.

A implementação das funções de agregação propostas exige que estas aceitem como entrada um tipo complexo não nativo do SGBD. Para isso utilizamos o framework *ODCIAggregate*[5] fornecido pelo Oracle. A criação de funções definidas pelo usuário é feita pela implementação de quatro rotinas definidas pelo framework citado em um tipo específico criado para cada função de agregação com atributos necessários para computação da mesma: *(i)* a primeira rotina é invocada para inicialização do contexto de agregação; *(ii)* a segunda realiza a iteração de fato sendo chamada repetidamente e, a cada chamada, computa e atualiza o valor do contexto; *(iii)* a terceira rotina é chamada quando não existem mais iterações a serem realizadas e computa o resultado a partir do contexto de agregação final; *(iv)* a última rotina é usada quando há necessidade de realizar o merge entre dois contextos de agregação que foram paralelizados. Por fim é criada uma função que utiliza o tipo especificado para a função de agregação e que recebe um objeto *TrajectoryType* como parâmetro. Demonstramos a implementação de uma função de agregação através do exemplo TR_AVG_QTD_CLI apresentado nas Figuras 35, 36 e

[5]link para a documentação

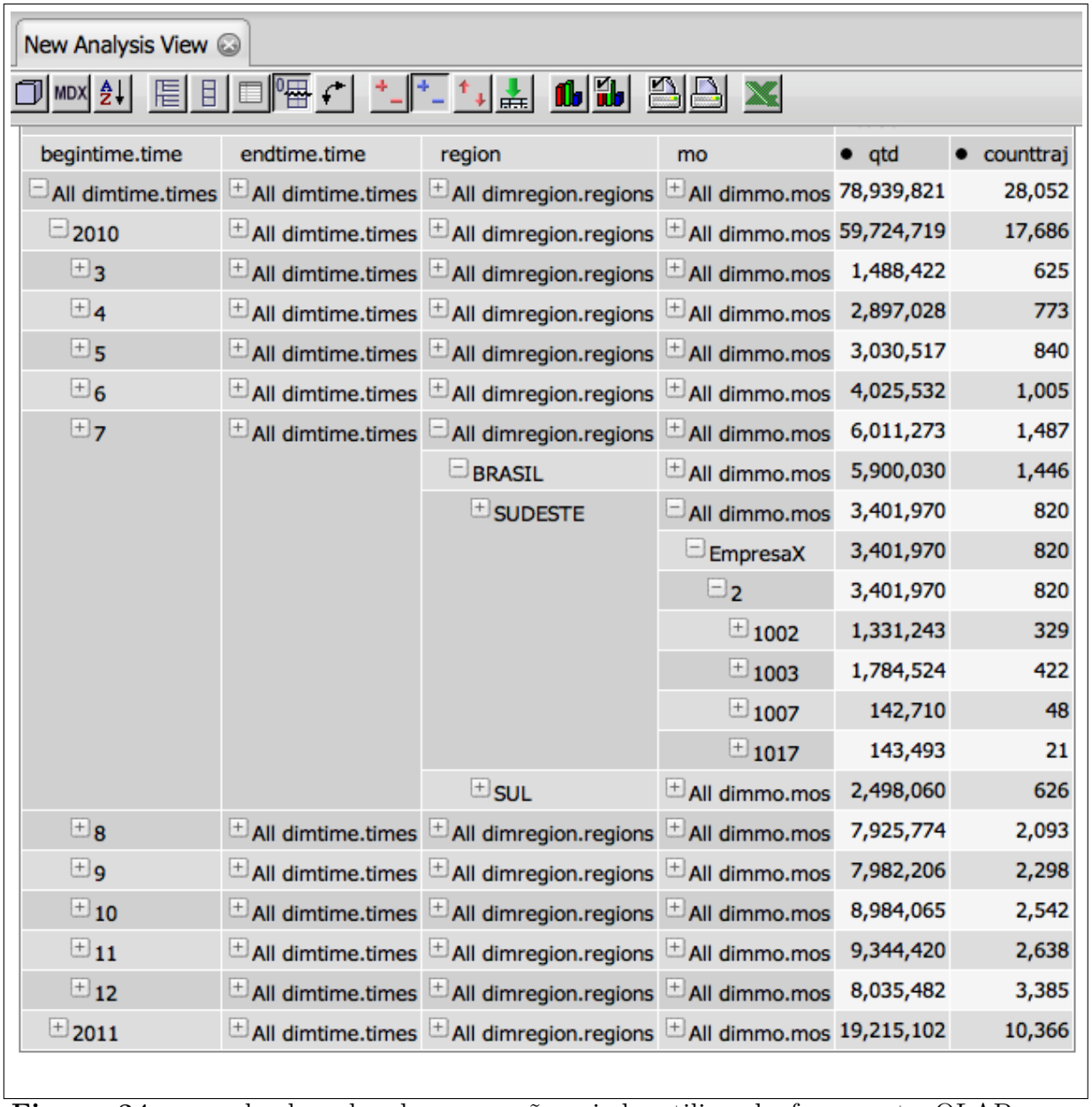


begintime.time	endtime.time	region.region	mo.mo	avglength	avgduration	counttraj
All dimtime.times	All dimtime.times	All dimregion.regions	All dimmo.mos	102.177	22,161.787	28,052
2010	All dimtime.times	All dimregion.regions	All dimmo.mos	108.294	23,627.52	17,686
3	All dimtime.times	All dimregion.regions	All dimmo.mos	118.173	27,475.4	625
4	All dimtime.times	All dimregion.regions	All dimmo.mos	113.113	26,726.395	773
5	All dimtime.times	All dimregion.regions	All dimmo.mos	106.502	25,747.087	840
6	All dimtime.times	All dimregion.regions	All dimmo.mos	115.712	27,418.929	1,005
7	All dimtime.times	All dimregion.regions	All dimmo.mos	117.506	26,417.309	1,487
8	All dimtime.times	All dimregion.regions	All dimmo.mos	114.988	24,090.016	2,093
9	All dimtime.times	All dimregion.regions	All dimmo.mos	111.245	24,023.841	2,298
10	All dimtime.times	All dimregion.regions	All dimmo.mos	107.324	22,894.129	2,542
11	All dimtime.times	All dimregion.regions	All dimmo.mos	113.005	22,949.402	2,638
		BRASIL	All dimmo.mos	110.842	22,545.226	2,494
		CENTRO-OESTE	All dimmo.mos	60.082	16,539.367	98
		SUDESTE	All dimmo.mos	121.837	23,263.047	1,119
			EmpresaX	121.837	23,263.047	1,119
			2	121.837	23,263.047	1,119
			1002	118.174	30,782.587	281
			1003	130.665	21,359.216	582
			1007	161.42	23,297.26	50
			1017	92.836	18,126.945	55
			25	39.294	17,687.353	17
			27	98.784	18,566.037	134
		SUL	All dimmo.mos	105.103	22,377.123	1,277
12	All dimtime.times	All dimregion.regions	All dimmo.mos	90.481	19,856.422	3,385
2011	All dimtime.times	All dimregion.regions	All dimmo.mos	91.741	19,661.02	10,366

Figura 33: Exemplo de cubo de navegação criado utilizando ferramenta OLAP para visualização das médias de comprimento e duração

37.

Na Figura 35 é exibido o código de criação do tipo utilizado pela função de agregação, bem como a definição dos métodos exigidos pelo framework *ODCIAggregate* para computação do resultado. Nas linhas 3 e 4 são declaradas as variáveis utilizadas na manipulação do contexto de agregação, e nas linhas 5, 6. 7 e 8 são declaradas as assinaturas dos quatro métodos do framework responsáveis pelo processamento da agregação, onde os parâmetros trajAgg, trajAgg2 e self representam os contextos de agregação passados para a iteração atual e traj o objeto trajetória do tipo *TrajectoryType* a ser manipulado no contexto atual de iteração. A Figura 36 apresenta o código para o corpo do tipo no qual são implementados os métodos de acordo com as funcionalidades descritas de cada um no parágrafo anterior, nos quais a linha 3 executa a criação do contexto de agregação e inici-



begintime.time	endtime.time	region	mo	qtd	counttraj
All dimtime.times	All dimtime.times	All dimregion.regions	All dimmo.mos	78,939,821	28,052
2010	All dimtime.times	All dimregion.regions	All dimmo.mos	59,724,719	17,686
3	All dimtime.times	All dimregion.regions	All dimmo.mos	1,488,422	625
4	All dimtime.times	All dimregion.regions	All dimmo.mos	2,897,028	773
5	All dimtime.times	All dimregion.regions	All dimmo.mos	3,030,517	840
6	All dimtime.times	All dimregion.regions	All dimmo.mos	4,025,532	1,005
7	All dimtime.times	All dimregion.regions	All dimmo.mos	6,011,273	1,487
		BRASIL	All dimmo.mos	5,900,030	1,446
		SUDESTE	All dimmo.mos	3,401,970	820
			EmpresaX	3,401,970	820
			2	3,401,970	820
			1002	1,331,243	329
			1003	1,784,524	422
			1007	142,710	48
			1017	143,493	21
		SUL	All dimmo.mos	2,498,060	626
8	All dimtime.times	All dimregion.regions	All dimmo.mos	7,925,774	2,093
9	All dimtime.times	All dimregion.regions	All dimmo.mos	7,982,206	2,298
10	All dimtime.times	All dimregion.regions	All dimmo.mos	8,984,065	2,542
11	All dimtime.times	All dimregion.regions	All dimmo.mos	9,344,420	2,638
12	All dimtime.times	All dimregion.regions	All dimmo.mos	8,035,482	3,385
2011	All dimtime.times	All dimregion.regions	All dimmo.mos	19,215,102	10,366

Figura 34: xemplo de cubo de navegação criado utilizando ferramenta OLAP para visualização da quantidade de produto entregue

alização das variáveis do mesmo; as linhas 5 e 6 computam a atualização dos valores das variáveis do contexto em cada iteração utilizando métodos do objeto trajetória; a linha 8 é responsável por computar e retornar o resultado final da agregação e as linhas 10 e 11 são executadas quando há necessidade de fazer a união de dois contextos de agregação que foram paralelizados. Por fim, na Figura 37 implementamos a função TR_AVG_QTD_CLI que recebe um objeto trajetória (i.e. *TrajectoryType*) e realiza a agregação utilizando o tipo descrito aqui.

