UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

PTV – Pacote de Tempo de Validade para o SGBD Oracle

por

SANDRO FAVIN PINHEIRO

Trabalho de Conclusão submetido à avaliação,
como requisito parcial para a obtenção do grau de Mestre
em Ciência da Computação

Profa. Dra. Nina Edelweiss
Orientadora

Porto Alegre, março de 2003

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Pinheiro, Sandro Favin

PTV – Pacote de Tempo de Validade para o SGBD Oracle / por Sandro Favin Pinheiro. Porto Alegre: PPGC da UFRGS, 2003. 76 f.: il.

Trabalho de Conclusão (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR-RS, 2003. Orientadora: Edelweiss, Nina.

1. Banco de dados. 2. Banco de dados temporal. 3. Séries de tempo. I. Edelweiss, Nina. II. Título.

Sumário

Lista de Abreviaturas

BD	Banco de Dados
BDT	Banco de Dados Temporal
PL/SQL	*Procedural Language/Structured Query Language*
PTV	Pacote de Tempo de Validade
SGBD	Sistema Gerenciados de Banco de Dados
SGBDT	Sistema Gerenciador de Banco de Dados Temporal
SQL	*Structured Query Language*
TSC	*Time Series Cartridge*
TSDB	*Time Series Databases*

Lista de Figuras

Lista de Tabelas

Resumo

Embora há muito tempo já se tenha conhecimento da importância da recuperação de informações temporais, não existe um SGBD temporal que tenha sido desenvolvido unicamente para aplicações comerciais, que supra todas as necessidades e abranja todos os aspectos temporais necessários a estas aplicações.

O SGBD da Oracle, a partir da versão 8i, possibilita a inserção de características temporais similares às de tempo de transação no BD, através de um pacote de tempo denominado *Time Series Cartridge*. Entretanto, em muitos casos, a utilização deste cartucho de tempo não é suficiente para que se possa implementar por completo tudo o que foi especificado na modelagem do sistema. A modelagem completa da realidade só é alcançada se forem utilizadas em conjunto as características de tempo de transação e de tempo de validade no banco de dados.

Neste trabalho são sugeridos e implementados mecanismos para a inserção e o gerenciamento do tempo de validade no SGBD Oracle. O tempo de validade é administrado através da execução de funções, procedimentos, gatilhos e objetos, organizados em forma de um pacote, de maneira que este possa ser utilizado em conjunto com o *Time Series Cartridge.*

Palavras-chave: Banco de Dados, Banco de Dados Temporal, Séries de Tempo.

TITLE: “VALID TIME PACKAGE TO THE ORACLE DBMS”

Abstract

Although the importance of the temporal information retrieval is known from a long time, a temporal SGBD for commercial applications, supporting all the necessities and that embodies all the temporal aspects essential to these aplications has not get been developed.

The Oracle DBMS introduced in his version 8 a temporal package – the T*ime Series Cartridge*. The use of this cartridge allows the insertion of temporal properties similar to the temporal databases transaction time. However, in many practical applications this temporal information is not enough to implement all the temporal features identified in the system’s modeling. A complete model of reality is only possible if valid time and transaction time characteristics are used together.

Actions to insert and management the validity time using the Oracle DBMS are proposed and implemented in this work. Validity time is managed through functions, procedures, triggers and objetcs execution. These processes were organized in a package, *Valid Time Package*, whitch can be used together with the Time Series Package.

KeyWords : Data Bases, Temporal Data Bases, Time Series

.

1 Introdução

Informações temporais estão presentes em um grande número de sistemas que modelam o mundo real. No entanto, em muitos casos, no processo de construção de sistemas, as informações temporais são especificadas indutivamente e modeladas sem visar uma dimensão temporal, adequando-se aos recursos do banco de dados.

Mesmo com a inovação da tecnologia, dos meios de armazenamento e da compactação de dados, bem como das inúmeras pesquisas e implementações relativas a bancos de dados temporais (BDTs), os BDTs ainda encontram resistência dos fabricantes para que seus recursos sejam acoplados aos bancos de dados (BDs) de uso comercial.

A versão 8i do SGBD Oracle oferece alguns recursos para tratamento de informações temporais, através da instalação de um cartucho de tempo, chamado *Time Series Cartridge.* O *Time Series Cartridge* (TSC) é um cartucho composto de um conjunto de objetos, procedimentos e funções, que torna possível situar um dado ou objeto no tempo [ORA 99].

A possibilidade de adicionar o tempo de transação aos dados levou a diversas pesquisas na área de *Time Series Databases* (TSDB), levando em conta principalmente a possibilidade de descoberta de conhecimento através de mineração de dados [Last 2001, HÖP 2001, PER 2000].

Apesar do BD Oracle fornecer alguns mecanismos para tratamento de informações temporais, estes recursos são muito superficiais e ficam muito aquém da necessidade dos sistemas que modelam o mundo real. Bancos de dados temporais permitem armazenar e recuperar todos os estados de um objeto, registrando sua evolução ao longo do tempo [EDE 98]. As informações relativas ao tempo são associadas aos dados de forma implícita, através do tempo de transação (tempo em que a informação foi inserida no banco de dados), do tempo de validade (tempo em que a informação modela a realidade), ou de forma explícita, através de um tempo definido pelo usuário [JEN 98].

No processo de construção de sistemas, na maioria dos casos a modelagem completa da realidade só é alcançada se forem utilizadas em conjunto as características de banco de dados de tempo de transação e de tempo de validade, possibilitando o acesso a todos os estados de um dado em um banco de dados. Estes bancos de dados são denominados banco de dados bitemporais [SNO 85].

Ao utilizar o TSC, as informações temporais são inseridas sobre o BD objeto-relacional convencional. O TSC pode ser comparado a uma ferramenta de apoio à inserção do tempo de transação, onde cada atributo ou tupla do banco de dados é associado a um rótulo temporal, ou seja, a um atributo que os situa no tempo.

1.1 Objetivo do Trabalho

O objetivo deste trabalho é buscar mecanismos para que aplicações possam ser implementadas em SGBDs comerciais mantendo ao máximo as características temporais especificadas no projeto do sistema. Para alcançar este objetivo, neste trabalho são sugeridos e implementados recursos para a inserção, a manutenção e o gerenciamento do tempo de validade em BD objeto-relacionais.

Os mecanismos implementados são independentes da arquitetura do cartucho de série de tempo existente no Oracle, permitindo assim a execução paralela de ambos mecanismos, buscando resultados próximos aos alcançados por BD bitemporais.

1.2 Divisão do Trabalho

O presente trabalho está organizado da forma a seguir. No capítulo 2 é apresentado um breve histórico sobre banco de dados temporais com conceitos necessários para o entendimento dos próximos capítulos. No capítulo 3 é apresentada uma introdução ao *Time Series Cartridge*, descrevendo conceitos, arquitetura, modelo de série de tempo e limitações. No capítulo 4 é apresentado o Pacote de Tempo de Validade. Na medida do possível é feita uma analogia com o TSC. Para finalizar o capítulo é apresentada a metodologia de trabalho dos dois pacotes. No capítulo 5 são descritas metodologias para a inserção e o gerenciamento do tempo de validade, bem como especificadas as regras que são implementadas para manter a consistência no banco de dados. No capítulo 6 são apresentadas as implementações realizadas. No capítulo 7 é realizado um estudo de caso utilizando o pacote de tempo de validade e o TSC. E, para finalizar, no capítulo 8 são apresentadas conclusões e perspectivas de trabalhos futuros.

2 Banco de Dados Temporais

Diferentemente de BDs convencionais onde a realidade é representada apenas pelo estado presente de um objeto, os bancos de dados temporais permitem armazenar e recuperar todos os estados do objeto ao longo do tempo [ÖZS 95, TAN 93, EDE 94, ZAN 97].

O processo de modelagem de dados na construção de sistemas nada mais é do que o mapeamento de informações do mundo real para um ambiente computacional. Normalmente em aplicações do mundo real estão presentes informações temporais relevantes à especificação dos sistemas, informações estas que tiveram, têm ou terão seus valores alterados. Durante a evolução da aplicação bancos de dados temporais são usados para armazenar aspectos temporais de informações do mundo real, apresentando alguma forma de representação de informações temporais [EDE 94].

2.1 Representação de Informações Temporais

Informações temporais são representadas em BDs através de datas, períodos, intervalos temporais e duração da validade de informações. A associação entre as informações temporais e os dados, ou simplesmente a associação de tempo a um dado, é possível através da inserção de uma dimensão temporal no banco de dados. Com esta dimensão é possível identificar quando uma informação foi definida e qual o tempo em que ela é válida [EDE 98]. Esta dimensão temporal associa um tempo a um atributo de forma que, se o valor do atributo for alterado, o valor anterior não seja perdido.

Conforme [EDE 98], a dimensão temporal é composta por uma seqüência de pontos consecutivos no tempo, que recebe o nome de *eixo temporal*. A maneira mais simples de armazenar informações temporais é através da associação de um valor temporal a uma *tupla* ou dado. Este valor temporal ou rótulo temporal (*timestamp*) é manipulado pelo SGBD. Desta forma os valores definidos antes da atualização do valor não são perdidos ficando armazenados no banco de dados.

2.2 Formas de Armazenamento de Informações Temporais

Os bancos de dados temporais podem ser classificados conforme a forma com que armazenam valores temporais. Em [JEN 98] foram classificados 3 tipos de banco de dados temporais:

- **banco de dados de tempo de transação** - é caracterizado por usar o tempo de transação como o rótulo temporal. O rótulo temporal representa o exato momento em que a informação foi inserida ou alterada no BD;

- **banco de dados de tempo de validade** - caracteriza-se por associar o tempo de validade a cada informação no BD. O tempo de validade expressa o tempo em que a informação é válida no banco de dados. A validade da informação pode ter início em outro instante de tempo diferentemente do tempo de transação. O tempo de validade é uma informação fornecida pelo usuário do banco de dados;

- **banco de dados bitemporal** - em muitos casos a representação do mundo real só é obtida através da utilização em conjunto do tempo de transação com o tempo de validade. O banco de dados bitemporal armazena informações de tempo de transação e de tempo de validade, possibilitando o acesso a toda história do banco de dados, inclusive a dados futuros.

3 Time Series Cartridge

Time Series Cartridge ou Cartucho de Séries de Tempo é um pacote de tempo inserido na versão 8 do SGBD Oracle, para possibilitar o armazenamento e a recuperação de dados temporais [ORA 99]. É formado de um conjunto de procedimentos, funções e objetos com características próprias, adaptadas ao modelo relacional. Uma série de tempo é o montante de dados de entrada referenciados através de rótulos temporais (*timestamps)*. Cada atributo ou coluna em uma tabela tem associado um rótulo temporal, ou seja, uma marca que situa um dado ou objeto no tempo.