Perceba que, na implementação desta função de agregação para trajetórias, executamos todas as manipulações em alto nível utilizando como entrada da função de agregação um objeto trajetória e a computação é realizada com auxílio dos métodos próprios do tipo

```
1 create or replace
  type TrAggImpl_AVG_QTD_CLI as object (

3 contEvt NUMBER,
4 sumEvtValue NUMBER,

5 static function ODCIAggregateInitialize
   (trajAgg IN OUT TrAggImpl_AVG_QTD_CLI)
   return number,

6 member function ODCIAggregateIterate
   (self IN OUT TrAggImpl_AVG_QTD_CLI, traj IN trajectoryType)
   return number,

7 member function ODCIAggregateTerminate
   (self IN TrAggImpl_AVG_QTD_CLI,
   returnValue OUT number, flags IN number) return number,

8 member function ODCIAggregateMerge
   (self IN OUT TrAggImpl_AVG_QTD_CLI,
    trajAgg2 IN TrAggImpl_AVG_QTD_CLI) return number
  );
```

Figura 35: Criação do tipo para a função de agregação TR_AVG_QTD_CLI

TrajectoryType, o que, mais uma vez, ressalta a importância de termos uma trajetória como um objeto de primeira classe.

```
1 create or replace
  type body TrAggImpl_AVG_QTD_CLI is

2 static function ODCIAggregateInitialize
   (trajavg IN OUT TrAggImpl_AVG_QTD_CLI)
   return number is
  begin
3   trajAgg := TrAggImpl_AVG_QTD_CLI(0,0);
    return ODCIConst.Success;
  end;

4 member function ODCIAggregateIterate
   (self IN OUT TrAggImpl_AVG_QTD_CLI, traj IN trajectoryType)
   return number is
  begin
5   self.sumEvtValue :=
      self.sumEvtValue + traj.getSumEvtValue1('Atendimento');
6   self.contEvt :=
      self.contEvt + traj.getContEvt('Atendimento');
    return ODCIConst.Success;
  end;

7 member function ODCIAggregateTerminate
   (self IN TrAggImpl_AVG_QTD_CLI,
    returnValue OUT number, flags IN number) return number is
  begin
8   if self.contEvt = 0 then
      returnValue := 0;
    else
      returnValue := self.sumEvtValue / self.contEvt;
    end if;
    return ODCIConst.Success;
  end;

9 member function ODCIAggregateMerge
   (self IN OUT TrAggImpl_AVG_QTD_CLI,
    trajAgg2 IN TrAggImpl_AVG_QTD_CLI) return number is
  begin
10  self.sumEvtValue := self.sumEvtValue + trajAgg2.sumEvtValue;
11  self.contEvt := self.contEvt + trajAgg2.contEvt;
    return ODCIConst.Success;
  end;

  end;
```

Figura 36: Criação do corpo do tipo para a função de agregação TR_AVG_QTD_CLI

```
create or replace
FUNCTION TR_AVG_QTD_CLI (input trajectory_ty)
  RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TrAggImpl_AVG_QTD_CLI;
```

Figura 37: Criação da função de agregação TR_AVG_QTD_CLI

6 CONCLUSÃO

Vários estudos têm sido realizados na comunidade de banco de dados para lidar com dados espaço-temporais de objetos móveis. Entretanto negligenciando o suporte ao conceito de trajetórias como entidades de primeira ordem com capacidade de representar a semântica do ponto de vista da aplicação. Há grande interesse na realização de análise sobre dados de trajetórias de objetos móveis considerando dois cenários de aplicabilidade: análises de tempo real e análises históricas, que podem ser usadas para tomada de decisões estratégicas.

Apresentamos, então, neste trabalho, um conjunto de requisitos para modelagem e análise de trajetórias de forma a permitir: (i) representar e consultar trajetórias de objetos móveis como objetos de primeira classe; (ii) adicionar semântica do domínio da aplicação às trajetórias e, desse modo, (iii) permitir a análises de tempo real e históricas. Para satisfazer esses requisitos nós propomos dois meta-esquemas, um para trajetórias semânticas em ambiente transacional objeto-relacional de SGBD e outro para trajetórias semânticas em banco de dados multidimensional. Para estes, propomos, também, um primeiro algoritmo para ajudar a construção de objetos trajetória a partir dos dados brutos em ambiente transacional, bem como um segundo algoritmo para povoar o esquema multidimensional a partir do ambiente transacional.

As aplicações de gerenciamento de trajetórias exigem a utilização de ferramentas de apoio à decisão estratégica em ambos os ambientes para realização de análises de tempo real e históricas. A realização de análises exige a sumarização de informações através de operações de agregação. Com isso, funções de agregação específicas para trajetórias de objetos móveis devem ser desenvolvidas. Por isso propomos, também, uma assinatura para guiar a especificação e construção de funções de agregação para trajetórias, bem como um conjunto de funções de agregação definidas de acordo com o estudo de caso.

Este trabalho contribui com uma solução integrada para os requisitos necessários para gerenciamento e análises de trajetórias de objetos móveis desde a modelagem conceitual até a representação em ambiente transacional e multidimensional para realização de análises com suporte de funções de agregação específicas. E, até onde vai nosso conhecimento, esse é o primeiro trabalho na literatura de gerenciamento de dados de trajetória

que lida com todos esses requisitos de forma integrada.

Como trabalhos futuros, nós estamos interessados em realizar um levantamento de requisitos de análises sobre dados de trajetórias mais detalhado e na criação de uma aplicação completa e otimizada para longos períodos de testes executados em um ambiente corporativo real. Além disso, utilizar outros cenários reais (e.g. tráfego de cidades, migração de animais, deslocamento de pessoas a pé) para levantamento de requisitos e avaliação de nossas contribuições em outros domínios de aplicações. Outro aspecto que requer investigações mais detalhadas diz respeito ao estudo e desenvolvimento de funções de agregação para trajetórias como elementos complexos de primeira grandeza, o qual necessita de estudos mais formais, principalmente em relação aos aspectos semânticos (i.e. events) e a outros objetos da aplicação (e.g. regiões de interesse). Na área mais prática, é preciso estender os mecanismos de ferramentas OLAP para que estas consigam trabalhar e manipular as trajetórias como objetos complexos e utilizar as funções de agregação próprias criadas pelos projetistas de aplicações.

REFERÊNCIAS

[1] Alvares, L. O., Bogorny, V., de Macêdo, J. A. F., Moelans, B., and Spaccapietra, S. Dynamic modeling of trajectory patterns using data mining and reverse engineering. In *Tutorials, posters, panels and industrial contributions at the 26th International Conference on Conceptual Modeling - ER* (Auckland, New Zealand, 2007), A. H. F. L. L. M. John Grundy, Sven Hartmann and J. F. Roddick, Eds., vol. 83 of *CRPIT*, ACS, pp. 149–154.

[2] Alvares, L. O., Bogorny, V., Kuijpers, B., de Macêdo, J. A. F., Moelans, B., and Vaisman, A. A. A model for enriching trajectories with semantic geographical information. In *Proceedings of the 15th ACM International Symposium on Geographic Information Systems* (New York, NY, USA, 2007), GIS '07, ACM, pp. 22:1–22:8.

[3] Andrienko, N. V., Andrienko, G. L., Pelekis, N., and Spaccapietra, S. Basic concepts of movement data. In *Mobility, Data Mining and Privacy*. 2008, pp. 15–38.

[4] Baltzer, O., Dehne, F., Hambrusch, S., and Rau-Chaplin, A. Olap for trajectories. In *Proceedings of the 19th international conference on Database and Expert Systems Applications* (Berlin, Heidelberg, 2008), DEXA '08, Springer-Verlag, pp. 340–347.

[5] Braz, F., Orlando, S., Orsini, R., Raffaela, A., Roncato, A., and Silvestri, C. Approximate aggregations in trajectory data warehouses. In *Data Engineering Workshop, 2007 IEEE 23rd International Conference on* (17-20 2007), pp. 536 –545.

[6] de Almeida, V. T., Guting, R. H., and Behr, T. Querying moving objects in secondo. In *Proceedings of the 7th International Conference on Mobile Data Management* (Washington, DC, USA, 2006), MDM '06, IEEE Computer Society, pp. 47–.

[7] de Macêdo, J. A. F., Vangenot, C., Othman, W., Pelekis, N., Frentzos, E., Kuijpers, B., Ntoutsi, I., Spaccapietra, S., and Theodoridis, Y. Trajectory data models. In *Mobility, Data Mining and Privacy*. 2008, pp. 123–150.

[8] Dieker, S., and Güting, R. H. Plug and play with query algebras: Secondo-a generic dbms development environment. In *IDEAS* (2000), pp. 380–392.

[9] Forlizzi, L., Güting, R. H., Nardelli, E., and Schneider, M. A data model and data structures for moving objects databases. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data* (New York, NY, USA, 2000), SIGMOD '00, ACM, pp. 319–330.

[10] Frentzos, E., Pelekis, N., Ntoutsi, I., and Theodoridis, Y. Trajectory database systems. In *Mobility, Data Mining and Privacy*. 2008, pp. 151–187.

[11] Giannotti, F., Nanni, M., Pinelli, F., and Pedreschi, D. Trajectory pattern mining. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* (New York, NY, USA, 2007), ACM, pp. 330–339.

[12] Guc, B., May, M., Saygin, Y., and Korner, C. Semantic annotation of gps trajectories. *11th AGILE International Conference on Geographic Information Science (AGILE), 2008.* (2008).

[13] Guting, R. H., Behr, T., Almeida, V., Ding, Z., Hoffmann, F., and Spiekermann, M. Secondo: An extensible dbms architecture and prototype. Tech. rep., Fernuniversitat Hagen, 2004.

[14] Güting, R. H., Behr, T., and Düntgen, C. Secondo: A platform for moving objects database research and for publishing and integrating research implementations. *IEEE Data Eng. Bull. 33*, 2 (2010), 56–63.

[15] Güting, R. H., Böhlen, M. H., Erwig, M., Jensen, C. S., Lorentzos, N. A., Schneider, M., and Vazirgiannis, M. A foundation for representing and querying moving objects. *ACM Trans. Database Syst. 25*, 1 (2000), 1–42.

[16] Güting, R. H., and Schneider, M. *Moving Objects Databases*. Morgan Kaufmann, 2005.

[17] Gómez, L., Kuijpers, B., Moelans, B., and Vaisman, A. A state-ofa-the-art in spatio-temporal data warehousing, olap and mining. In *Integrations of Data Warehousing, Data Mining and Database Technologies: Innovative Approaches*, D. Taniar and L. Chen, Eds. IGI Global, 2011, ch. 9, pp. 200–236.

[18] Jeung, H., Yiu, M. L., Zhou, X., Jensen, C. S., and Shen, H. T. Discovery of convoys in trajectory databases. *Proc. VLDB Endow. 1*, 1 (2008), 1068–1080.

[19] Kuijpers, B., Moelans, B., and Van de Weghe, N. Qualitative polyline similarity testing with applications to query-by-sketch, indexing and classification. In *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems* (New York, NY, USA, 2006), GIS '06, ACM, pp. 11–18.

[20] Laube, P., Imfeld, S., and Weibel, R. Discovering relative motion patterns in groups of moving point objects. *International Journal of Geographical Information Science 19*, 6 (2005), 639–668.

[21] Lee, J.-G., Han, J., and Whang, K.-Y. Trajectory clustering: a partition-and-group framework. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data* (New York, NY, USA, 2007), ACM, pp. 593–604.

[22] Lema, J. A. C., Forlizzi, L., Güting, R. H., Nardelli, E., and Schneider, M. Algorithms for moving objects databases. *Comput. J. 46*, 6 (2003), 680–712.

[23] Leonardi, L., Marketos, G., Frentzos, E., Giatrakos, N., Orlando, S., Pelekis, N., Raffaeta, A., Roncato, A., Silvestri, C., and Theodoridis, Y. T-warehouse: Visual olap analysis on trajectory data. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on* (mar. 2010), pp. 1141 –1144.