Muitas das funcionalidades oferecidas pelo TSC atualmente são implementadas pelos usuários utilizando os recursos disponíveis nos SGBDs relacionais, como os gatilhos, procedimentos e restrições sobre atributos, em conjunto com funções e tipos de dados tradicionais.

3.1 Estrutura do Banco de Dados de Séries de Tempo

A arquitetura do modelo para séries de tempo inserido no Oracle8i consiste em 3 níveis, o de armazenamento de informações de séries de tempo, o lógico ou de pacotes de códigos em PL/SQL, e o de interface de ligação:

- **o nível de armazenamento** é estruturado através de uma tabela de organização de índices (IOT – *index organized table*) associado a um objeto. Esta tabela pode ser do tipo *FLAT table* onde o rótulo temporal está associado a todos os atributos da tupla, ou *NESTED table*, no qual pode existir um rótulo temporal para cada atributo da tupla;
- **o nível lógico** consiste em pacotes de códigos em PL/SQL que determinam os calendários, as séries de tempo, as funções de tempo e os procedimentos que permitem realizar operações administrativas sobre um esquema de série de tempo;
- **a interface de ligação** é o meio de ligação entre o nível lógico e o nível de armazenamento. Pode-se dizer que a interface consiste na manipulação de objetos da classe *ORDSYS.*

A estrutura do banco de dados para séries de tempo consiste de: um mecanismo de controle de integridade obtido através da criação automática de um gatilho acionado quando operações de inserção, exclusão e atualização são efetuadas sobre uma visão relacional; e de um conjunto de tabelas e visões responsáveis pelo armazenamento e pela consistência dos dados de séries tempo.

No momento da criação de uma tabela temporal, o cartucho de tempo cria automaticamente um esquema de banco de dados, formado por três tabelas e duas visões. Para uma tabela com o nome *funcionário* o seguinte esquema seria criado:

- **visão *funcionario*** - é um objeto sobre o qual serão manipulados os dados. Pode ser usada somente para acessos de leitura usando as funções de tempo do TSC;
- **visão *funcionario_rvw*** - é uma visão relacional responsável pela integridade dos dados onde operações de inserção, modificação e remoção são aplicadas e controladas;
- **tabela *funcionario_tab*** - é a tabela que contém os dados (tuplas) propriamente ditos;
- **tabela *funcionario_map*** - armazena o identificador da tupla, que neste caso pode ser o código do funcionário;
- **tabela *funcionario_cal*** - armazena informações sobre os padrões do calendário, se este existir (maiores detalhes na seção 3.2).

O TSC fornece recursos para que a estrutura temporal possa ser inserida em um BD existente. Para isso é necessário que sejam especificadas as tabelas que pertencerão à série de tempo e o atributo de identificação. O atributo de identificação é necessário pois, junto com o rótulo temporal, formará uma chave composta. No caso do esquema de BD para cadastro de funcionários o atributo de identificação poderia ser o CPF ou o código do funcionário.

É comum a utilização do TSC para situações onde o BD já existe, ou mesmo para partes de um BD. A estrutura temporal pode ser inserida no BD para a recuperação de informações mais precisas e mais rápidas, através dos procedimentos e funções existentes.

A figura 3.1 mostra o esquema de banco de dados de séries de tempo criada pelo TSC para a tabela *funcionário*.

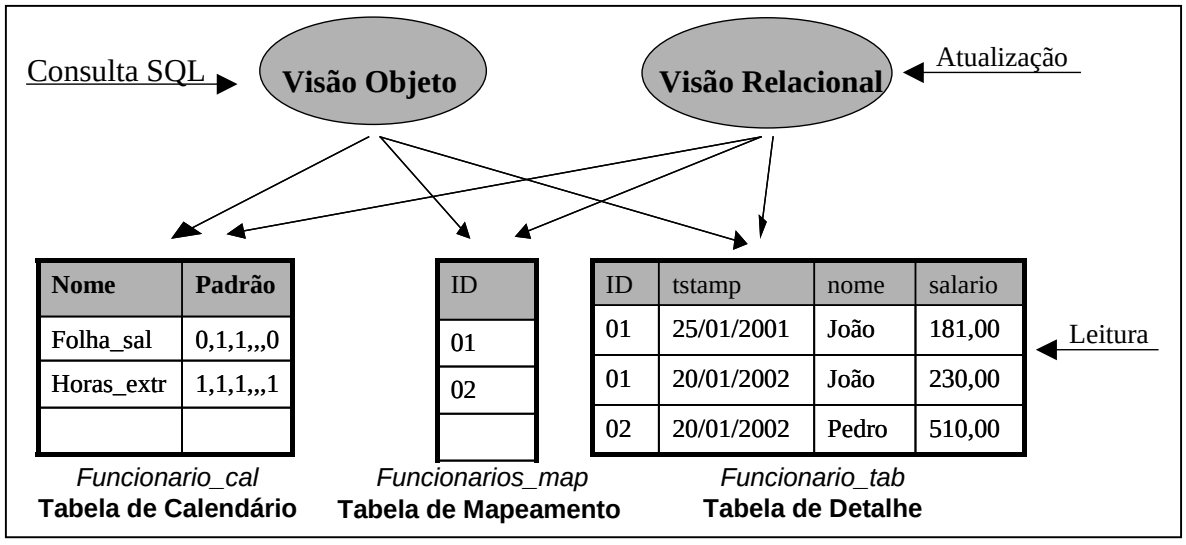


FIGURA 3.1 - Estruturação de informações temporais [ORA 99].

3.2 Modelos de Séries de Tempo

O TSC utiliza conceitos próprios para tratar informações temporais, como a utilização ou não de *calendários.* Aplicações de banco de dados temporais basicamente são implementadas de duas formas, com o uso de calendários ou sem calendários. Calendários são objetos do TSC que permitem vincular as informações a datas ou a períodos de tempo definidos [ORA 99]. Existem dois modelos para tratar dados de séries de tempo: o modelo de séries de tempo irregular e o regular. No modelo irregular os dados da série de tempo não necessariamente estão associados ao uso de calendários, enquanto que no modelo regular obrigatoriamente devem estar associados.

A definição da utilização de um ou outro modelo de série de tempo varia com a especificação da modelagem temporal. Calendários geralmente não são necessários quando:

1. o rótulo temporal não tiver um padrão;
2. o rótulo temporal tiver um padrão, mas este não for relevante para a recuperação dos dados;
3. o rótulo temporal tiver um padrão, mas este não estiver correto.

Toda a série de tempo regular é associada ao uso de um calendário. Utilizam-se calendários quando for conhecido o momento da inserção dos dados, através de intervalos de tempos pré-definidos associados a um rótulo temporal padrão. Os calendários são utilizados também para validar a inserção de dados, definindo períodos, datas limites e datas não válidas para a inserção, como feriados, por exemplo.

A tabela de calendários é criada automaticamente quando a estrutura de tempo é inserida no BD. A validação de um calendário é obtida através da inserção de um padrão na tabela de calendários.

Conforme exemplificado na figura 3.2, para definir um padrão de séries de tempo no BD são necessários:

- especificar um objeto do tipo calendário, conforme mostrado nas linhas 1 e 2;
- um nome para o calendário, conforme a linha 4;
- a definição da granularidade, mostrada na linha 5;
- padrões e exceções, relacionados nas linhas 6 a 10 .

Maiores detalhes sobre a definição e uso de calendários podem ser vistos em [PIN 01].

A figura 3.2 mostra a inserção de um padrão em um calendário da tabela *funcionários*. Após a inserção de um calendário um objeto é criado, como mostra a tabela *funcionario_cal* da figura 3.1.

```
1. INSERT INTO funcionario_cal VALUES (
2.    ORDSYS.ORDTCalendar(
3.    0,                        -- tipo do Calendario (0 = Standard)
4.    'folha_salarial',         -- nome do Calendário
5.    4,                        -- freqüência para o Dia
6.    ORDSYS.ORDTPattern (    -- definição do Padrão (exigido)
7.          ORDSYS.ORDTPatternBits(0,1,1,1,1,1,0),
8.                TO_DATE('15/08/2001','DD/MM/YYYY') ),
9. '01/01/2001','01/01/2003', -- rótulos temporais limitantes
10. NULL, NULL ); -- Semfinal e exceções
```

FIGURA 3.2 - Exemplo de criação de calendário.

3.3 Consistência de Dados de Séries de Tempo

Quando se está trabalhando com séries de tempo e funções de escalonamento de tempo, pressupõe-se que os calendários estão consistentes com os dados de séries de tempo. Quando isso acontece, pode-se utilizar o calendário como uma base para a utilização de funções de tempo e a recuperação de dados. Para que uma série de tempo seja consistente é necessário que as seguintes condições sejam verdadeiras [ORA 99]:

- todos os rótulos temporais devem ser ordenados em ordem crescente;
- não existam rótulos temporais duplicados;
- todos os rótulos temporais tenham a mesma freqüência definida no calendário;
- não exista rótulo temporal que não esteja compreendido entre o menor e o maior rótulo temporal permitidos conforme definidos no calendário;
- os dados da série de tempo sejam contíguos, ou seja, entre o maior e o menor valor do rótulo temporal deve haver dados para todos os valores válidos para rótulos temporais, mesmo que sejam nulos.

A figura 3.3 "*A*" mostra uma série de tempo consistente, enquanto a figura 3.3 "*B*" mostra uma série de tempo inconsistente. Esta inconsistência é verificada quando o valor do rótulo temporal inserido não esta compreendido entre o menor ou maior rótulo temporal definido no calendário. Conforme mostra a figura 3.2 os valores mínimo e máximo definidos são "`01/01/2000`" e "`01/01/2003`".

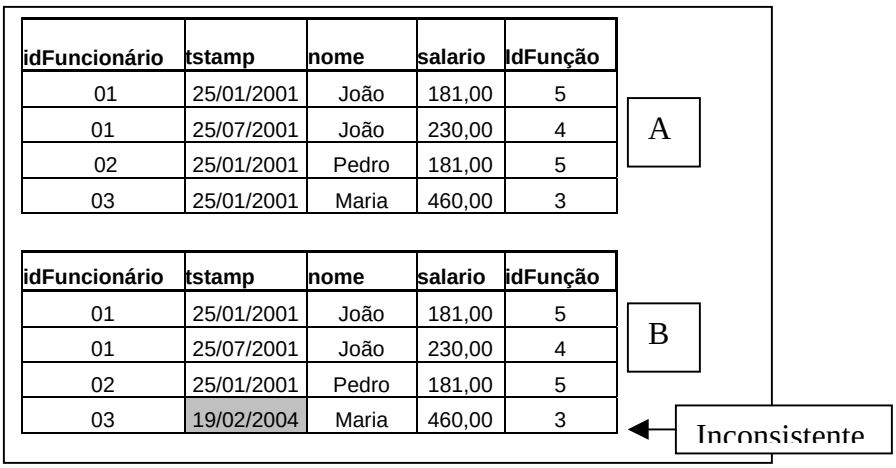


FIGURA 3.3 - Séries de tempo consistente(A) e inconsistente(B).

O SGBD Oracle fornece formas de forçar a consistência de dados de séries de tempo em visões relacionais (*relational views)* usando um "*INSTEAD OF TRIGGER"*. Apesar deste recurso não fazer parte do TSC, o uso possibilita limitar ou moderar

operações de inserção, atualização e remoção sem a necessidade de mudanças dos dados da série de tempo.

Ao criar uma tabela temporal, o TSC automaticamente cria uma *INSTEAD OF TRIGGER*. Este procedimento garante que:

1. operações de inserção (*INSERT)*, para uma série de tempo vazia, o novo rótulo temporal seja um valor válido para o calendário. Para uma série de tempo que não está vazia, a inserção ocorrerá imediatamente depois do último rótulo temporal;

2. operações de remoção *(DELETE)*, para uma série de tempo vazia, uma exceção é executada. Para uma série de tempo que não está vazia, somente podem ser excluídos dados do primeiro e do último rótulo temporal;

3. operações de atualização *(UPDATE)*, um rótulo temporal deve existir na série de tempo. Não são permitidas atualizações do rótulo temporal e da coluna de identificação da tupla em uma visão relacional.

3.4 Arquitetura

O TSC nada mais é do que um pacote de procedimentos desenvolvidos em PL/SQL. O pacote consiste da criação de duas visões, sobre as quais são aplicados mecanismos de recuperação e inserção dos dados. Ao inserir características temporais através do TSC são criadas: uma visão objeto que fornece recursos para consulta e administração dos objetos; e uma visão relacional responsável pela atualização dos dados e administração de entrada de rótulos temporais. O BD é estruturado segundo especificação do usuário, através de *Flat Tables* ou *Nested Tables,* detalhadas na seção 3.1. A arquitetura do TSC é mostrada na figura 3.4.

O pacote de códigos em PL/SQL do TSC é composto de um gatilho que controla o acesso, a atualização e a inserção de dados nas visões. Fazem parte também do pacote funções de recuperação e otimização de informações relacionadas ao rótulo temporal.

Conforme mostra a figura 3.4, é através da visão objeto que são realizadas consultas ao calendário, se este foi definido, bem como às séries de tempo, ou seja, de um montante de informações relacionadas ao rótulo temporal, e das funções para otimização de consultas, denominadas na figura como ferramentas.

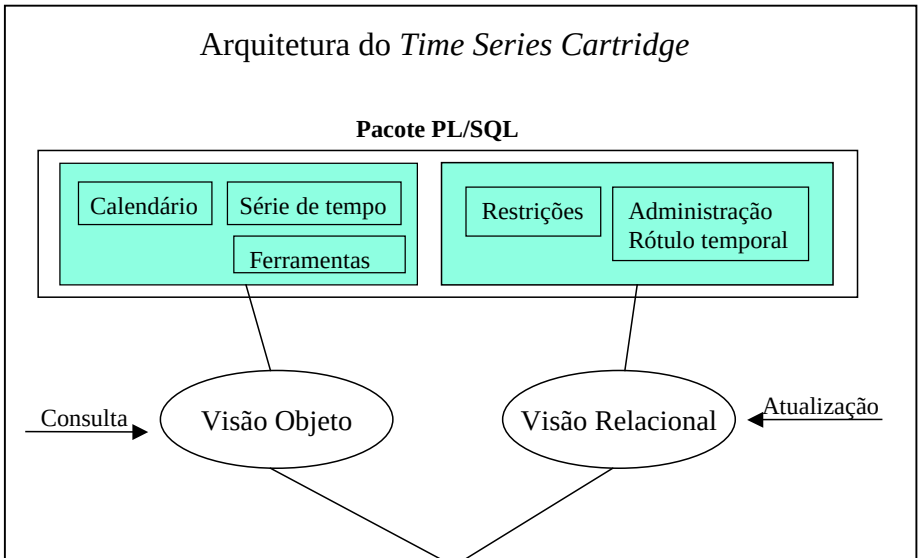


FIGURA 3.4 – Arquitetura do Time Series Cartridge (adaptado de [ORA 99]).

É através da visão relacional que a consistência do BD é garantida. Nesta visão é feita a administração do tempo de transação através da execução das regras que são aplicadas ao rótulo temporal, bem como são executadas restrições que foram definidas no calendário.

3.5 Limitações

O TSC fornece recursos para que novas funções e procedimentos sejam criados ou estendidos de forma a melhor se adequarem aos requisitos de cada aplicação. É possível, por exemplo, criar funções que operem sobre qualquer atributo de uma tabela associada, não apenas do atributo de rótulo temporal (*timestamp*), como ocorre na maioria das funções oferecidas.

Entretanto, ao inserir o tempo em um banco de dados, o TSC restringe os tipos de dados que podem ser operados. Uma série de tempo pode conter apenas atributos do tipo número (*NUMBER*) e do tipo seqüência de caracteres (*VARCHAR*). A única exceção é o atributo do rótulo temporal que é do tipo data (*DATE*). Esta limitação reduz o campo de aplicação do TSC, bem como da extensão dos seus recursos.

4 Pacote de Tempo de Validade (PTV)

Neste capítulo são apresentadas as propostas de inserção de tempo de validade no SGBD Oracle. Da mesma forma que no TSC, as ferramentas e o controle de administração do tempo de validade sugeridos e implementados neste trabalho são inseridos em um pacote, que foi denominado PTV (Pacote de Tempo de Validade). O PTV consiste de funções, procedimentos e gatilhos que fornecem recursos para a inserção e a administração do tempo de validade, bem como recursos para facilitar a recuperação de informações temporais.

Neste capítulo é apresentado o Pacote de Tempo de Validade (PTV). É descrita a estrutura do pacote, bem como é mostrada e exemplificada a forma como o tempo de validade é representado. É mostrada também a reação do PTV a situações cotidianas que ocorrem sobre bancos de dados. É apresentada a arquitetura do PTV fazendo uma analogia à arquitetura do TSC. Para encerrar o capítulo, é apresentada a metodologia de execução do PTV e do TSC. Na medida do possível, este capítulo apresenta o PTV fazendo sempre uma analogia a conceitos e à estrutura do TSC. O objetivo principal deste capítulo é apresentar as características gerais do pacote implementado, que será mais detalhado nos capítulos a seguir.

4.1 Representação do Tempo de Validade

Em banco de dados de tempo de transação, o tempo pode ser representado através de um único valor (um ponto no tempo) ou através de um intervalo temporal representando o início e o fim da "vida" da informação. As mesmas formas de representação podem ser usadas para representar a validade da informação em BD de tempo de validade. No tempo de validade a informação temporal pode conter apenas um valor identificando o passado, presente ou futuro, ou ter como valor um intervalo temporal identificando o período pelo qual a informação foi, é ou será válida no mundo real.

Diferentemente do TSC que administra o tempo de transação através de um único ponto no tempo, o PTV administra o tempo de validade através de um intervalo temporal, identificando a validade inicial e final da informação. Desta forma, foram definidos dois atributos para cada informação, o atributo *TvalidadeI* que representa o tempo de validade inicial e o *TvalidadeF* representando o tempo de validade final. A figura 4.1 mostra um

BDT onde estão sendo executados o PTV e o TSC. A figura 4.1 “A” mostra uma tabela atemporal e a “B” mostra a mesma tabela após a execução dos pacotes. O atributo *Tstamp* corresponde ao tempo de transação necessário para a execução do TSC. Mesmo podendo ser executado em conjunto com o TSC, o PTV é independente, podendo ser executado isoladamente.

cpf	IdFunção	nome	salario	Data
66062244074	5	João	381,00	25/01/2001
66062244074	4	João	430,00	25/07/2001
70062244070	5	Pedro	381,00	27/01/2001
81162244071	3	Maria	200,00	25/01/2001

A

id	tstamp	TvalidadeI	TvalidadeF	Nome	salario	idFunção
66062244074	25/01/2001	27/01/2001	24/07/2001	João	381,00	5
66062244074	25/07/2001	25/07/2001	Null	João	430,00	4
70062244070	25/01/2001	27/01/2001	Null	Pedro	381,00	5
81162244071	25/07/2001	25/07/2001	25/08/2001	Maria	200,00	3

B

FIGURA 4.1 – Exemplo de tabela de BD após a execução do TSC e do PTV.

4.2 Arquitetura

Apesar de utilizar os mesmos recursos que foram utilizados para o desenvolvimento do TSC, a arquitetura para inserção e administração do tempo em BDs do PTV difere significativamente da arquitetura do TSC. O motivo principal desta diferença é a quantidade de recursos disponíveis para manipulação de informações no BD.

O PTV tem como objetivo principal a preocupação com o controle de consistência do BDT, fornecendo mecanismos que são executados durante operações de inserção, alteração e remoção de informações temporais. Já o TSC tem como objetivo principal fornecer recursos para a otimização de consultas sobre o tempo de transação, mesmo porque, a complexidade do controle de consistência em BDT sobre o tempo de transação não é tão grande quanto a do tempo de validade. Além disso, o TSC fornece mecanismos para que um rótulo temporal seja associado a um único dado em uma tupla através da utilização de *FLAT Tables*, ou associar rótulos temporais para cada dado da tupla através da estrutura de *NESTED Tables*, conforme descrito na seção 3.1. Este recurso não é fornecido pelo PTV, que associa o rótulo temporal a um único dado da tupla.

Para um melhor entendimento, a seguir é feita uma analogia entre as arquiteturas do TSC e do PTV.

4.2.1 Arquitetura do PTV

Diferentemente do TSC que utiliza visões para o acesso ao BD, o PTV é aplicado diretamente sobre as tabelas. Basicamente o PTV é composto de:

- ***mecanismos de consistência*** - associados a cada tabela, são responsáveis pelo controle de entrada de dados no BD de forma a manter a consistência através da execução de regras para o tempo de validade detalhadas no capítulo 5;
- ***mecanismos de controle*** **–** responsáveis pela ativação, inserção e desativação do tempo de validade no banco de dados;
- ***ferramentas de consulta*** **–** têm como objetivo facilitar a recuperação de informações temporais.

A arquitetura do pacote de tempo de validade é mostrada na figura 4.2.