[24] Leonardi, L., Orlando, S., Raffaetà, A., Roncato, A., and Silvestri, C. Frequent spatio-temporal patterns in trajectory data warehouses. In *SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing* (New York, NY, USA, 2009), ACM, pp. 1433–1440.

[25] Malinowski, E., and Zimányi, E. Representing spatiality in a conceptual multidimensional model. In *GIS '04: Proceedings of the 12th annual ACM international workshop on Geographic information systems* (New York, NY, USA, 2004), ACM, pp. 12–22.

[26] Marketos, G., Frentzos, E., Ntoutsi, I., Pelekis, N., Raffaetà, A., and Theodoridis, Y. Building real-world trajectory warehouses. In *MobiDE '08: Proceedings of the Seventh ACM International Workshop on Data Engineering for Wireless and Mobile Access* (New York, NY, USA, 2008), ACM, pp. 8–15.

[27] Moreno, B., Times, V. C., Renso, C., and Bogorny, V. Looking inside the stops of trajectories of moving objects. In *GeoInfo* (2010), V. Bogorny and L. Vinhas, Eds., MCT/INPE, pp. 9–20.

[28] Nanni, M., and Pedreschi, D. Time-focused clustering of trajectories of moving objects. *J. Intell. Inf. Syst. 27* (November 2006), 267–289.

[29] Orlando, S., Orsini, R., Raffaeta, A., and Roncato, A. Trajectory data warehouses: Design and implementation issues. *Journal of Computing Science and Engineering 1*, 2 (December 2007), 211–232.

[30] Orlando, S., Orsini, R., Raffaetà, A., Roncato, A., and Silvestri, C. Spatio-temporal aggregations in trajectory data warehouses. In *DaWaK* (2007), pp. 66–77.

[31] Oueslati, W., and Akaichi, J. Mobile information collectors' trajectory data warehouse design. *International Journal of Managing Information Technology 2* (2010), 1–20.

[32] Palma, A. T., Bogorny, V., Kuijpers, B., and Alvares, L. O. A clustering-based approach for discovering interesting places in trajectories. In *Proceedings of the 2008 ACM symposium on Applied computing* (New York, NY, USA, 2008), SAC '08, ACM, pp. 863–868.

[33] Pelekis, N. *STAU: A spatio-temporal extension for the ORACLE DBMS*. PhD thesis, UMIST, 2002.

[34] Pelekis, N., Kopanakis, I., Ntoutsi, I., Marketos, G., and Theodoridis, Y. Mining trajectory databases via a suite of distance operators. In *Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering Workshop* (Washington, DC, USA, 2007), IEEE Computer Society, pp. 575–584.

[35] Pelekis, N., Raffaeta, A., Damiani, M. L., Vangenot, C., Marketos, G., Frentzos, E., Ntoutsi, I., and Theodoridis, Y. Towards trajectory data warehouses. In *Mobility, Data Mining and Privacy*. Springer, 2008, pp. 189–211.

[36] Pelekis, N., and Theodoridis, Y. Boosting location-based services with a moving object database engine. In *Proceedings of the 5th ACM international workshop on Data engineering for wireless and mobile access* (New York, NY, USA, 2006), MobiDE '06, ACM, pp. 3–10.

[37] Pelekis, N., Theodoridis, Y., Vosinakis, S., and Panayiotopoulos, T. Hermes - a framework for location-based data management. In *EDBT* (2006), Y. E. Ioannidis, M. H. Scholl, J. W. Schmidt, F. Matthes, M. Hatzopoulos, K. Böhm, A. Kemper, T. Grust, and C. Böhm, Eds., vol. 3896 of *Lecture Notes in Computer Science*, Springer, pp. 1130–1134.

[38] Pelekis, N., Theodoulidis, B., Theodoridis, Y., and Kopanakis, I. An oracle data cartridge for moving objects. Tech. rep., Laboratory of Information Systems, University of Piraeus, Piraeus, Hellas, March 2005.

[39] Porto, F., Moura, A. M. C., Palazzi, D., da Silva, F. C., de Castro, L. E. V., and Cameron, L. C. Towards a scientific database for olympic athletes. In *Challenges in eScience Workshop* (Hotel Vila Real, Itaipava, Petropolis, Rio de Janeiro, October 2010).

[40] Rocha, J., Oliveira, G., Alvares, L., Bogorny, V., and Times, V. Db-smot: A direction-based spatio-temporal clustering method. In *Intelligent Systems (IS), 2010 5th IEEE International Conference* (july 2010), pp. 114 –119.

[41] Sistla, A. P., Wolfson, O., Chamberlain, S., and Dao, S. Modeling and querying moving objects. In *Proceedings of the Thirteenth International Conference on Data Engineering* (Washington, DC, USA, 1997), ICDE '97, IEEE Computer Society, pp. 422–432.

[42] Spaccapietra, S., Parent, C., Damiani, M. L., de Macedo, J. A., Porto, F., and Vangenot, C. A conceptual view on trajectories. Tech. rep., EPFL – Ecole Polytechnique Federale de Lausanne, 2007.

[43] Spaccapietra, S., Parent, C., Damiani, M. L., de Macedo, J. A., Porto, F., and Vangenot, C. A conceptual view on trajectories. *Data & Knowledge Engineering 65*, 1 (2008), 126 – 146. Including Special Section: Privacy Aspects of Data Mining Workshop (2006) - Five invited and extended papers.

[44] Vega Lopez, I. F., Snodgrass, R. T., and Moon, B. Spatiotemporal aggregate computation: A survey. *IEEE Trans. on Knowl. and Data Eng. 17* (February 2005), 271–286.

[45] Wolfson, O., Xu, B., Chamberlain, S., and Jiang, L. Moving objects databases: issues and solutions. In *Scientific and Statistical Database Management, 1998. Proceedings. Tenth International Conference on* (jul 1998), pp. 111 –122.

[46] Yan, Z., Chakraborty, D., Parent, C., Spaccapietra, S., and Aberer, K. Semitri: a framework for semantic annotation of heterogeneous trajectories. In *Proceedings of the 14th International Conference on Extending Database Technology - EDBT* (New York, NY, USA, 2011), EDBT/ICDT '11, ACM, pp. 259–270.

APÊNDICE A – CÓDIGOS DE IMPLEMENTAÇÃO DO AMBIENTE TRANSACIONAL

Neste apêndice apresentamos os scripts e códigos de implementação do ambiente transacional utilizado nos experimentos.

A.1 Tipos

Listing A.1: implementação dos tipos

```
CREATE OR REPLACE TYPE event_ty
;
/

CREATE OR REPLACE TYPE moving_object_ty
;
/

CREATE OR REPLACE TYPE position_ty
;
/

CREATE OR REPLACE TYPE trajectory_ty
;
/

CREATE OR REPLACE TYPE events_list_ty
    IS TABLE OF REF event_ty
;
/

CREATE OR REPLACE TYPE positions_list_ty
    IS TABLE OF REF position_ty
;
/

CREATE OR REPLACE TYPE trajectories_list_ty
    IS TABLE OF REF trajectory_ty
;
/
```

```
create or replace
TYPE position_ty
    AS OBJECT
    (
        p_id NUMERIC ,
        tr_id REF trajectory_ty ,
        mo_id REF moving_object_ty ,
        spatial_point SDO_GEOMETRY ,
        instant Timestamp(6) ,
        speed NUMERIC
    ) NOT FINAL
;
/

create or replace
TYPE event_ty
    AS OBJECT
    (
        e_id NUMERIC ,
        mo_id REF moving_object_ty ,
        tr_id REF trajectory_ty ,
        type VARCHAR(100) ,
        beginning position_ty ,
        ending position_ty ,
        value1 number ,
        value2 number ,
        duration number ,

        member procedure set_duration
    ) NOT FINAL
;
/

create or replace
TYPE BODY EVENT_TY AS

  member procedure set_duration AS
    resultado number := 0;
  BEGIN
    resultado := get_times_diff(self.beginning.instant, self.ending.instant);

    if (resultado is not null) then
      self.duration := resultado;
    else
      self.duration := 0;
    end if;
  END set_duration;

END;
/

create or replace
TYPE moving_object_ty
    AS OBJECT
    (
        mo_id NUMERIC ,
        mo_codigo NUMERIC ,
        trajectories_lst trajectories_list_ty ,
```

```
        member function get_cont_traj return number
    ) NOT FINAL
;
/

create or replace
TYPE trajectory_ty
    AS OBJECT
    (
        tr_id NUMERIC ,
        mo_id REF moving_object_ty ,
        beginning REF position_ty ,
        ending REF position_ty ,
        speed NUMERIC ,
        accelerate NUMERIC ,
        direction VARCHAR(25) ,
        duration NUMERIC ,
        length NUMERIC ,
        trline SDO_GEOMETRY ,
        positions_lst positions_list_ty ,
        events_lst events_list_ty ,

        MEMBER procedure set_avgspeed ,
        MEMBER procedure set_ending ,
        MEMBER procedure set_direction ,
        MEMBER procedure set_avgaccelerate ,
        MEMBER procedure set_beginning ,
        MEMBER procedure set_duration ,
        MEMBER procedure set_trLength ,
        MEMBER procedure set_line ,
        member function get_cont_evt(evtType in varchar2) return number ,
        member function get_sum_dur_evt(evtType in varchar2) return number ,
        member function get_sum_value1_evt(evtType in varchar2) return number ,
        member function get_sum_value2_evt(evtType in varchar2) return number
    ) NOT FINAL
;
/

create or replace
TYPE BODY trajectory_ty AS

    MEMBER PROCEDURE set_avgspeed AS
      speed number;
    BEGIN
      select PT_AVG_SPEED(deref(p.column_value)) into speed
      from table(self.positions_lst) p;

      self.speed := speed;
    END set_avgspeed;

    MEMBER PROCEDURE set_ending AS
    BEGIN
      select fim into self.ending
      from (
        select p.column_value as fim, row_number()
          over (order by p.column_value.instant desc) i
        from table(self.positions_lst) p
      )
      where i = 1;
```

```
END set_ending;

MEMBER PROCEDURE set_direction AS
begin
  null;
end;

MEMBER PROCEDURE set_avgaccelerate AS
begin
  null;
end;

member procedure set_beginning AS
BEGIN
  select inicio into self.beginning
  from (
    select p.column_value as inicio, row_number()
      over (order by p.column_value.instant asc) i
    from table(self.positions_lst) p
  )
  where i = 1;
END set_beginning;

MEMBER PROCEDURE set_duration AS
  pos1 position_ty;
  pos2 position_ty;
begin
  select deref(self.beginning), deref(self.ending) into pos1, pos2
  from dual;

  self.duration := get_times_diff(pos1.instant, pos2.instant);
end;

MEMBER PROCEDURE set_trLength AS
begin
  self.length := SDO_GEOM.SDO_LENGTH(self.trline, 0.005, 'unit=KM');
end;

MEMBER PROCEDURE set_Line AS
  cursor cursor_p is
    select  p.column_value.spatial_point as ponto,
            p.column_value.spatial_point.sdo_point.x as pontox,
            p.column_value.spatial_point.sdo_point.y as pontoy
    from table(self.positions_lst) p;
  indx cursor_p%rowtype;
  i number;
  ord MDSYS.SDO_ORDINATE_ARRAY;
  point_aux SDO_GEOMETRY;
  point_aux2 SDO_GEOMETRY;
  flaginicio number;
begin
  ord := MDSYS.SDO_ORDINATE_ARRAY();

  i := 1;
  flaginicio := 0;
  point_aux := null;
  point_aux2 := null;
  for indx in cursor_p LOOP
    if (flaginicio = 0) then
```

```
      point_aux := indx.ponto;
      ord.extend;
      ord(i) := indx.pontox;
      i := i + 1;
      ord.extend;
      ord(i) := indx.pontoy;
      i := i + 1;
      flaginicio := 9;
    else
     if ( SDO_GEOM.SDO_DISTANCE(indx.ponto, point_aux, 0.005, 'unit=KM') <= 4 ) then
      point_aux := indx.ponto;
      ord.extend;
      ord(i) := indx.pontox;
      i := i + 1;
      ord.extend;
      ord(i) := indx.pontoy;
      i := i + 1;
     end if;
    end if;
  end loop;

  self.trline := SDO_GEOMETRY(2002, 8307, null, SDO_ELEM_INFO_ARRAY(1,2,1), ord);
end;

MEMBER function get_cont_evt(evtType in varchar2) return number is
  resultado number := 0;
begin
  select count(e.column_value) into resultado
  from table(self.events_lst) e
  where e.column_value.type = evtType;

  if (resultado is null) then
    return 0;
  else
    return resultado;
  end if;
end;

MEMBER function get_sum_dur_evt(evtType in varchar2) return number is
  resultado number := 0;
begin
  select sum(deref(e.column_value).duration) into resultado
  from table(self.events_lst) e
  where e.column_value.type = evtType;

  if (resultado is null) then
    return 0;
  else
    return resultado;
  end if;
end;

MEMBER function get_sum_value1_evt(evtType in varchar2) return number is
  resultado number := 0;
begin
  select sum(e.column_value.value1) into resultado
  from table(self.events_lst) e
  where e.column_value.type = evtType;
```

```
      if (resultado is null) then
        return 0;
      else
        return resultado;
      end if;
    end;

    MEMBER function get_sum_value2_evt(evtType in varchar2) return number is
    begin
      null;
    end;

END;
/

create or replace
TYPE AREA_TY AS OBJECT (
  a_id numeric,
  abbreviation varchar2(8),
  name varchar2(100),
  type varchar2(100),
  region sdo_geometry
);
/
```