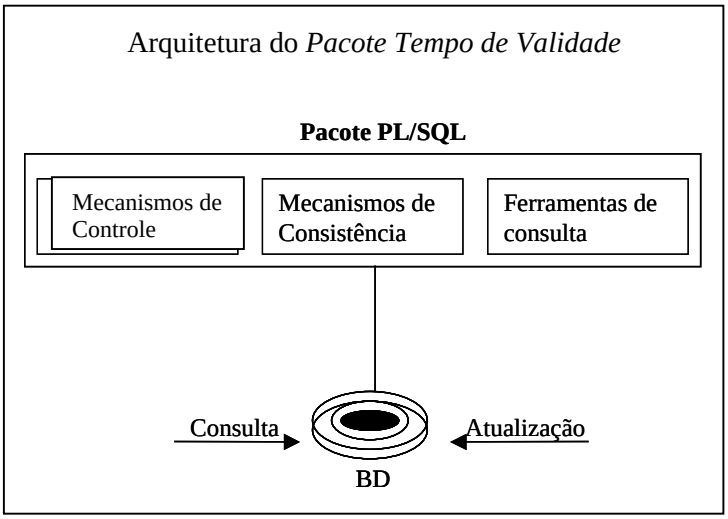


FIGURA 4.2 – Arquitetura do Pacote de Tempo de Validade (PTV)

Os mecanismos de controle são procedimentos executados pelo usuário a fim de criar o ambiente que possibilite a administração do tempo de validade, inserindo os atributos necessários no BD. É também através de procedimentos do mecanismo de controle (que são executados pelo usuário) que é acionado o mecanismo de consistência sobre o banco de dados. O mecanismo de consistência é composto por gatilhos associados ao BDT, garantindo a consistência através execução das regras descritas na seção 5.3. O modulo de ferramentas de consulta é composto de uma função chamada *PERIODO* que é utilizada em conjunto com a expressão *SELECT* do *SQL*. Apesar ser um protótipo de uma ferramenta com o objetivo de otimizar a realização de consultas de informações temporais, a função auxilia na exemplificação e no funcionamento do PTV. O modulo de ferramentas de consulta está aberto a futuras extensões.

4.3 Reação do PTV à Inserção de Dados e Inconsistências no BDT

Após a execução do PTV, as tabelas associadas passam a conter uma chave composta, aplicada sobre o atributo de identificação (*ID*) e o atributo de tempo de validade inicial (*TvalidadeI*). Desta forma é evitado que haja informações redundantes ou até mesmo inconsistentes. Também é criado um gatilho a fim de não permitir que operações realizadas sobre o SGBD possam gerar alguma inconsistência temporal.

Em alguns casos, procedimentos são tomados sem a percepção do usuário. Pode-se citar como exemplo a tentativa de inserção do valor nulo para o tempo de validade inicial (*tvi*). Em BDTs o valor de *tvi* não pode ser nulo. Se esta operação for realizada, o PTV insere como *default* o valor do rótulo temporal de tempo de transação (*tt*). Outro procedimento que não necessita acesso ao BDT, é a verificação dos valores de *tvi* e do tempo de validade final (*tvf*). O PTV não permite que o valor de *tvi* seja maior que o valor de *tvf*.

Na maior parte das vezes, para tratar as operações executadas sobre o BDT é necessária a recuperação de alguns dados da base de dados. Estes dados são necessários para que sejam confrontados com os dados submetidos pelas operações do usuário. Desta forma, é evitado que operações como a da inserção de um valor para *tvi* seja menor ou igual ao valor de *tvf* de uma tupla já existente. A figura 4.3 mostra esta situação.

Maiores detalhes sobre a reação do PTV a operações submetidas ao BDT são descritos na seção 5.3.2.

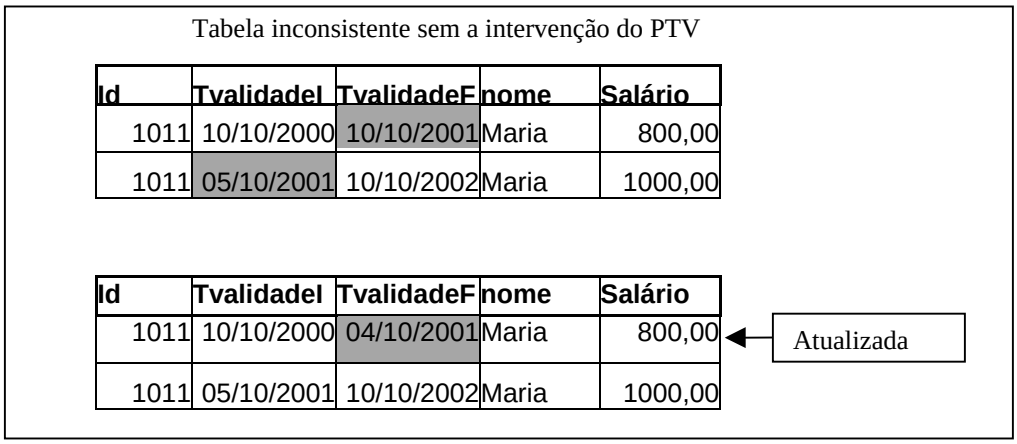


FIGURA 4.3 – Reação do PTV a operações que geram inconsistência

4.4 Metodologia de Trabalho

Apesar de ter como base algumas características do TSC, o PTV é inserido no BD de forma independente, podendo ser executado sozinho ou em paralelo. Através do TSC é possível criar um BD ou inserir características temporais em um existente, mesmo que este esteja populado. Já o PTV, não cria um banco de dados, apenas insere atributos temporais sobre um BD existente. Entretanto, quando o PTV for executado sobre um BD populado é necessária a execução do procedimento *verConsistenciaTV*, que verifica se existem inconsistências no rótulo temporal, procedimento este apresentado na seção 4.2.1. A figura 4.4 apresenta uma forma de iniciar o TSC sobre um BD existente, e a figura 4.5 apresentada na seção 4.4.1 mostra como inserir o tempo de validade através do PTV.

```
DECLARE -- criacao do esquema de série de tempo
BEGIN
  DBMS_OUTPUT.Enable(100000);
  ordsys.tstools.trace_on;
 ORDSYS.TSTools.Begin_Create_Ts_Group('vendas_ts','flat');
  ordsys.tstools.set_flat_attributes(
        detail_table_name    => 'vendas',
```

FIGURA 4.4 – Inserção do TSC em um BD existente

A figura 4.4 mostra a criação de um esquema de séries de tempo denominada *vendas_ts* do tipo *flat table*. No exemplo são definidas as tabelas do BD requisitadas pelo esquema do TSC. Como descrito na seção 3.1 são necessárias: a tabela de detalhes "*tab*"; a tabela de metadados "*map*" e a tabela de calendário "*cal*". Se no BD atual existirem as tabelas requisitadas pelo TSC, no instante da criação do esquema de tempo as tabelas deverão ser especificadas através do valor "*1"*, conforme o exemplo da figura 4.4, e não existentes através do valor "*0"*. Todas as tabelas declaradas como não existentes serão criadas automaticamente pelo TSC. No exemplo da figura 4.4 a tabela *vendas_calendario* é criada, e as demais são adaptadas. Para completar, o atributo *filial1* é inserido na tabela de detalhes.

4.4.1 Ativando o Pacote de Tempo de Validade – PTV

```
-- inserção do tempo de validade no BD

-- insere atributos de tempo de validade
EXECUTE pTV.InsereAtributoTV(vendas);

-- ativa o controle de tempo de validade
EXECUTE pTV.AtivaTV(vendas);
```

FIGURA 4.5 – Inserção do tempo de validade através do PTV.

A figura 4.5 mostra a inserção do tempo de validade através do PTV. Para o processo de inserção dos atributos de tempo de validade e da ativação do mecanismo de controle de consistência temporal foram implementados dois procedimentos. Estes procedimentos podem ser executados sobre o BD a qualquer instante. Conforme exemplificado na figura 4.5, primeiramente são inseridos na tabela vendas os atributos de tempo de validade e um identificador, através do procedimento *InsereAtributoTV*. Este procedimento cria o ambiente necessário para que o tempo de validade possa ser inserido e administrado. Depois da inserção dos atributos de tempo de validade, é possível ativar o mecanismo de controle de tempo, utilizando o procedimento *ativaTV*. Este procedimento

ativa um gatilho associado à tabela especificada, controlando operações de inserção, modificação e remoção dos dados.

Foi implementado e inserido no PTV um procedimento para desativar o mecanismo de controle do tempo de validade. O processo de desativação pode ser executado em qualquer instante através da execução do procedimento "*desativaTV*". Os procedimentos de inserção, ativação e desativação do tempo de validade no BD são mostrados com maiores detalhes no capítulo 6.

4.4.2 Procedimentos para Verificação de inconsistências

Tanto o TSC quanto o PTV, podem ser inseridos em um BD que já esteja populado. Este procedimento requer que o BD esteja consistente, conforme as regras estabelecidas para cada pacote. As regras são mostradas nos capítulos 3 para o TSC e no próximo para o PTV.

O TSC fornece recursos para verificar se o BD está consistente segundo as especificações do calendário. Apesar de acusar a existência de uma inconsistência através da execução de uma exceção no momento da definição do calendário, o TSC não fornece recursos para tratar ou até mesmo corrigi-la.

Da mesma forma que o TSC, o PTV fornece um mecanismo para a verificação de inconsistências, através da execução do procedimento "*verConsistenciaTV*". Este procedimento varre o BD a procura de inconsistências, com base nas regras descritas no capítulo 5. Na medida do possível as inconsistências são corrigidas automaticamente pelo PTV. Entretanto, pode haver situações críticas onde o PTV não pode tomar decisões sem a intervenção do usuário. Nestas situações, são impressos os pontos críticos para que o usuário faça as correções manualmente. Também são verificados por este procedimento se os rótulos temporais estão presentes.

Ao inserir características temporais em um BD existente através do TSC ou do PTV, é necessário que os rótulos temporais estejam consistentes. De outra forma, os procedimentos não terão os resultados esperados.

5 Inserção e Gerência do Tempo de Validade no SGBD Oracle

Neste capítulo são apresentadas as dificuldades da implementação do tempo de validade, bem como os recursos e limitações do SGBD. No decorrer do capítulo, é feita uma analogia entre as características do TSC e das propostas para a implementação do pacote de tempo de validade. São apresentadas, também, as regras para consistência do BDT e as soluções encontradas para a implementação.

5.1 Recursos do SGBD

O SGBD Oracle permite que o TSC seja expandido através da criação e da alteração de funções e procedimentos que estão relacionados às informações temporais. Apesar do TSC conter uma estrutura de BD própria, o controle e as funcionalidades são fornecidos pelo SGBD relacional. Desta forma, como o SGBD não fornece suporte a operações sobre dados temporais, o TSC utiliza recursos como os *triggers* (gatilhos) para manter a consistência dos dados de tempo em visões relacionais.

Da mesma forma como no TSC, para a implementação do tempo de validade este trabalho utiliza recursos fornecidos pelo SGBD Oracle, tais como gatilhos, funções, procedimentos e restrições sobre atributos. O tempo de validade não é inserido sobre a estrutura criada pelo TSC, podendo o mecanismo de administração do tempo de validade ser executado em paralelo ou de forma independente.

Um sistema gerenciador de banco de dados temporais (SGBDT) é responsável por manter a consistência do BDT, fornecer mecanismos de armazenamento e recursos para operações de inserção, exclusão e atualização de forma rápida e segura. Estas operações sobre dados temporais demandam muito tempo de processamento, o que pode comprometer o desempenho e a confiabilidade dos BDTs. Este processo pode piorar se os recursos forem administrados através de gatilhos, funções e procedimentos.

Em [HÜB 2000,HÜB 2000a] são analisados procedimentos para inserir informações temporais sobre BD convencionais, utilizando os conceitos aplicados a BDTs. Não é medido o tempo de processamento para garantir a consistência e o tempo de resposta na execução de operações sobre os dados. Presume-se que a utilização dos conceitos de BDTs como um todo, sobre BD convencionais, utilizando recursos como gatilhos e funções, degradariam sensivelmente o desempenho de acesso às informações do BD.

Deste modo, esta seção sugere mecanismos de inserção e administração de informações temporais, levando em consideração a performance do acesso e da manipulação dos dados no BD. As características de BDTs são preservadas. Entretanto, algumas funcionalidades não serão permitidas por imposição das regras de consistência, integridade e gerenciamento implementadas, bem como das limitações impostas pelo SGBD Oracle.

5.2 Consistência sobre Atributos de Tempo de Validade

A principal diferença entre os bancos de dados não temporais e os temporais é a maneira como o SGBD controla o acesso e a manipulação dos dados. Em BDTs nenhuma informação ou dado é excluído ou sobrescrito fisicamente. O que ocorre é a inserção de um novo valor que substituirá o antigo.

5.2.1 Operações Físicas de Exclusão e Atualização

O TSC permite operações físicas de atualização sobre atributos temporais, bem como operações de exclusão. Para garantir a consistência dos dados, o TSC não permite estas operações sobre o atributo de identificação da tupla, bem como do rótulo temporal, que juntos formam uma chave composta. Da mesma forma como ocorre no TSC, a operação de exclusão física no PTV, poderá ser realizada desde que não comprometa a consistência do BD.

Para manter a consistência do BD foram implementadas restrições para operações físicas de exclusão e atualização. Estas restrições são apresentadas na próxima seção.

5.2.2 *Vacuuming* ou Exclusão

Embora em BDTs não se deva alterar dados armazenados para preservar o histórico, em alguns casos é necessário excluir ou alterar dados. A operação de exclusão física das informações em BDTs é chamada de *vacuuming*, e é executada quando existem dados no BD que não são mais relevante para a aplicação. A operação de *vacuuming* deve ser executada somente pelo administrador do BD (DBA), sob sua inteira responsabilidade, não sendo permitida aos demais usuários. No pacote implementado (PTV) foi acrescentado o suporte a *vacuuming* ressaltando que somente o DBA deverá executa-lo. A seguir são analisadas as operações de exclusão sendo apresentadas as soluções.

A operação de *vacuuming* é permitida para qualquer tupla do BD. No entanto, este procedimento pode causar lacunas na base de dados se for realizado em tuplas que não sejam das extremidades (as extremidades são medidas pela ordem crescente do tempo de validade inicial), ou seja, intervalos entre o tempo de validade final (*tvf)* de uma tupla e o tempo de validade inicial (*tvi*) de outra tupla de mesmo valor, como é mostrado na figura 5.1. Em BDTs é permitido que haja lacunas para informações de tempo de validade, desde que sejam inseridas pelo usuário de forma explícita, e não por operação de exclusão física.

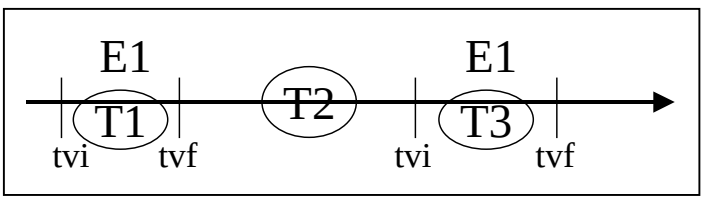


FIGURA 5.1 - Lacunas no tempo de validade

A figura 5.1 mostra uma linha de tempo onde o empregado E1 trabalhou na empresa no período T1, ficou fora da empresa durante o período T2, e retornou no período T3. Esta operação é válida em BDTs, desde que no período T2 não tenha existido uma tupla para o empregado E1.

5.2.3 Atualização

Existem duas formas de realizar operações de atualização em BD não temporais: através da operação de atualização (*UPDATE)* onde o valor anterior é perdido*;* e através de uma operação de inserção, de forma que o valor anterior da tupla não seja perdido. Em operações de inserção, a atualização é caracterizada no BDT se existir uma tupla com o mesmo ID.

Diferentemente da forma que ocorre em BDTs, onde a atualização física dos dados não temporais é permitida, no PTV não é permitida a atualização física sobre qualquer dado. Entende-se por operação física de atualização quando o dado anterior à operação de atualização deixa de existir no BD. O procedimento de proibir a atualização até mesmo dos atributos não temporais é resultado da deficiência de recursos do SGBD Oracle para a implementação deste procedimento.

O TSC não restringe qualquer atualização sobre os atributos temporais e não temporais. Entretanto, se ambos pacotes estiverem executando simultaneamente prevalecerá a proibição de operações de alteração física sobre quaisquer dados.

5.3 Regras de Consistência de Dados de Tempo de Validade

O TSC não é um BD de tempo de transação e sim um pacote que fornece recursos para a inserção do rótulo temporal similar ao de tempo de transação. No TSC algumas regras básicas de BDT são violadas, como por exemplo, a atualização e exclusão física dos dados.

O mecanismo implementado para a administração do tempo de validade no PTV, implementa regras que têm como objetivo principal manter a consistência do BDT preservando ao máximo suas características. Entretanto, como acontece no TSC, questões relativas à praticidade e facilidade de implementação de sistemas comerciais e à performance do acesso e manipulação das informações armazenadas são consideradas.

As regras de consistência sobre o tempo de validade levam em consideração a estrutura e as atuais regras de consistência fornecidas pelo TSC, mostradas na seção 3.3. As implementações das regras são mostradas no capítulo 6.

Os dados de tempo de validade, diferentemente dos dados de tempo de transação que são fornecidos automaticamente pelo SGBD, devem ser fornecidos pelo usuário. Deste modo, foram classificados dois tipos de regras para o tempo de validade, as regras de usuário e as de banco de dados.

A seguir são apresentadas as regras implementadas para gerenciar informações de tempo de validade, onde:

tstamp	- representa o rótulo temporal;
tvi	- representa o tempo de validade inicial;
tvf	- representa o tempo de validade final;
new	- representa o novo valor de tempo de validade da tupla;
old	- representa o valor já inserido do tempo de validade da tupla;
maxOld	- representa o maior valor de tempo de validade existente no BD;
minOld	- representa o menor valor de tempo de validade existente no BD;
tvfOldNext	- representa o tempo de validade final da próxima tupla ;
tviOldPrior	- representa o tempo de validade inicial da tupla anterior;

123 - (fonte vazada) representa os dados inconsistentes no BD;

123 - (fonte preta) representa os valores que sofreram alterações.

5.3.1 Regras de Usuários

São regras de validação aplicadas sobre os dados fornecidos pelo usuário, não havendo a necessidade de acesso ao BD. As regras podem ser controladas por recursos do SGBD.

Regra 1
tvi <> NULL
- O tempo de validade inicial de uma informação não pode ser nulo. Se este valor não for fornecido pelo usuário será assumido o valor do rótulo temporal do tempo de transação (*timestamping*).

Regra 2.
tvi < tvf
- O tempo de validade inicial de uma informação nunca poderá ser maior que o tempo de validade final.

Regra 3.
UPDATE (tvi e tvf)
- Não são permitidas operações de atualização sobre os atributos de tempo de validade.

5.3.2 Regras de Banco de dados

Além das regras de usuários apresentadas em 5.3.1 são aplicadas regras a partir da análise dos dados armazenados. O mecanismo para controle de consistência será acionado se existir uma tupla no banco de dados identificada pelo atributo "*ID*" que corresponda a uma nova tupla com o mesmo valor para "*ID*", ou seja, será definido um novo valor para um atributo. Em algumas situações o gerenciador realiza alterações automaticamente sobre valores fornecidos pelo usuário ou mesmo armazenados a fim de corrigir ou evitar

inconsistências. A seguir são apresentadas as regras de BD, exemplificadas através de tabelas que representam os dados relevantes a cada operação.

REGRA 1. *para TVI.NEW > TVI.MaxOLD*

Esta regra aplica-se quando o valor de *tvi* da tupla a ser inserida no BDT (nova tupla) é maior que o maior valor de *tvi* de uma tupla que já tenha sido inserida no BDT (tupla armazenada).

Regra 1.1
SE TVI.NEW > TVI.MaxOLD **E**
SE tvi.new < tvf.old
ENTÃO tvf.old := tvi.new – (uma unidade de tempo)

➢ Se o valor de *tvi* da nova tupla for menor do que o valor de *tvf* da tupla armazenada, uma inconsistência no BDT é verificada, como mostra a tabela 5.1, onde o BDT está inconsistente. Nesta ocasião uma operação de consulta requisitando o salário da funcionária Maria na data 07/10/2001, resultaria dois valores.

Para melhor visualizar os exemplos nas tabelas, presume-se que o BDT esteja organizado pela ordem crescente do tempo de validade inicial.

TABELA 5.1 – Regra 1.1 *tvi.new < tvf.old* (BDT inconsistente)

Id	tvi	tvf	nome	Salário	
1011	10/10/2000	10/10/2001	Maria	800,00	
1011	05/10/2001	10/10/2002	Maria	1000,00	Nova

A solução encontrada para manter a consistência no banco de dados foi alterar automaticamente o valor do *tvf* da tupla armazenada pelo valor de *tvi* da nova tupla, diminuído de uma unidade de tempo. Deste modo optou-se por diminuir o intervalo de validade da tupla. A unidade mínima é de um dia, como mostra a tabela 5.2. A tabela 5.2 mostra um BDT consistente a partir da aplicação da regra 1.1 sobre o exemplo da tabela 5.1.

TABELA 5.2 – Regra1.1 *tvi.new < tvf.old* (BDT consistente)

Id	Tvi	tvf	nome	Salário	
1011	10/10/2000	04/10/2001	Maria	800,00	
1011	05/10/2001	10/10/2002	Maria	1000,00	Nova

Regra 1.2
SE TVI.NEW > TVI.MaxOLD **E**

SE tvi.new = tvf.old
ENTÃO tvf.old := tvi.new – (uma unidade de tempo)

➢ Da mesma forma como na regra 1.1 uma inconsistência no BDT é verificada se o valor de *tvi* da nova tupla for igual ao valor de *tvf* da tupla armazenada. Para este caso a solução também é a mesma, conforme exemplificado nas tabelas 5.1 e 5.2.