A.2 Tabelas

Listing A.2: implementação das tabelas

```
CREATE TABLE events
    OF event_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    e_id  NOT NULL ,
    beginning  NOT NULL ,
    ending NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;

ALTER TABLE EVENTS
ADD CONSTRAINT EVENTS_PK PRIMARY KEY
(
  E_ID
)
ENABLE;

CREATE TABLE moving_objects
    OF moving_object_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    mo_id  NOT NULL ,
    mo_codigo  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
    nested table trajectories_lst store as trajectories_nt
;
```

```
ALTER TABLE moving_objects
ADD CONSTRAINT moving_objects_PK PRIMARY KEY
(
  MO_ID
)
ENABLE;


CREATE TABLE positions
    OF position_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    p_id  NOT NULL ,
    spatial_point  NOT NULL ,
    instant  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;

ALTER TABLE positions
ADD CONSTRAINT positions_PK PRIMARY KEY
(
  P_ID
)
ENABLE;

CREATE TABLE trajectories
    OF trajectory_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    tr_id  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
    nested table positions_lst store as positions_nt
    nested table events_lst store as events_nt
;

ALTER TABLE trajectories
ADD CONSTRAINT trajectories_PK PRIMARY KEY
(
  TR_ID
)
ENABLE;

CREATE TABLE areas
    OF area_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    a_id  NOT NULL ,
    abbreviation  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;

ALTER TABLE areas
ADD CONSTRAINT areas_PK PRIMARY KEY
(
  A_ID
)
```

```
ENABLE;

CREATE SEQUENCE  OR_POSITIONS_PK  MINVALUE 1 MAXVALUE 9999999999999999999999999999
  INCREMENT BY 1 START WITH 1 NOCACHE  NOORDER  NOCYCLE ;
/

CREATE SEQUENCE OR_EVENTS_PK MINVALUE 1 MAXVALUE 9999999999999999999999999999
  INCREMENT BY 1 START WITH 1 NOCACHE NOORDER NOCYCLE ;
/

CREATE SEQUENCE  OR_TRAJECTORIES_PK  MINVALUE 1 MAXVALUE 9999999999999999999999999999
  INCREMENT BY 1 START WITH 1 NOCACHE  NOORDER  NOCYCLE ;
/

CREATE SEQUENCE  OR_MOVING_OBJECTS_PK  MINVALUE 1 MAXVALUE 9999999999999999999999999999
  INCREMENT BY 1 START WITH 1 NOCACHE  NOORDER  NOCYCLE ;
/

CREATE SEQUENCE  OR_AREAS_PK  MINVALUE 1 MAXVALUE 9999999999999999999999999999
  INCREMENT BY 1 START WITH 1 NOCACHE  NOORDER  NOCYCLE ;
/

CREATE OR REPLACE
TRIGGER OR_POSITIONS_PK_SEQ
BEFORE INSERT ON POSITIONS
FOR EACH ROW
BEGIN
  select OR_POSITIONS_PK.nextval
  into :new.p_id
  from dual;
END;
/

create or replace
TRIGGER OR_EVENTS_PK_SEQ
BEFORE INSERT ON EVENTS
FOR EACH ROW
BEGIN
  select OR_EVENTS_PK.nextval
  into :new.e_id
  from dual;

  select OR_EVENTS_PK.nextval
  into :new.beginning.p_id
  from dual;

   select OR_EVENTS_PK.nextval
  into :new.ending.p_id
  from dual;
END;
/

create or replace
TRIGGER OR_TRAJECTORIES_PK_SEQ
BEFORE INSERT ON TRAJECTORIES
FOR EACH ROW
BEGIN
  select OR_TRAJECTORIES_PK.nextval
  into :new.tr_id
```

```
  from dual;
END;
/

create or replace
TRIGGER OR_MO_PK_SEQ
BEFORE INSERT ON MOVING_OBJECTS
FOR EACH ROW
BEGIN
  select OR_MOVING_OBJECTS_PK.nextval
  into :new.mo_id
  from dual;
END;
/

create or replace
TRIGGER OR_AREA_PK_SEQ
BEFORE INSERT ON AREAS
FOR EACH ROW
BEGIN
  select OR_AREAS_PK.nextval
  into :new.a_id
  from dual;
END;
/

CREATE INDEX indx_type ON events
    (
     type ASC
    )
;

CREATE INDEX indx_starttime ON EVENTS
  (
    beginning.instant ASC
  )
;

CREATE INDEX indx_endtime ON EVENTS
  (
    ending.instant ASC
  )
;


CREATE INDEX indx_mo_codigo ON moving_objects
    (
     mo_codigo ASC
    )
;


CREATE INDEX indx_instant ON positions
    (
     instant ASC
    )
;

CREATE INDEX indx_speed ON positions
```

```
    (
     speed ASC
    )
;

CREATE INDEX indx_tr_speed ON trajectories
    (
     speed ASC
    )
;

CREATE INDEX indx_tr_length ON trajectories
    (
     length ASC
    )
;

CREATE INDEX indx_tr_duration ON trajectories
    (
     duration ASC
    )
;

CREATE INDEX indx_a_type ON areas
    (
     type ASC
    )
;

CREATE INDEX indx_a_name ON areas
    (
     name ASC
    )
;

INSERT INTO USER_SDO_GEOM_METADATA
(TABLE_NAME, COLUMN_NAME, DIMINFO, SRID)
VALUES (
'POSITIONS',
'SPATIAL_POINT',
SDO_DIM_ARRAY (
SDO_DIM_ELEMENT('X', -180, 180, 0.005),
SDO_DIM_ELEMENT('Y', -90, 90, 0.005)),
8307);
/

CREATE INDEX POSITIONS_INDEX ON POSITIONS (SPATIAL_POINT)
  INDEXTYPE IS MDSYS.SPATIAL_INDEX;
/

INSERT INTO USER_SDO_GEOM_METADATA
(TABLE_NAME, COLUMN_NAME, DIMINFO, SRID)
VALUES (
'TRAJECTORIES',
'TRLINE',
SDO_DIM_ARRAY (
SDO_DIM_ELEMENT('X', -180, 180, 0.005),
SDO_DIM_ELEMENT('Y', -90, 90, 0.005)),
8307);
```

```
/

CREATE INDEX TRAJ_INDEX_TRLINE ON TRAJECTORIES (TRLINE)
  INDEXTYPE IS MDSYS.SPATIAL_INDEX;
/

INSERT INTO USER_SDO_GEOM_METADATA
(TABLE_NAME, COLUMN_NAME, DIMINFO, SRID)
VALUES (
'EVENTS',
'BEGINNING.SPATIAL_POINT',
SDO_DIM_ARRAY (
SDO_DIM_ELEMENT('X', -180, 180, 0.005),
SDO_DIM_ELEMENT('Y', -90, 90, 0.005)),
8307);
/

CREATE INDEX EVENTS_INDEX_BEGIN ON EVENTS (BEGINNING.SPATIAL_POINT)
  INDEXTYPE IS MDSYS.SPATIAL_INDEX;
/

INSERT INTO USER_SDO_GEOM_METADATA
(TABLE_NAME, COLUMN_NAME, DIMINFO, SRID)
VALUES (
'EVENTS',
'ENDING.SPATIAL_POINT',
SDO_DIM_ARRAY (
SDO_DIM_ELEMENT('X', -180, 180, 005),
SDO_DIM_ELEMENT('Y', -90, 90, 005)),
8307);
/

CREATE INDEX EVENTS_INDEX_END ON EVENTS (ENDING.SPATIAL_POINT)
  INDEXTYPE IS MDSYS.SPATIAL_INDEX;
/

INSERT INTO USER_SDO_GEOM_METADATA
(TABLE_NAME, COLUMN_NAME, DIMINFO, SRID)
VALUES (
'AREAS',
'REGION',
SDO_DIM_ARRAY (
SDO_DIM_ELEMENT('X', -180, 180, 0.005),
SDO_DIM_ELEMENT('Y', -90, 90, 0.005)),
8307);
/

CREATE INDEX AREAS_INDEX ON AREAS (REGION)
  INDEXTYPE IS MDSYS.SPATIAL_INDEX;
/
```

A.3 Procedimentos e Funções

Listing A.3: implementação dos procedimentos e funções

```
create or replace
FUNCTION GET_CONVERTED_POINT (longitude IN NUMBER, latitude IN NUMBER)
RETURN SDO_GEOMETRY AS
```

```
BEGIN
  RETURN SDO_GEOMETRY(2001, 8307,SDO_POINT_TYPE(longitude, latitude, NULL),NULL, NULL);
END GET_CONVERTED_POINT;
/

create or replace
FUNCTION GET_TIMES_DIFF (begintime IN timestamp, endtime IN timestamp)
RETURN NUMBER AS
  timediff number;
BEGIN
  SELECT  (extract(DAY FROM endtime-begintime)*24*60*60)+
          (extract(HOUR FROM endtime-begintime)*60*60)+
          (extract(MINUTE FROM endtime-begintime)*60)+
          extract(SECOND FROM endtime-begintime)
  into timediff FROM dual;

  return timediff;
END GET_TIMES_DIFF;
/

create or replace
PROCEDURE POVOAMENTOOBJETOS AS
  cursor cursor_mos is (select distinct s.spl_obj_id from samples s);
  indx cursor_mos%rowtype;
BEGIN
  for indx in cursor_mos LOOP
    insert into moving_objects values(1, indx.spl_obj_id, trajectories_list_ty());
  END LOOP;
  commit;
END POVOAMENTOOBJETOS;
/

create or replace
PROCEDURE POVOAMENTO_POSICOES AS

  cursor cursor_s is
    select s.spl_obj_id, s.spl_dt, s.spl_speed, s.spl_x, s.spl_y
    from samples s
    order by s.spl_obj_id;
  indx cursor_s%rowtype;

  flagmo number;
  moref ref moving_object_ty;
BEGIN
  flagmo := -1;
  for indx in cursor_s LOOP

    if (flagmo = indx.spl_obj_id) then
      insert into positions values(
        position_ty(1, null, moref,
        GET_CONVERTED_POINT(indx.spl_x, indx.spl_y), indx.spl_dt, indx.spl_speed)
      );
    else
      select ref(mo) into moref
      from moving_objects mo
      where mo.mo_codigo = indx.spl_obj_id;

      flagmo := indx.spl_obj_id;
```

```
      insert into positions values(
        position_ty(1, null, moref,
        GET_CONVERTED_POINT(indx.spl_x, indx.spl_y), indx.spl_dt, indx.spl_speed)
      );
    end if;

  end LOOP;
  commit;
END POVOAMENTO_POSICOES;
/

create or replace
PROCEDURE POVOAMENTO_EVENTOS AS

  cursor cursor_e is
    select  e.evt_type as tipo, e.evt_dt_start as evtstart, e.evt_dt_end as evtend,
            e.evt_obj_id as objid, e.evt_x, e.evt_y
    from event e
    order by e.evt_obj_id, e.evt_dt_start;
  indx cursor_e%rowtype;

  evt event_ty;

  moref ref moving_object_ty;
BEGIN
  for indx in cursor_e LOOP
    select ref(mo) into moref
    from moving_objects mo
    where mo.mo_codigo = indx.objid;

    evt := event_ty(
      1, moref, null, indx.tipo,
      position_ty(1, null, moref,
        get_converted_point(indx.evt_x, indx.evt_y), indx.evtstart, null),
      position_ty(1, null, moref,
        get_converted_point(indx.evt_x, indx.evt_y), indx.evtend, null),
      null,
      null,
      null
    );

    insert into events values(evt);

  end LOOP;
END POVOAMENTO_EVENTOS;
/

create or replace
PROCEDURE POVOAMENTO_TRAJ_2 AS
  cursor cursor_p is
    select e.type, e.beginning.instant as evtstart, e.mo_id
    from events e
    where e.type = 'Inicio de viagem' or e.type = 'Final de viagem'
    order by e.mo_id.mo_codigo,e.beginning.instant;
  indx cursor_p%rowtype;

  inicio varchar2(100) := 'Inicio de viagem';
  fim varchar2(100) := 'Final de viagem';
```

```
  begintime timestamp;
  endtime timestamp;

  poslist POSITIONS_LIST_TY;
  evtlist EVENTS_LIST_TY;
  traj trajectory_ty;

  moid ref moving_object_ty := null;
BEGIN

  poslist := POSITIONS_LIST_TY();
  evtlist := EVENTS_LIST_TY();

  for indx in cursor_p LOOP
    if (indx.type = inicio) then
      begintime := indx.evtstart;
    else
      endtime := indx.evtstart;

      select ref(p) BULK COLLECT into poslist
      from positions p
      where p.mo_id = indx.mo_id and
            p.instant >= begintime and
            p.instant <= endtime
      order by p.instant;

      select ref(e) bulk collect into evtlist
      from events e
      where e.mo_id = indx.mo_id and
            e.beginning.instant >= begintime and
            e.beginning.instant <= endtime
      order by e.beginning.instant;

      traj := trajectory_ty(1, indx.mo_id, null, null,
        null, null, null, null, null, null, poslist, evtlist);

      insert into trajectories values(traj);
    end if;
  end LOOP;

  commit;
END POVOAMENTO_TRAJ_2;
/

create or replace
PROCEDURE POVOAMENTO_OBJ_TRAJ AS

  cursor cursor_o is select mo.mo_id from moving_objects mo;
  indx cursor_o%rowtype;

  moref ref moving_object_ty;
BEGIN
  for indx in cursor_o LOOP

    select ref(mo) into moref
    from moving_objects mo
    where mo.mo_id = indx.mo_id;

    insert into table (select mo.trajectories_lst
```

```
      from moving_objects mo where mo.mo_id = indx.mo_id)
        select ref(t)
        from trajectories t
        where t.mo_id = moref;

  end LOOP;
  commit;
END POVOAMENTO_OBJ_TRAJ;
/

create or replace
PROCEDURE SET_TRAJS_MEASURES AS
  cursor cursor_t is select t.tr_id from trajectories t;
  indx cursor_t%rowtype;
  traj trajectory_ty;
  pointref ref position_ty;
BEGIN
  for indx in cursor_t LOOP

    select value(t) into traj
    from trajectories t
    where t.tr_id = indx.tr_id;

    traj.set_beginning;
    traj.set_ending;
    traj.set_duration;
    traj.set_trlength;
    traj.set_line;
    traj.set_avgspeed;

    update trajectories t
    set t.beginning = traj.beginning
    where t.tr_id = traj.tr_id;

  end loop;
  commit;
END SET_TRAJS_MEASURES;
/

create or replace
PROCEDURE SET_EVTS_MEASURES AS
  cursor cursor_e is select e.e_id from events e;
  indx cursor_e%rowtype;
  evt event_ty;
BEGIN
  for indx in cursor_e LOOP

    select value(e) into evt
    from events e
    where e.e_id = indx.e_id;

    evt.set_duration;

    update events e
    set e.duration = evt.duration
    where e.e_id = evt.e_id;

  end loop;
  commit;
```

```
END SET_EVTS_MEASURES;
/
```

A.3.1 Procedimentos auxiliares

Listing A.4: implementação dos procedimentos e funções auxiliares

```
create or replace
PROCEDURE POVOAMENTO_EVT_TR AS
  cursor cursor_t is select t.tr_id from trajectories t;
  indx cursor_t%rowtype;

  trajref ref trajectory_ty;
BEGIN
  for indx in cursor_t LOOP

    select ref(t) into trajref
    from trajectories t
    where t.tr_id = indx.tr_id;

    update events e
    set e.tr_id = trajref
    where e.e_id in (
            select e1.column_value.e_id
            from trajectories t, table(t.events_lst) e1
            where t.tr_id = indx.tr_id
          );

  end loop;
  commit;
END POVOAMENTO_EVT_TR;
/

create or replace
FUNCTION TRAJECTORY_EVT_SEQ (
    TRAJ       IN TRAJECTORY_TY
  , EVT_TYPE_1 IN VARCHAR2
  , EVT_TYPE_2 IN VARCHAR2
  , EVT_TYPE_3 IN VARCHAR2
)
RETURN VARCHAR2 AS
  tr trajectory_ty := traj;
  cursor cursor_e is
    select e.column_value.type as tipo
    from table(tr.events_lst) e
    order by e.column_value.beginning.instant;
  indx cursor_e%rowtype;

  flag1 number := 0; flag2 number := 0; flag3 number := 0;
BEGIN
  for indx in cursor_e LOOP
    if (flag1 = 0 and flag2 = 0 and flag3 = 0) then
      if (indx.tipo = evt_type_1) then
        flag1 := 1;
      end if;
    end if;

    if (flag1 = 1 and flag2 = 0 and flag3 = 0) then
```

```
      if (indx.tipo = evt_type_2) then
        flag2 := 1;
      end if;
    end if;

    if (flag1 = 1 and flag2 = 1 and flag3 = 0) then
      if (indx.tipo = evt_type_3) then
        flag3 := 1;
      end if;
    end if;
  end LOOP;

  if (flag1 = 1 and flag2 = 1 and flag3 = 1) then
    return 'TRUE';
  else
    return 'FALSE';
  end if;

END TRAJECTORY_EVT_SEQ;
/

create or replace
FUNCTION TRAJECTORY_RECORT (traj in trajectory_ty, inicio in timestamp, fim in timestamp)
RETURN TRAJECTORY_TY AS

  traux trajectory_ty;
  poslist POSITIONS_LIST_TY;
  evtlist EVENTS_LIST_TY;

  cont number;

BEGIN

  select count(p.column_value) into cont
  from trajectories t, table(t.positions_lst) p
  where t.tr_id = traj.tr_id and
        p.column_value.instant >= inicio and
        p.column_value.instant <= fim;

  if (cont >= 2) then

    select p.column_value bulk collect into poslist
    from trajectories t, table(t.positions_lst) p
    where t.tr_id = traj.tr_id and
          p.column_value.instant >= inicio and
          p.column_value.instant <= fim
    order by p.column_value.instant;

    select e.column_value bulk collect into evtlist
    from trajectories t, table(t.events_lst) e
    where t.tr_id = traj.tr_id and
          e.column_value.beginning.instant >= inicio and
          e.column_value.ending.instant <= fim
    order by e.column_value.beginning.instant;

    traux := trajectory_ty(1, traj.mo_id, null, null,

    return traux;
  else
```

```
    return null;
  end if;

END TRAJECTORY_RECORT;
/

create or replace
FUNCTION TRAJECTORY_RECORT_AREA (TRAJ IN TRAJECTORY_TY, AREA IN AREA_TY)
RETURN TRAJECTORY_TY AS
  inicio timestamp;
  fim timestamp;

  cont number;
BEGIN

  cont := -1;

  select count(p.column_value) into cont
  from table (traj.positions_lst) p
  where SDO_GEOM.RELATE(p.column_value.spatial_point,'inside', area.region, 0.005) =
    'INSIDE';

  if (cont >= 2) then
    select min(p.column_value.instant), max(p.column_value.instant) into inicio, fim
    from table (traj.positions_lst) p
    where SDO_GEOM.RELATE(p.column_value.spatial_point,'inside', area.region, 0.005) =
      'INSIDE';

    return trajectory_recort(traj, inicio, fim);
  else
    return null;
  end if;
END TRAJECTORY_RECORT_AREA;
/

create or replace
TYPE PTAGGIMPL_AVG_SPEED AS OBJECT (
  sum_speed number,
  count_positions number,

  static function ODCIAggregateInitialize(pos_speed IN OUT PTAGGIMPL_AVG_SPEED)
  return number,

  member function ODCIAggregateIterate(self IN OUT PTAGGIMPL_AVG_SPEED,
    value IN position_ty)
  return number,

  member function ODCIAggregateTerminate(self IN PTAGGIMPL_AVG_SPEED,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT PTAGGIMPL_AVG_SPEED,
    position2 IN PTAGGIMPL_AVG_SPEED)
  return number
);
/

create or replace
TYPE BODY PTAGGIMPL_AVG_SPEED AS
```

```
  static function ODCIAggregateInitialize(pos_speed IN OUT PTAGGIMPL_AVG_SPEED)
  return number AS
  BEGIN
    pos_speed := PTAGGIMPL_AVG_SPEED(0,0);
    return ODCIConst.Success;
  END ODCIAggregateInitialize;

  member function ODCIAggregateIterate(self IN OUT PTAGGIMPL_AVG_SPEED,
    value IN position_ty)
  return number AS
  BEGIN
    self.sum_speed := self.sum_speed + value.speed;
    self.count_positions := self.count_positions + 1;
    return ODCIConst.Success;
  END ODCIAggregateIterate;

  member function ODCIAggregateTerminate(self IN PTAGGIMPL_AVG_SPEED,
    returnValue OUT number, flags IN number)
  return number AS
  BEGIN
    if self.count_positions = 0 then
      returnValue := 0;
    else
      returnValue := self.sum_speed / self.count_positions;
    end if;
    return ODCIConst.Success;
  END ODCIAggregateTerminate;

  member function ODCIAggregateMerge(self IN OUT PTAGGIMPL_AVG_SPEED,
    position2 IN PTAGGIMPL_AVG_SPEED)
  return number AS
  BEGIN
    self.sum_speed := self.sum_speed + position2.sum_speed;
    self.count_positions := self.count_positions + position2.count_positions;
    return ODCIConst.Success;
  END ODCIAggregateMerge;

END;
/

create or replace
FUNCTION PT_AVG_SPEED (input position_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING PTAGGIMPL_AVG_SPEED;
/

create or replace
type TrAggImpl_AVG_QTD_CLI as object
(

  sum_vm NUMBER,
  cont_tr NUMBER,

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_QTD_CLI)
    return number,

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_QTD_CLI,
    value IN trajectory_ty)
  return number,
```

```
  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_QTD_CLI,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_QTD_CLI,
    traj2 IN TrAggImpl_AVG_QTD_CLI)
  return number
);
```