Regra 1.3
SE TVI.NEW > TVI.MaxOLD **E**
SE tvf.old = NULL
ENTÃO tvf.old := tvi.new – (unidade temporal)

➢ O valor nulo para *tvf* significa que o tempo de validade final da tupla é indeterminado até que outra tupla seja inserida no banco de dados, ou até que o valor seja especificado pelo usuário. Neste caso, o valor NULL é substituído automaticamente pelo tempo de validade inicial da tupla a ser inserida, diminuída de uma unidade temporal. O processo é o mesmo aplicado para as regras 1.1 e 1.2. As tabelas 5.3 para o BDT inconsistente e 5.4 para o BDT consistente após a aplicação da regra, mostram esta situação.

TABELA 5.3 – Regra 1.3 *tvf = NULL* (BDT inconsistente).

id	Tvi	Tvf	Nome	salario	
1011	10/01/2001	19/09/2001	Maria	800,00	
1011	20/09/2001	10/10/2002	Maria	1000,00	Nova

TABELA 5.4 – Regra 1.3 *tvf = NULL* (BDT consistente).

id	tvi	tvf	Nome	salario	
1011	10/01/2001	NULL	Maria	800,00	
1011	20/09/2001	10/10/2002	Maria	1000,00	Nova

REGRA 2. *para TVI.NEW < TVI.MinOLD*

Esta regra aplica-se quando o valor de *tvi* da tupla a ser inserida no BDT é menor que o menor valor de *tvi* de uma tupla que já tenha sido inserida no BDT.

Regra 2.1
SE *TVI.NEW < TVI.MinOLD* **E**
SE (tvf.new > tvi.old) E (tvf.old + (uma unidade de tempo) > tvi.old)
ENTÃO tvi.old := tvf.new + (uma unidade de tempo)
SENÃO exceção

➢ Se o valor de *tvf* da nova tupla for maior que o valor de *tvi* da tupla armazenada, uma inconsistência é verificada, como mostra a tabela 5.5. Neste caso, mais de uma tupla teria intervalos de tempo em comum para informações diferentes. A solução para esta situação é

substituir o valor de *tvi* da tupla já armazenada pelo valor de *tvf* da nova tupla, acrescentado de uma unidade de tempo, que neste caso é de um dia. A tabela 5.6 mostra esta situação.

TABELA 5.5 – Regra 2.1 *tvf.new* > *tvi.old* (BDT inconsistente).

Id	Tvi	tvf	Nome	salario	
1011	13/01/2001	NULL	Maria	1000,00	
1011	10/01/2001	15/01/2001	Maria	800,00	Nova

TABELA 5.6 – Regra 2.1 *tvf.new* > *tvi.old* (BDT consistente).

Id	Tvi	Tvf	Nome	salario	
1011	16/01/2001	NULL	Maria	1000,00	
1011	10/01/2001	15/01/2001	Maria	800,00	Nova

Este procedimento pode ser aplicado se o novo valor inserido para o *tvi* for menor que o valor de *tvf* da mesma tupla, evitando assim a inconsistência verificada na regra de usuário de número 2.

Regra 2.2
SE *TVI.NEW* < *TVI.MinOLD* **E**
SE tvf.new = tvi.old
ENTÃO tvi.old := tvf.new + (uma unidade de tempo)

➢ Diferentemente da regra 2.1 que evita que mais de uma tupla tenha intervalos de tempo em comum para informações diferentes, esta regra evita que mais de uma tupla tenha uma mesma unidade temporal para informações diferentes. Desta forma, a solução para evitar a inconsistência do BDT pode ser a mesma aplicada para a regra 2.1, conforme visualizado nas tabelas 5.5 e 5.6.

Regra 2.3
SE *TVI.NEW* < *TVI.MinOLD* **E**
SE tvf.new = NULL
ENTÃO exceção

➢ Valor nulo para o *tvf* da nova tupla implica na invalidade do aspecto temporal das demais tuplas armazenadas no BDT. Deste modo, presume-se que a pretensão do usuário é a exclusão de todas as tuplas subseqüentes. Para este caso duas soluções foram encontradas: a exclusão automática das tuplas de mesmo ID e conseqüentemente a inserção da nova tupla; e a execução de uma exceção proibindo a operação.

Foi viabilizada a opção de execução de uma exceção, de maneira que se a intenção do usuário for a de exclusão, esta deveria ser feita através da operação de *vacuuming*

descrita na seção 5.2.2. No capítulo 6 são mostrados maiores detalhes sobre a implementação das regras de consistência.

A tabela 5.7 mostra o exemplo de um banco de dados inconsistente para a situação da regra 2.3.

TABELA 5.7 – Regra 2.3 *tvf = NULL*. (BDT inconsistente)

Id	Tvi	tvf	nome	salario	
1011	10/01/2001	15/05/2001	Maria	800,0	
1011	16/05/2001	25/12/2001	Maria	1000,0	
1011	05/01/2001	NULL	Maria	1200,0	Nova

.

REGRA 3. *para TVI.NEW = TVI.OLD (não extremo)*

Esta regra aplica-se quando o valor de *tvi* da tupla a ser inserida no BDT é igual ao valor de *tvi* de qualquer outra tupla que já tenha sido inserida no BDT, desde que este valor não seja o maior ou o menor valor de *tvi* da tupla de mesmo "ID" existente na tabela, ou seja, não esteja localizada nas extremidades da tabela. As extremidades são medidas pela ordem crescente do tempo de validade inicial. O conceito de extremidades é apenas explicativo, pois apesar da tabela ter como índice o valor de *tvi* o BD não tem uma organização seqüencial física. Entretanto, neste trabalho o controle sobre tuplas de extremidades e de não extremidades é diferenciado, conforme verificado analisando as regras 3 e 4.

Regra 3.1
SE *TVI.NEW = TVI.OLD (não extremo)* **E**
SE (tvi.new = tvi.old) E (tvf.new = tvf.old)
ENTÃO exceção

- Se o valor de *tvi* da nova tupla já existir para outra tupla no BDT e o valor de *tvf* da nova tupla coincidir com o valor de *tvf* da tupla armazenada, duas soluções são encontradas: a atualização dos dados ; ou a execução de uma exceção proibindo a operação.

Em BDTs não é permitida a atualização física dos dados. Bem como para BDTs, optou-se pela implementação da proibição da operação de atualização. Se a intenção do usuário é a de atualização das informações no BDT por não serem verídicas, por exemplo, a alternativa é realizar a exclusão física da informação (através do DBA) e realizar uma nova inserção. A tabela 5.8 mostra o BD inconsistente baseado nesta situação.

TABELA 5.8 – Regra 3.1 (*tvi.new = tvi.old*) E (*tvf.new = tvf.old*) – (Inconsistente)

Id	Tvi	tvf	Nome	salario	
1011	10/01/2001	15/05/2001	Maria	800,0	
1011	16/05/2001	25/12/2001	Maria	1000,0	
1011	26/12/2001	29/06/2002	Maria	1200,0	
1011	16/05/2001	25/12/2001	Maria	1200,0	N

Regra 3.2
SE *TVI.NEW = TVI.OLD (não extremo)* **E**
SE tvf.new > tvf.old
ENTÃO tvf.old := tvf.new
E tvi.oldNext := tvf.new + (uma unidade temporal)
OU
ENTÃO exceção

➢ Para a situação onde o valor de *tvf* da nova tupla é maior do que o valor *tvf* da tupla armazenada, duas soluções foram encontradas: a atualização do valor do *tvf* da tupla armazenada pelo valor do *tvf* da tupla nova e a atualização de todos os valores de tuplas armazenadas subseqüentes, cujos valores de *tvi* são menores aos valores de *tvf* da tupla nova, como mostra a tabela 5.9 ; ou a execução de uma exceção proibindo a operação.

Nesta situação é verificado um gasto excessivo de tempo de processamento para realizar a operação de atualização dos dados temporais, visto que o intervalo temporal de atualização pode ser muito grande ou até mesmo indefinido, bem como não é sabido o número de tuplas que serão atingidas. Por este motivo optou-se pela implementação da execução de uma exceção, proibindo esta operação.

No exemplo da tabela 5.9, o valor de *tvf* da nova tupla seria maior que o período de validade de duas tuplas subseqüentes que deveriam ser excluídas.

TABELA 5.9 – Regra 3.2 *tvf.new* > *tvf.old* (BDT inconsistente)

Id	Tvi	tvf	Nome	Salário	
1011	10/01/2001	15/05/2001	Maria	800,0	
1011	16/05/2001	10/07/2001	Maria	1100,0	
1011	11/07/2001	20/08/2001	Maria	1150,0	
1011	22/08/2001	25/01/2002	Maria	1200,0	
1011	16/05/2001	21/08/2001	Maria	1000,0	⊣ Nova

Regra 3.3
SE *TVI.NEW = TVI.OLD (não extremo)* **E**
SE tvf.new < tvf.old
ENTÃO tvf.old := tvf.new
OU ENTÃO exceção

➢ Para a situação em que o valor do *tvf* da nova tupla é menor do que o valor do *tvf* da tupla armazenada existem duas soluções: atualizar o valor do *tvf* da tupla armazenada com

o valor do *tvf* da nova tupla; ou não permitir esta operação. Ao permitir a operação de atualização do valor do *tvf*, um período de tempo fica sem valor especificado. Esta situação ocorre em BDTs, entretanto este intervalo não é proposital. A tabela 5.10 mostra um BDT inconsistente antes da aplicação desta regra baseando-se na atualização do valor de *tvf,* e a tabela 5.11 mostra um BDT consistente fisicamente, mas supostamente inconsistente logicamente, devido à possibilidade da ocorrência de uma situação em que o usuário de BD não tenha clareza sobre o resultado. Desta maneira optou-se pela implementação da execução de uma exceção.

TABELA 5.10 – Regra 3.3 *tvf.new* < *tvf.old* (BD inconsistente).

Id	Tvi	tvf	nome	salario	
1011	10/01/2001	15/06/2001	Maria	800,00	
1011	16/06/2001	30/11/2001	Maria	1000,00	
1011	01/12/2001	NULL	Maria	1400,00	
1011	16/06/2001	10/11/2001	Maria	1000,00	Nova

Após a atualização do valor do *tvf,* uma lacuna de tempo será verificada, ou seja, um período de tempo que em algum momento no banco de dados teve um valor associado e agora não tem mais. Na análise da tabela 5.10 é verificado que antes da atualização o período de tempo para a informação era contínuo, e após a atualização, como é mostrada na tabela 5.11, uma lacuna de tempo é encontrada, correspondente ao período de 11/11/2001 a 30/11/2001.

TABELA 5.11 – Regra 3.3 *tvf.new* < *tvf.old* (BDT supostamente consistente).

Id	Tvi	tvf	Nome	salario	
1011	10/01/2001	15/06/2001	Maria	800,0	
1011	16/06/2001	10/11/2001	Maria	1000,0	Atualizada
1011	01/12/2001	NULL	Maria	1400,0	

Regra 3.4
SE *TVI.NEW = TVI.OLD (não extremo)* **E**
SE tvf.new = NULL
ENTÃO exceção

➢ Para a situação em que uma tupla foi inserida com o valor de *tvf* igual a nulo e o valor de *tvi* da mesma tupla não for o maior já inserido no banco de dados, será necessária a exclusão de todas as tuplas subseqüentes à nova tupla.