A.4 Funções de agregação

Listing A.5: implementação das funções de agregação

```
create or replace
type TrAggImpl_AVG_Speed as object
(

  sum_vm NUMBER,
  cont_tr NUMBER,

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_Speed)
    return number,

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_Speed,
    value IN trajectory_ty)
  return number,

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_Speed,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_Speed,
    traj2 IN TrAggImpl_AVG_Speed)
  return number
);
/

create or replace
type body TrAggImpl_AVG_Speed is

static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_Speed)
return number is
begin
  trajavg := TrAggImpl_AVG_Speed(0,0);
  return ODCIConst.Success;
end;

member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_Speed,
  value IN trajectory_ty)
return number is
begin
  self.sum_vm := self.sum_vm + value.speed;
  self.cont_tr := self.cont_tr + 1;
  return ODCIConst.Success;
end;

member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_Speed,
```

```
  returnValue OUT number, flags IN number)
return number is
begin
  if self.cont_tr = 0 then
    returnValue := 0;
  else
    returnValue := self.sum_vm / self.cont_tr;
  end if;
  return ODCIConst.Success;
end;

member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_Speed,
    traj2 IN TrAggImpl_AVG_Speed)
return number is
begin
  self.sum_vm := self.sum_vm + traj2.sum_vm;
  self.cont_tr := self.cont_tr + traj2.cont_tr;
  return ODCIConst.Success;
end;

end;
/

create or replace
FUNCTION TR_AVG_SPEED (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TrAggImpl_AVG_Speed;
/

create or replace
TYPE TRAGGIMPL_MAX_LENGTH AS OBJECT (

  biggest number,

  static function ODCIAggregateInitialize(traj IN OUT TRAGGIMPL_MAX_LENGTH)
    return number,

  member function ODCIAggregateIterate(self IN OUT TRAGGIMPL_MAX_LENGTH,
    value IN trajectory_ty)
  return number,

  member function ODCIAggregateTerminate(self IN TRAGGIMPL_MAX_LENGTH,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TRAGGIMPL_MAX_LENGTH,
    traj2 IN TRAGGIMPL_MAX_LENGTH)
  return number
);
/

create or replace
TYPE BODY TRAGGIMPL_MAX_LENGTH AS

  static function ODCIAggregateInitialize(traj IN OUT TRAGGIMPL_MAX_LENGTH)
    return number AS
  BEGIN
    traj := TRAGGIMPL_MAX_LENGTH(-1);
    RETURN ODCIConst.Success;
  END ODCIAggregateInitialize;
```

```
  member function ODCIAggregateIterate(self IN OUT TRAGGIMPL_MAX_LENGTH,
    value IN trajectory_ty)
  return number AS
  BEGIN
    if value.length > self.biggest then
      self.biggest := value.length;
    end if;
    RETURN ODCIConst.Success;
  END ODCIAggregateIterate;

  member function ODCIAggregateTerminate(self IN TRAGGIMPL_MAX_LENGTH,
    returnValue OUT number, flags IN number)
  return number AS
  BEGIN
    returnValue := self.biggest;
    return ODCIConst.Success;
  END ODCIAggregateTerminate;

  member function ODCIAggregateMerge(self IN OUT TRAGGIMPL_MAX_LENGTH,
    traj2 IN TRAGGIMPL_MAX_LENGTH)
  return number AS
  BEGIN
    if traj2.biggest > self.biggest then
      self.biggest := traj2.biggest;
    end if;
    return ODCIConst.Success;
  END ODCIAggregateMerge;

END;
/

create or replace
FUNCTION TR_MAX_LENGTH (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TRAGGIMPL_MAX_LENGTH;
/

create or replace
type TrAggImpl_AVG_contAtendimento as object
(

  contEvt NUMBER,
  contTr NUMBER,

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_contAtendimento)
  return number,

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_contAtendimento,
    value IN trajectory_ty)
  return number,

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_contAtendimento,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_contAtendimento,
    traj2 IN TrAggImpl_AVG_contAtendimento)
  return number
);
```

```
/

create or replace
type body TrAggImpl_AVG_contAtendimento is

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_contAtendimento)
  return number is
  begin
    trajavg := TrAggImpl_AVG_contAtendimento(0,0);
    return ODCIConst.Success;
  end;

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_contAtendimento,
    value IN trajectory_ty)
  return number is
  begin
    self.contEvt := self.contEvt + value.get_cont_evt('Atendimento');
    self.contTr := self.contTr + 1;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_contAtendimento,
    returnValue OUT number, flags IN number)
  return number is
  begin
    if self.contTr = 0 then
      returnValue := 0;
    else
      returnValue := self.contEvt / self.contTr;
    end if;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_contAtendimento,
      traj2 IN TrAggImpl_AVG_contAtendimento)
  return number is
  begin
    self.contEvt := self.contEvt + traj2.contEvt;
    self.contTr := self.contTr + traj2.contTr;
    return ODCIConst.Success;
  end;

end;
/

create or replace
FUNCTION TR_AVG_CONTAtendimento (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TrAggImpl_AVG_contAtendimento;
/

create or replace
TYPE TRAGGIMPL_SUM_EVT_PARADA AS OBJECT  (
    sum_evt NUMBER,

  static function ODCIAggregateInitialize(trajavg IN OUT TRAGGIMPL_SUM_EVT_PARADA)
    return number,

  member function ODCIAggregateIterate(self IN OUT TRAGGIMPL_SUM_EVT_PARADA,
    value IN trajectory_ty)
```

```
  return number,

  member function ODCIAggregateTerminate(self IN TRAGGIMPL_SUM_EVT_PARADA,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TRAGGIMPL_SUM_EVT_PARADA,
    traj2 IN TRAGGIMPL_SUM_EVT_PARADA)
  return number
);
/

create or replace
TYPE BODY TRAGGIMPL_SUM_EVT_PARADA AS

  static function ODCIAggregateInitialize(trajavg IN OUT TRAGGIMPL_SUM_EVT_PARADA)
  return number is
  begin
    trajavg := TRAGGIMPL_SUM_EVT_PARADA(0);
    return ODCIConst.Success;
  end;

  member function ODCIAggregateIterate(self IN OUT TRAGGIMPL_SUM_EVT_PARADA,
    value IN trajectory_ty)
  return number is
    aux number := 0;
  begin
    aux := value.get_sum_dur_evt('Parada');
    self.sum_evt := self.sum_evt + aux;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateTerminate(self IN TRAGGIMPL_SUM_EVT_PARADA,
    returnValue OUT number, flags IN number)
  return number is
  begin
    returnValue := self.sum_evt;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateMerge(self IN OUT TRAGGIMPL_SUM_EVT_PARADA,
    traj2 IN TRAGGIMPL_SUM_EVT_PARADA)
  return number is
  begin
    self.sum_evt := self.sum_evt + traj2.sum_evt;
    return ODCIConst.Success;
  end;

END;
/

create or replace
FUNCTION TR_SUM_EVT_PARADA (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TRAGGIMPL_SUM_EVT_PARADA;
/

create or replace
type TrAggImpl_AVG_sumEvtParada as object
(
```

```
  sumParada NUMBER,
  contEvt NUMBER,

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_sumEvtParada)
    return number,

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_sumEvtParada,
    value IN trajectory_ty)
  return number,

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_sumEvtParada,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_sumEvtParada,
    traj2 IN TrAggImpl_AVG_sumEvtParada)
  return number
);
/

create or replace
type body TrAggImpl_AVG_sumEvtParada is

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_sumEvtParada)
  return number is
  begin
    trajavg := TrAggImpl_AVG_sumEvtParada(0,0);
    return ODCIConst.Success;
  end;

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_sumEvtParada,
    value IN trajectory_ty)
  return number is
  begin
    self.sumParada := self.sumParada + value.get_sum_dur_evt('Parada');
    self.contEvt := self.contEvt + 1;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_sumEvtParada,
    returnValue OUT number, flags IN number)
  return number is
  begin
    if self.contEvt = 0 then
      returnValue := 0;
    else
      returnValue := self.sumParada / self.contEvt;
    end if;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_sumEvtParada,
    traj2 IN TrAggImpl_AVG_sumEvtParada)
  return number is
  begin
    self.sumParada := self.sumParada + traj2.sumParada;
    self.contEvt := self.contEvt + traj2.contEvt;
    return ODCIConst.Success;
```

```
  end;

end;
/

create or replace
FUNCTION TR_AVG_SUMEVTPARADA (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TRAGGIMPL_AVG_SUMEVTPARADA;
/

create or replace
TYPE TRAGGIMPL_SUM_EVT_ATENDIMENTO AS OBJECT  (
    sum_evt NUMBER,

  static function ODCIAggregateInitialize(trajavg IN OUT TRAGGIMPL_SUM_EVT_ATENDIMENTO)
    return number,

  member function ODCIAggregateIterate(self IN OUT TRAGGIMPL_SUM_EVT_ATENDIMENTO,
    value IN trajectory_ty)
  return number,

  member function ODCIAggregateTerminate(self IN TRAGGIMPL_SUM_EVT_ATENDIMENTO,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TRAGGIMPL_SUM_EVT_ATENDIMENTO,
    traj2 IN TRAGGIMPL_SUM_EVT_ATENDIMENTO)
  return number
);
/

create or replace
TYPE BODY TRAGGIMPL_SUM_EVT_ATENDIMENTO AS

  static function ODCIAggregateInitialize(trajavg IN OUT TRAGGIMPL_SUM_EVT_ATENDIMENTO)
  return number is
  begin
    trajavg := TRAGGIMPL_SUM_EVT_ATENDIMENTO(0);
    return ODCIConst.Success;
  end;

  member function ODCIAggregateIterate(self IN OUT TRAGGIMPL_SUM_EVT_ATENDIMENTO,
    value IN trajectory_ty)
  return number is
    aux number := 0;
  begin
    aux := value.get_sum_dur_evt('Atendimento');
    self.sum_evt := self.sum_evt + aux;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateTerminate(self IN TRAGGIMPL_SUM_EVT_ATENDIMENTO,
    returnValue OUT number, flags IN number) return number is
  begin
    returnValue := self.sum_evt;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateMerge(self IN OUT TRAGGIMPL_SUM_EVT_ATENDIMENTO,
```

```
    traj2 IN TRAGGIMPL_SUM_EVT_ATENDIMENTO)
  return number is
  begin
    self.sum_evt := self.sum_evt + traj2.sum_evt;
    return ODCIConst.Success;
  end;

END;
/

create or replace
FUNCTION TR_SUM_EVT_ATENDIMENTO (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TRAGGIMPL_SUM_EVT_ATENDIMENTO;
/

create or replace
type TrAggImpl_AVG_QTD_Length as object
(

  sumQtd NUMBER,
  sumLength NUMBER,

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_QTD_Length)
    return number,

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_QTD_Length,
    value IN trajectory_ty)
  return number,

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_QTD_Length,
    returnValue OUT number, flags IN number)
  return number,

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_QTD_Length,
    traj2 IN TrAggImpl_AVG_QTD_Length)
  return number
);
/

create or replace
type body TrAggImpl_AVG_QTD_Length is

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_QTD_Length)
  return number is
  begin
    trajavg := TrAggImpl_AVG_QTD_Length(0,0);
    return ODCIConst.Success;
  end;

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_QTD_Length,
    value IN trajectory_ty)
  return number is
  begin
    self.sumQtd := self.sumQtd + value.get_sum_value1_evt('Atendimento');
    self.sumLength := self.sumLength + value.length;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_QTD_Length,
```

```
    returnValue OUT number, flags IN number)
  return number is
  begin
    if self.sumLength = 0 then
      returnValue := 0;
    else
      returnValue := self.sumQtd / self.sumLength;
    end if;
    return ODCIConst.Success;
  end;

  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_QTD_Length,
    traj2 IN TrAggImpl_AVG_QTD_Length)
  return number is
  begin
    self.sumQtd := self.sumQtd + traj2.sumQtd;
    self.sumLength := self.sumLength + traj2.sumLength;
    return ODCIConst.Success;
  end;

end;
/

create or replace
FUNCTION TR_AVG_QTD_LENGTH (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TrAggImpl_AVG_QTD_LENGTH;
/

create or replace
type body TrAggImpl_AVG_QTD_CLI is

  static function ODCIAggregateInitialize(trajavg IN OUT TrAggImpl_AVG_QTD_CLI)
  return number is
  begin
    trajavg := TrAggImpl_AVG_QTD_CLI(0,0);
    return ODCIConst.Success;
  end;

  member function ODCIAggregateIterate(self IN OUT TrAggImpl_AVG_QTD_CLI,
    value IN trajectory_ty)
  return number is
  begin
    self.sum_vm := self.sum_vm + value.get_sum_value1_evt('Atendimento');
    self.cont_tr := self.cont_tr + value.get_cont_evt('Atendimento');
    return ODCIConst.Success;
  end;

  member function ODCIAggregateTerminate(self IN TrAggImpl_AVG_QTD_CLI,
    returnValue OUT number, flags IN number)
  return number is
  begin
    if self.cont_tr = 0 then
      returnValue := 0;
    else
      returnValue := self.sum_vm / self.cont_tr;
    end if;
    return ODCIConst.Success;
  end;
```

```
  member function ODCIAggregateMerge(self IN OUT TrAggImpl_AVG_QTD_CLI,
    traj2 IN TrAggImpl_AVG_QTD_CLI)
  return number is
  begin
    self.sum_vm := self.sum_vm + traj2.sum_vm;
    self.cont_tr := self.cont_tr + traj2.cont_tr;
    return ODCIConst.Success;
  end;

end;
/

create or replace
FUNCTION TR_AVG_QTD_CLI (input trajectory_ty) RETURN NUMBER
  PARALLEL_ENABLE AGGREGATE USING TrAggImpl_AVG_QTD_CLI;
/
```

APÊNDICE B – CÓDIGOS DE CRIAÇÃO DO AMBIENTE MULTIDIMENSIONAL

Neste apêndice apresentamos os scripts e códigos de implementação do ambiente multidimensional utilizado nos experimentos.

B.1 Tipos

Listing B.1: implementação dos tipos

```
CREATE OR REPLACE TYPE dw_fact_trajectory_ty
;
/

CREATE OR REPLACE TYPE dw_movingobject_ty
;
/

CREATE OR REPLACE TYPE dw_time_ty
;
/

CREATE OR REPLACE TYPE dw_trajectory_ty
;
/

CREATE OR REPLACE TYPE dw_fact_event_ty
;
/

create or replace type dw_region_ty
;
/

CREATE OR REPLACE TYPE dw_event_ty
    AS OBJECT
    (
        begintime Timestamp(6) ,
        endtime Timestamp(6) ,
        beginning SDO_GEOMETRY,
        ending  SDO_GEOMETRY,
```

```
        value1 number,
        value2 number,
        duration NUMERIC
    ) NOT FINAL
;
/

CREATE OR REPLACE TYPE dw_evt_type_ty
    AS OBJECT
    (
        type_id NUMERIC ,
        type VARCHAR2 (100)
    ) NOT FINAL
;
/

CREATE OR REPLACE TYPE dw_fact_event_ty
    AS OBJECT
    (
        fact_evt_id NUMERIC ,
        dw_event dw_event_ty ,
        begintime_fk NUMERIC ,
        endtime_fk NUMERIC ,
        mo_fk NUMERIC ,
        evt_type_fk NUMERIC ,
        region_fk NUMERIC ,
        tr_fk NUMERIC
    ) FINAL
;
/

CREATE OR REPLACE TYPE dw_fact_trajectory_ty
    AS OBJECT
    (
        fact_tr_id NUMERIC ,
        dw_trajectory dw_trajectory_ty ,
        begintime_fk NUMERIC ,
        endtime_fk NUMERIC ,
        mo_fk NUMERIC,
        region_fk NUMERIC
    ) FINAL
;
/

CREATE OR REPLACE TYPE dw_movingobject_ty
    AS OBJECT
    (
        mo_id NUMERIC ,
        enterprise VARCHAR2 (30) ,
        branch VARCHAR2 (30) ,
        base VARCHAR2 (30) ,
        mo_codigo NUMBER
    ) FINAL
;
/

CREATE OR REPLACE TYPE dw_time_ty
    AS OBJECT
    (
```

```
        t_id NUMERIC ,
        year NUMBER (4) ,
        month NUMBER (2) ,
        day NUMBER (2)
    ) FINAL
;
/

CREATE OR REPLACE TYPE dw_trajectory_ty
    AS OBJECT
    (
        begintime Timestamp(6) ,
        endtime Timestamp(6) ,
        beginposition SDO_GEOMETRY ,
        endposition SDO_GEOMETRY ,
        speed NUMBER ,
        accelerate NUMBER ,
        direction VARCHAR2 (25) ,
        duration NUMBER ,
        length NUMBER ,
        line SDO_GEOMETRY ,

        positions_lst bruno5.positions_list_ty ,
        events_lst bruno5.events_list_ty,

        member function get_cont_evt(evtType in varchar2) return number ,
        member function get_sum_dur_evt(evtType in varchar2) return number ,
        member function get_sum_value1_evt(evtType in varchar2) return number ,
        member function get_sum_value2_evt(evtType in varchar2) return number
    ) FINAL
;
/

create or replace type dw_region_ty as object(
  r_id numeric,
  country varchar2(100),
  region varchar2(100),
  state varchar2(100),
  city varchar2(100)
);
/
```