Como para a situação da regra 3.2, é verificado um gasto excessivo de tempo de processamento para realizar a operação, visto que não é sabido o número de tuplas que serão atingidas. É verificado também que uma possível operação acidental do usuário ao inserir nulo para o valor de *tvf,* pode acarretar na exclusão física de tuplas do BDT. Por estes motivos optou-se pela implementação da execução de uma exceção, proibindo esta operação. Esta situação é mostrada na figura 5.12.

TABELA 5.12 – Regra 3.4 *tvf.new = NULL* (BDT inconsistente).

Id	Tvi	tvf	nome	salario	
1011	10/01/2001	15/06/2001	Maria	800,00	
1011	16/06/2001	30/11/2001	Maria	1000,00	
1011	01/12/2001	30/06/2002	Maria	1400,00	
1011	16/06/2001	NULL	Maria	1000,00	← Nova

REGRA 4. *para TVI.NEW <> TVI.OLD (não extremo)*

Esta regra aplica-se quando o valor de *tvi* de uma nova tupla não existe no BDT, e este valor não é um valor de extremidades, ou seja, o valor de *tvi* não é o maior ou o menor valor de *tvi* existente no BD.

Regra 4.1

SE *TVI.NEW <> TVI.OLD (não extremo)* **E**
SE tvf.new >= tvi.oldNext
SE tvf.new +1 <= tvf.oldNext
ENTÃO tvi.oldNext := tvf.new +1
SENÃO exceção

➢ A solução encontrada para a situação onde o valor de *tvf* da nova tupla é maior ou igual ao valor de *tvi* da próxima tupla, é a atualização do valor de *tvi* da tupla subseqüente, utilizando o valor de *tvf* da nova tupla acrescentado de uma unidade temporal. Entretanto, este procedimento somente poderá ser realizado se o valor de *tvf* da nova tupla não exceder o valor de *tvf* da tupla já armazenada. Tal procedimento necessitaria da atualização de um número de tuplas indeterminadas. Deste modo, optou-se pela execução de uma exceção. A tabela 5.13 mostra esta situação para um BDT inconsistente.

TABELA 5.13 – Regra 4.1 *tvf.new > tvi.oldNext* (BDT inconsistente).

Id	Tvi	tvf	Nome	Salário	
1011	10/01/2001	15/02/2001	Maria	800,0	
1011	25/02/2001	15/03/2001	Maria	1100,0	
1011	16/03/2001	15/04/2001	Maria	1150,0	
1011	16/02/2001	30/02/2001	Maria	1000,0	← Nova

Regra 4.2

SE *TVI.NEW <> TVI.OLD (não extremo)* **E**
SE tvi.new <= tvf.oldPrior

SE tvi.new -1 > tvi.oldPrior
ENTÃO tvf.oldPrior := tvi.new –1
SENÃO exceção

➢ Para a situação onde o valor de *tvi* da nova tupla é menor ou igual ao valor de *tvf* da tupla que a antecede, a solução encontrada foi diminuir de uma unidade temporal o valor de *tvf* da tupla antecessora. Para garantir a consistência é necessário que o valor de *tvf* não seja inferior ao valor de *tvi* da mesma tupla. Se isso ocorrer, a operação é abortada e uma exceção é executada. A tabela 5.14 mostra esta situação para um BDT inconsistente.

TABELA 5.14 – Regra 4.2 *tvi.new* <= *tvf.oldPrior* (BDT inconsistente).

d	Tvi	tvf	Nome	Salário
1011	10/01/2001	15/02/2001	Maria	800,0
1011	25/02/2001	15/03/2001	Maria	1100,0
1011	16/03/2001	15/04/2001	Maria	1150,0
1011	13/02/2001	30/02/2001	Maria	1000,0

Nova

REGRA 5.

Esta regra aplica-se quando o valor de *tvi* da nova tupla coincide com o valor de *tvi* da tupla de maior extremidade, ou seja, quando o valor de *tvi* de uma nova tupla é a igual ao maior valor de *tvi* já inserido no BDT.

Regra 5.1

SE *TVI.NEW = TVI.MaxOLD* **E**
SE tvf.new = tvf.old
ENTÃO exceção

➢ Esta situação é a mesma ocorrida para a regra 3.1, onde o valor de *tvf* da nova tupla coincide com o valor de *tvf* da tupla armazenada. Desta maneira as duas soluções encontradas são: atualização dos dados não temporais; ou execução de uma exceção proibindo a operação. Como para o caso da regra 3.1 será executada uma exceção proibindo esta operação. Esta situação pode ser verificada na tabela 5.8.

Regra 5.2

SE *TVI.NEW = TVI.MaxOLD* **E**
SE tvf.new > tvf.old
ENTÃO tvf.old := tvf.new
OU ENTÃO exceção

➢ Para esta situação, onde o valor do *tvf* da nova tupla é maior que o valor do *tvf* da tupla armazenada, a atualização do valor do *tvf* da tupla armazenada pode ser realizado desde que os valores dos atributos temporais sejam iguais, produzindo assim uma postergação do tempo de validade da informação. Entretanto, como a comparação entre os valores dos

atributos não é possibilitada por não ter a descrição destes atributos, optou-se pela execução de uma exceção proibindo esta operação.

A tabela 5.15 mostra esta situação. Apesar dos dados não temporais serem iguais, a atualização sobre o *tvf* da tupla armazenada não será efetuada, e a operação será abortada.

TABELA 5.15 – Regra 5.2 *tvf.new > tvf.old*

Id	Tvi	tvf	Nome	salario	
1011	10/01/2001	15/06/2001	Maria	800,0	
1011	16/06/2001	10/11/2001	Maria	1000,0	
1011	16/06/2001	20/11/2001	Maria	1000,0	Nova e Inválida

Regra 5.3

SE *TVI.NEW = TVI.MaxOLD* **E**
SE tvf.new < tvf.old
ENTÃO tvf.old := tvf.new
OU ENTÃO exceção

- Para esta situação, onde o valor do *tvf* da nova tupla é menor que o valor do *tvf* da tupla armazenada, da mesma forma que para regra 5.2, a atualização do valor do *tvf* da tupla armazenada poderia ser realizado desde que os valores dos atributos não temporais fossem iguais, produzindo assim uma diminuição ao invés de uma postergação do tempo de validade da informação. Como para a situação da regra 5.2, nesta situação optou-se também pela execução de uma exceção proibindo esta operação em decorrência da comparação entre os valores dos atributos ser impossibilitada.

Regra 5.4

SE *TVI.NEW = TVI.MaxOLD* **E**
SE tvf.old = NULL
ENTÃO tvf.old := tvf.new
OU ENTÃO exceção

- Para a situação onde o valor do *tvf* da tupla armazenada é nulo, ou seja, indeterminado, da mesma forma que para as regra 5.2 e 5.3, a atualização do valor do *tvf* não é realizada em decorrência da impossibilidade da identificação dos atributos. Desta maneira optou-se também pela execução de uma exceção proibindo esta operação.

REGRA 6. *para TVI.NEW = TVI.MinOLD*

Nesta regra, para todas as circunstâncias envolvidas: (*tvf.new < tvf.old)* ; (*tvf.new > tvf.old) ;(tvf.new = tvf.old) e (tvf.new = NULL)* existem duas soluções: a execução de uma exceção proibindo a operação; ou a atualização dos dados. A descrição dos dados é especificado pelo usuário e varia de aplicação para aplicação, de forma que não há mecanismos pelo qual possam ser identificados. Deste modo não é possível a sua atualização, como foi mencionado na regra 3.1.

Desta forma, a implementação para esta regra não seguirá os padrões do *Times Series Cartridge*, o qual oferece a possibilidade da alteração dos dados. Através da implementação das regras para a inserção do tempo de validade é garantida uma das propriedades dos BDTs, em que não são permitidas operações físicas sobre os atributos.

5.4 Considerações Finais

A dificuldade de implementação do tempo de validade como um todo em SGBDs relacionais e objeto-relacionais é muito grande. Algumas características de BD de tempo de validade não são implementadas ou permitidas no pacote de tempo de validade implementado, devido à limitação imposta pelos recursos do SGBD utilizado. Entretanto, é possível implementar regras que garantam a consistência do BDT, mesmo que estas resultem na proibição da execução de uma determinada operação.

A tabela 5.16 mostra, através de um comparativo, as operações permitidas pelo TSC, as operações permitidas pelo mecanismo de administração do tempo de validade que foi implementado e as operações permitidas se os dois pacotes (TSC e PTV) estiverem executando simultaneamente.

TABELA 5.16 - Comparativo entre as operações do TSC e o PTV

	TSC	**PTV**	**TSC + PTV**
Exclusão física	Sim	Sim (pelo DBA)	Sim (pelo DBA)
Atualização sobre rótulos temporais	Não	Não	Não
Atualização sobre atributos temporais	Sim	Não	Não

Além das operações de exclusão e alteração físicas, o mecanismo que controla o tempo de validade deve considerar as operações de inserção no BDT, que neste caso caracterizam uma operação de atualização lógica das informações. Desta maneira, na maioria dos casos, é necessária uma consulta ao BDT para que sejam recuperadas informações que serão previamente comparadas com os dados a serem inseridos, tornando possível verificar uma possível inconsistência no BDT.

Operações que causariam inconsistência não são permitidas pelo gerenciados do PTV. A tabela 5.17 mostra os recursos fornecidos pelo TSC e pelo PTV através de um comparativo.

TABELA 5.17 – Recursos do TSC e do PTV

	TSC	**PTV**

Rótulo temporal único	Sim	Sim
Validação do rótulo temporal	Sim	Sim
Correção automática de inconsistências	Não	Sim
Verificação de consistência sobre BD populado	Não	Sim
Gerência da Granularidade Temporal	Sim	Não
Mecanismo de Otimização de consultas	Sim	Sim (protótipo)
Controle sobre períodos validos para inserções	Sim	Não

6 Implementação

A inserção e a administração do tempo de validade no BD consistem da execução de três procedimentos: a inserção dos atributos de tempo de validade; a ativação do mecanismo de consistência; e a desativação. Todos os procedimentos implementados no PTV são executados sobre o SQL-PLUS do Oracle, podendo ser utilizados em qualquer outro sistema através de chamadas de procedimentos do SQL. Não existe uma ferramenta específica para interação com o usuário. Neste capítulo são apresentados os procedimentos e os recursos utilizados para a implementação do PTV, e como estes foram implementados.