B.2 Tabelas

Listing B.2: implementação das tabelas

```
CREATE TABLE dw_dim_mo
    OF dw_movingobject_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    mo_id  NOT NULL ,
    enterprise  NOT NULL ,
    branch  NOT NULL ,
    base  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;
```

```
CREATE TABLE dw_dim_region
    OF dw_region_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    r_id  NOT NULL ,
    country  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;


CREATE TABLE dw_dim_time
    OF dw_time_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    t_id  NOT NULL ,
    year  NOT NULL ,
    month  NOT NULL ,
    day  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;

CREATE TABLE dw_fact_trajectory
    OF dw_fact_trajectory_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    fact_tr_id  NOT NULL ,
    begintime_fk  NOT NULL ,
    endtime_fk  NOT NULL ,
    mo_fk  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
    nested table dw_trajectory.positions_lst store as positions_nt
    nested table dw_trajectory.events_lst store as events_nt
;

CREATE TABLE dw_dim_evt_type
    OF dw_evt_type_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    type_id  NOT NULL ,
    type  NOT NULL
    )
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;

CREATE TABLE dw_fact_event
    OF dw_fact_event_ty
    SUBSTITUTABLE AT ALL LEVELS
    (
    fact_evt_id  NOT NULL ,
    dw_event  NOT NULL ,
    begintime_fk  NOT NULL ,
    endtime_fk  NOT NULL ,
    mo_fk  NOT NULL ,
    evt_type_fk  NOT NULL
    )
```

```
    OBJECT IDENTIFIER IS SYSTEM GENERATED
;

ALTER TABLE dw_fact_event
ADD CONSTRAINT DW_FACT_EVT_PK PRIMARY KEY
(
  fact_evt_id
)
ENABLE;

ALTER TABLE DW_DIM_EVT_TYPE
ADD CONSTRAINT DW_DIM_EVT_TYPE_PK PRIMARY KEY
(
  TYPE_ID
)
ENABLE;

ALTER TABLE DW_DIM_MO
ADD CONSTRAINT DW_DIM_MO_PK PRIMARY KEY
(
  MO_ID
)
ENABLE;

ALTER TABLE DW_DIM_REGION
ADD CONSTRAINT DW_DIM_REGION_PK PRIMARY KEY
(
  R_ID
)
ENABLE;

ALTER TABLE DW_DIM_TIME
ADD CONSTRAINT DW_DIM_TIME_PK PRIMARY KEY
(
  T_ID
)
ENABLE;

ALTER TABLE DW_FACT_TRAJECTORY
ADD CONSTRAINT DW_FACT_TR_PK PRIMARY KEY
(
  FACT_TR_ID
)
ENABLE;


ALTER TABLE dw_fact_trajectory
    ADD CONSTRAINT fact_tr_begintime_fk FOREIGN KEY
    (
     begintime_fk
    )
    REFERENCES dw_dim_time
;

ALTER TABLE dw_fact_trajectory
    ADD CONSTRAINT fact_tr_region_fk FOREIGN KEY
    (
     region_fk
    )
```

```
    REFERENCES dw_dim_region
;


ALTER TABLE dw_fact_trajectory
    ADD CONSTRAINT fact_tr_endime_fk FOREIGN KEY
    (
     endtime_fk
    )
    REFERENCES dw_dim_time
;


ALTER TABLE dw_fact_trajectory
    ADD CONSTRAINT fact_tr_mo_fk FOREIGN KEY
    (
     mo_fk
    )
    REFERENCES dw_dim_mo
;

ALTER TABLE dw_fact_event
    ADD CONSTRAINT e_begintime_FK FOREIGN KEY
    (
     begintime_fk
    )
    REFERENCES dw_dim_time
;


ALTER TABLE dw_fact_event
    ADD CONSTRAINT e_region_FK FOREIGN KEY
    (
     region_fk
    )
    REFERENCES dw_dim_region
;

ALTER TABLE dw_fact_event
    ADD CONSTRAINT e_endtime_FK FOREIGN KEY
    (
     endtime_fk
    )
    REFERENCES dw_dim_time
;


ALTER TABLE dw_fact_event
    ADD CONSTRAINT e_mo_FK FOREIGN KEY
    (
     mo_fk
    )
    REFERENCES dw_dim_mo
;


ALTER TABLE dw_fact_event
    ADD CONSTRAINT e_type_FK FOREIGN KEY
    (
```

```
     evt_type_fk
    )
    REFERENCES dw_dim_evt_type
;

ALTER TABLE dw_fact_event
    ADD CONSTRAINT e_tr_FK FOREIGN KEY
    (
     tr_fk
    )
    REFERENCES dw_fact_trajectory
;

CREATE SEQUENCE DW_DIM_TIME_PK_SEQ
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 999999999999999999999999999
    MINVALUE 1
    NOCACHE
;

CREATE OR REPLACE TRIGGER DW_DIM_TIME_PK
    BEFORE INSERT ON dw_dim_time
    FOR EACH ROW
    BEGIN
 select DW_DIM_TIME_PK_SEQ.nextval
 into :new.t_id
 from dual;

 END;
/

CREATE SEQUENCE DW_DIM_MO_SEQ
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 9999999999999999999999999999
    MINVALUE 1
    NOCACHE
;

CREATE OR REPLACE TRIGGER DW_DIM_MO_PK
    BEFORE INSERT ON dw_dim_mo
    FOR EACH ROW
    BEGIN
 select DW_DIM_MO_SEQ.nextval
 into :new.mo_id
 from dual;

 END;
/

CREATE SEQUENCE DW_FACT_TRAJECTORY_SEQ
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 999999999999999999999999999
    MINVALUE 1
    NOCACHE
;
```

```
CREATE OR REPLACE TRIGGER DW_FACT_TRAJECTORY_PK
    BEFORE INSERT ON dw_fact_trajectory
    FOR EACH ROW
    BEGIN
 select DW_FACT_TRAJECTORY_SEQ.nextval
 into :new.fact_tr_id
 from dual;

 END;
/

CREATE SEQUENCE DW_EVT_TYPE_SEQ
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 9999999999999999999999999999
    MINVALUE 1
    NOCACHE
;

CREATE OR REPLACE TRIGGER DW_EVT_TYPE_PK
    BEFORE INSERT ON dw_dim_evt_type
    FOR EACH ROW
    BEGIN
 select DW_EVT_TYPE_SEQ.nextval
 into :new.type_id
 from dual;

 END;
/

CREATE SEQUENCE DW_FACT_EVT_SEQ
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 9999999999999999999999999999
    MINVALUE 1
    NOCACHE
;

CREATE OR REPLACE TRIGGER DW_FACT_EVT_PK
    BEFORE INSERT ON dw_fact_event
    FOR EACH ROW
    BEGIN
 select DW_FACT_EVT_SEQ.nextval
 into :new.fact_evt_id
 from dual;

 END;
/

CREATE SEQUENCE DW_REGION_SEQ
    START WITH 1
    INCREMENT BY 1
    MAXVALUE 9999999999999999999999999999
    MINVALUE 1
    NOCACHE
;

CREATE OR REPLACE TRIGGER DW_REGION_PK
    BEFORE INSERT ON dw_dim_region
```

```
    FOR EACH ROW
    BEGIN
 select DW_REGION_SEQ.nextval
 into :new.r_id
 from dual;

 END;
/
```

B.3 Procedimentos e Funções

Listing B.3: implementação dos procedimentos e funções

```
create or replace
PROCEDURE DW_POPULATE_DIM_MO AS
  cursor cursor_mo_codigo is
    (select distinct(mo.mo_codigo) from moving_objects mo);
  indx cursor_mo_codigo%rowtype;

  enterprise varchar2(10);
  branch varchar2(10);
  base varchar2(10);
  veiculo number;
BEGIN
  for indx in cursor_mo_codigo LOOP
    insert into bruno6.dw_dim_mo
      values (1,'liquigas','sp','base1', indx.mo_codigo);
  end loop;
  commit;
END DW_POPULATE_DIM_MO;
/

create or replace
PROCEDURE DW_POPULATE_DIM_TIME AS
  start_date timestamp :=
    TO_TIMESTAMP('01-01-2009␣00:00:00','dd-mm-yyyy␣HH24:MI:SS');
  end_date timestamp :=
    TO_TIMESTAMP('30-06-2011␣23:59:59','dd-mm-yyyy␣HH24:MI:SS');
  t_full_date TIMESTAMP;
  t_day_of_month NUMBER;
  t_month_number NUMBER;
  t_calendar_year NUMBER;
  t_hour_of_day NUMBER;
  t_minute_of_hour NUMBER;
  t_second_of_minute NUMBER;
BEGIN
  t_full_date := start_date;

  WHILE t_full_date <= end_date LOOP
    BEGIN
      t_day_of_month := extract(day from t_full_date);
      t_month_number := extract(month from t_full_date);
      t_calendar_year := extract(year from t_full_date);
      INSERT INTO bruno6.dw_dim_time
        VALUES (1, t_calendar_year, t_month_number, t_day_of_month);
      t_full_date := t_full_date + interval '1' day;
    END;
  END LOOP;
```

```
  commit;
END DW_POPULATE_DIM_TIME;
/

create or replace
PROCEDURE DW_POPULATE_DIM_EVENT_TYPE AS
  cursor cursor_et is select distinct(e.type) from events e;
  indx cursor_et%rowtype;
BEGIN
  for indx in cursor_et LOOP
    insert into bruno6.dw_dim_evt_type values(1, indx.type);
  end LOOP;
  commit;
END DW_POPULATE_DIM_EVENT_TYPE;
/

create or replace
PROCEDURE DW_POPULATE_FACT_TRAJECTORY AS
  cursor cursor_t is
    select  t.tr_id, t.mo_id.mo_codigo as mocodigo,
            t.beginning.instant as begintime,
            t.beginning.spatial_point as beginposition,
            t.ending.instant as endtime,
            t.ending.spatial_point as endposition,
            t.speed, t.accelerate, t.direction,
            t.duration, t.length, t.trline
    from  trajectories t;
  indx cursor_t%rowtype;

  dim_time_key_begin number;
  dim_time_key_end number;
  dim_mo_key number;
  dim_region_key number;
    rname varchar2(100); sname varchar2(100); cname varchar2(100);
    auxr varchar2(100); auxs varchar2(100); auxc varchar2(100);

  cursor cursor_r1 is
    select r.a_id, r.name, r.type, r.region
    from areas r
    where r.type = 'REGION';
  indxr cursor_r1%rowtype;

  cursor cursor_r2 is
    select r.a_id, r.name, r.type, r.region
    from areas r
    where r.type = 'STATE';
  indxs cursor_r2%rowtype;

  moaux number;

  poslist positions_list_ty;
  evtlist events_list_ty;

BEGIN
  for indx in cursor_t LOOP

    select dt.t_id into dim_time_key_begin
    from bruno6.dw_dim_time dt
    where dt.year = extract(year from indx.begintime) and
```

```
      dt.month = extract(month from indx.begintime) and
      dt.day = extract(day from indx.begintime);

select dt.t_id into dim_time_key_end
from bruno6.dw_dim_time dt
where dt.year = extract(year from indx.endtime) and
      dt.month = extract(month from indx.endtime) and
      dt.day = extract(day from indx.endtime);

dim_mo_key := null;
moaux := 0;

select count(*) into moaux
from bruno6.dw_dim_mo mo
where mo.mo_codigo = indx.mocodigo;

if (moaux = 0) then
  insert into bruno6.dw_dim_mo
    values(1, 'EmpresaX', '2', '1003', indx.mocodigo);

  select mo.mo_id into dim_mo_key
  from bruno6.dw_dim_mo mo
  where mo.mo_codigo = indx.mocodigo;
else
  select mo.mo_id into dim_mo_key
  from bruno6.dw_dim_mo mo
  where mo.mo_codigo = indx.mocodigo;
end if;

auxr := null;
rname := null;
for indxr in cursor_r1 LOOP
  SELECT SDO_GEOM.RELATE(indx.trline,'inside', r.region, 0.005) into auxr
  FROM areas r
  WHERE r.a_id = indxr.a_id;

  if (UPPER(auxr) = 'INSIDE') then
    rname := indxr.name;
  end if;
end LOOP;

sname := null;
auxs := null;
for indxs in cursor_r2 LOOP
  SELECT SDO_GEOM.RELATE(indx.trline,'inside', r.region, 0.005) into auxs
  FROM areas r
  WHERE r.a_id = indxs.a_id;

  if (UPPER(auxs) = 'INSIDE') then
    sname := indxs.name;
  end if;
end LOOP;

dim_region_key := null;
if (rname is not null) then
  if (sname is not null) then
    select r.r_id into dim_region_key
    from BRUNO6.dw_dim_region r
    where r.region = UPPER(rname) and
```

```
            r.state = UPPER(sname) and
            r.city is null;
      else
        select r.r_id into dim_region_key
        from BRUNO6.dw_dim_region r
        where r.region = rname and
            r.state is null;
      end if;
    end if;

    poslist := positions_list_ty();
    evtlist := events_list_ty();

    select t.positions_lst into poslist
    from trajectories t
    where t.tr_id = indx.tr_id;

    select t.events_lst into evtlist
    from trajectories t
    where t.tr_id = indx.tr_id;

    insert into bruno6.dw_fact_trajectory values(1,
      bruno6.dw_trajectory_ty(indx.begintime, indx.endtime, indx.beginposition,
        indx.endposition, indx.speed, indx.accelerate,
        indx.direction, indx.duration, indx.length, indx.trline, poslist, evtlist),
      dim_time_key_begin, dim_time_key_end, dim_mo_key, dim_region_key);
  end LOOP;
  commit;
END DW_POPULATE_FACT_TRAJECTORY;
```