6.1 Inserção de Atributos de Tempo de Validade

Os atributos necessários para o controle do tempo de validade no BD podem ser inseridos pelo usuário de forma manual, ou utilizando o procedimento *insereAtributoTV*. Entretanto, a inserção manual dos atributos de tempo de validade requer um conhecimento detalhado da estrutura do PTV. O procedimento *insereAtributoTV* requer como parâmetro o nome tabela que conterá os novos atributos.

Ao executar o procedimento *insereAtributoTV*, são inseridos na tabela três atributos:

- o ***ID***, que se refere ao atributo de identificação da tupla. Este atributo também é requerido pelo TSC;
- o ***TvalidadeI***, que representa o tempo de validade inicial;
- o ***TvalidadeF***, que representa o tempo de validade final.

A figura 6.1 apresenta a implementação deste procedimento.

```
PROCEDURE insereAtributoTV(tabela VARCHAR2) AS
BEGIN
      -- cria os atributos de tempo de validade
      EXECUTE IMMEDIATE ' ALTER TABLE '||tabela||' ADD ID VARCHAR2';
      EXECUTE IMMEDIATE ' ALTER TABLE '||tabela||' ADD TvalidadeI DATE';
      EXECUTE IMMEDIATE ' ALTER TABLE '||tabela||' ADD TvalidadeF DATE';
      -- cria indice
      EXECUTE IMMEDIATE ' CREATE INDEX idxPTV ON '||tabela||'
                                                    (ID,TvalidadeI)';
```

FIGURA 6.1 - Implementação do procedimento **insereAtributoTV**

6.2 Ativação dos Mecanismos de Controle do Tempo de Validade

A ativação do mecanismo de controle do tempo de validade consiste da criação de um gatilho que implementa as regras de consistência apresentadas no capítulo 5. A ativação é executada através do procedimento *ativaTV*. Cada tabela especificada pelo usuário terá um gatilho a ela associada. Este gatilho é acionado antes de qualquer operação de inserção, remoção e alteração dos dados.

Conforme apresentado no capítulo 5, foram definidos dois tipos de regras de consistência, as de usuário e as de BD. As regras de usuário são aplicadas sem necessidade de executar qualquer consulta no BD, e sim através de um teste sobre as informações fornecidas pelo usuário. A implementação em PL/SQL das regras de usuários (seção 5.3.1) é mostrada na figura 6.2.

Para referenciar novos e antigos atributos ao utilizar gatilhos, o Oracle utiliza um método de referência através dos objetos *new* e *old*, precedidos de " : " (dois pontos). Apesar da possibilidade de renomear estes objetos, optou-se pela permanência dos nomes *default*, conforme os exemplos das figuras desta seção.

Para a implementação das regras de BD, apresentadas no capítulo 5, antes de efetuar qualquer operação, é necessária uma consulta ao BD a fim de recolher informações sobre os atributos de tempo de validade armazenados. Através da obtenção destas informações é possível realizar uma comparação com os dados submetidos a operações de inserção e atualização, a fim de evitar a inconsistência temporal.

```
CREATE OR REPLACE TRIGGER TV'||tabela||'
BEFORE INSERT OR UPDATE ON '||tabela||'
FOR EACH ROW
… … …
IF INSERTING THEN

  -- regra: tvi <> NULL
  IF :new.TvalidadeI IS NULL THEN
       :new.TvalidadeI := :new.Tstamp;
  END IF;

  -- regra: tvi < tvf
  IF :new.TvalidadeI > :new.TvalidadeF THEN
       Raise_application_error(-20201, ObtemMsgErr(20201));
  END IF;

END IF; -- Inserting

IF UPDATING THEN

  -- regra UPDATE(tvi e tvf)
  IF ((:new.TvalidadeI <> :old.TvalidadeI) OR
      (:new.TvalidadeF <> :old.TvalidadeF)) THEN
       Raise_application_error(-20202, ObtemMsgErr(20202));
  END IF;
```

FIGURA 6.2 – Implementação das regras de usuários

As pré-consultas ao BD são realizadas através do recurso de *variáveis cursores* do Oracle, e as informações obtidas são armazenadas em variáveis para posterior utilização. As informações necessárias e as variáveis utilizadas são:

- *tviMaxOld* DATE -- maior valor de tvi no BD para o ID especificado;
- *tviMinOld* DATE -- menor valor de tvi no BD para o ID especificado;
- *tvfMaxOld* DATE -- TvalidadeF da tupla recuperada de maior TvalidadeI;
- *tvfMinOld* DATE -- TvalidadeF da tupla recuperada de menor TvalidadeI;
- *tviOldNext* DATE -- TvalidadeI da tupla subseqüente para o ID;
- *tvfOldNext* DATE -- TvalidadeF da tupla subseqüente para o ID;
- *tviOldPrior* DATE -- TvalidadeI da tupla que antecede para o ID;
- *tvfOldPrior* DATE -- TvalidadeF da tupla que antecede para o ID;

A figura 6.3 mostra como estas informações são recuperadas através de variáveis cursores.

```
-- definição do cursor para recuperação de informações sobre o TV.

CURSOR maxOld IS SELECT TvalidadeI,TvalidadeF FROM '||tabela||
' WHERE TvalidadeI IN
       (SELECT MAX(TvalidadeI) FROM '||tabela||
       ' WHERE ID = :new.ID);

CURSOR existeTV IS SELECT TvalidadeI,TvalidadeF FROM '||tabela||
' WHERE ID = :new.ID AND TvalidadeI = :new.TvalidadeI;

CURSOR minOld IS SELECT TvalidadeI,TvalidadeF FROM '||tabela||
' WHERE TvalidadeI IN
       (SELECT MIN(TvalidadeI) FROM '||tabela||
       ' WHERE ID = :new.ID);

CURSOR proximo IS SELECT TvalidadeI,TvalidadeF  FROM '||tabela||
' WHERE TvalidadeI IN
       (SELECT MIN(TvalidadeI) FROM '||tabela||
       ' WHERE ID = :new.ID AND TvalidadeI > :new.TvalidadeI);

CURSOR anterior IS SELECT TvalidadeI,TvalidadeF FROM '||tabela||
' WHERE TvalidadeI IN
       (SELECT MAX(TvalidadeI) FROM '||tabela||
              ' WHERE ID = :new.ID AND TvalidadeI < :new.TvalidadeI);

-- Abre os Cursores
OPEN minOld;
FETCH minOld INTO tviMinOld,tvfMinold;

OPEN maxOld;
FETCH maxOld INTO tviMaxOld,tvfMaxOld;

OPEN existeTV;
```

FIGURA 6.3 – Recuperação de informações através de variáveis cursor

As regras de consistência temporal do BD apresentadas no capítulo 5 podem ser aplicadas a partir da obtenção destas informações. As figuras 6.4 a 6.9 mostram respectivamente a implementação das regras de consistência de 1 a 5 apresentadas no capítulo 5.

```
-- implementação das regras  1.1, 1.2 e  1.3

IF (:new.TvalidadeI >  tviMaxOld) AND
  ((:new.TvalidadeI <= tvfMaxOld)  OR
   tvfMaxOld IS NULL) THEN
      UPDATE '||tabela||
      ' SET TvalidadeF = :new.TvalidadeI -1
      WHERE ID = :new.ID AND TvalidadeI = tviMaxOld;
END IF;
```

FIGURA 6.4 – Implementação das regras de consistência 1.1, 1.2 e 1.3

```
-- implementação das regras 2.1,  2.2 e 2.3

IF ((:new.TvalidadeI <  tviMinOld) AND
    (:new.TvalidadeF >= tviMinOld)) THEN
      IF ((:new.TvalidadeF +1 < tvfMinOld) AND
            :new.TvalidadeF IS NOT NULL) THEN
               UPDATE '||tabela||
                ' SET TvalidadeI = :new.TvalidadeF +1
                WHERE ID = :new.ID AND TvalidadeI = tviMinOld;
      ELSE
             Raise_application_error(-20203, ObtemMsgErr(20203));
       END IF;
END IF;
```

FIGURA 6.5 – Implementação das regras de consistência 2.1, 2.2 e 2.3

```
-- implementação das regras 3.1,  3.2 e 3.3

-- se existir um valor de tvi igual ou valor do tvi.new
IF ((:new.TvalidadeI > tviMinOld) AND (:new.TvalidadeI < tviMaxOld)) THEN
       IF existeTV%FOUND THEN
          Raise_application_error(-20204, ObtemMsgErr(20204));
       END IF;
END;
```

FIGURA 6.6 – Implementação das regras de consistência 3.1, 3.2 e 3.3

```
-- implementação das regras 4.1 e 4.2

-- se não existir um valor de tvi igual ou valor do tvi.new
IF existeTV%NOTFOUND THEN
       -- regra 4.1
       IF (:new.TvalidadeF >= tviOldNext) THEN
              IF (:new.TvalidadeF + 1 <= tvfOldNext) THEN
                     UPDATE '||tabela||
                     ' SET TvalidadeI = :new.TvalidadeF +1
                     WHERE ID = :new.ID AND TvalidadeI = tviOldNext;
              ELSE
                     Raise_application_error(-20205, ObtemMsgErr(20205));
              END IF;
       ELSE
       -- regra 4.2
              IF (:new.TvalidadeI < tvfOldPrior) THEN
                     UPDATE '||tabela||
                     ' SET TvalidadeF = :new.TvalidadeI - 1
                     WHERE ID = :new.ID AND TvalidadeI = tviOldPrior;
              END IF;
       END IF;
END IF;
```

FIGURA 6.7 – Implementação das regras de consistência 4.1, 4.2

```
-- implementação das regras 5.1, 5.2, 5.3 e 5.4

IF (:new.TvalidadeI = tviMaxOld) THEN
            Raise_application_error(-20206, ObtemMsgErr(20206));
END IF;
```

FIGURA 6.8 – Implementação das regras de consistência 5.1, 5.2, 5.3 e 5.4

```
-- implementação da regra 6
IF (:new.TvalidadeI = tviMinOld) THEN
      Raise_application_error(-20207, ObtemMsgErr(20207));
END IF;
```

FIGURA 6.9 – Implementação das regras de consistência 6

6.3 Desativação dos Mecanismos de Controle do Tempo de Validade

A desativação do mecanismo de controle do tempo de validade consiste na exclusão do gatilho associado à tabela especificada. Este processo é efetivado através da execução do procedimento *desativaTV*. Este procedimento não exclui os atributos de tempo, somente desativa a verificação de consistência de informações temporais durante operações sobre o BD. Os procedimentos para ativação e desativação do PTV podem ser realizados a qualquer instante. Entretando, se o PTV for ativado sobre uma base de dados populada é necessário a utilização do procedimento *verConsistenciaTV* para verificação de inconsistências temporais.

A figura 6.10 apresenta a implementação do procedimento *desativaTV*.

```
-- implementação do procedimento desativaTV
PROCEDURE desativaTV(tabela VARCHAR2) AS
BEGIN
             -- exclui o gatilho associado a tabela
             EXECUTE IMMEDIATE ' DROP TRIGGER TV'||tabela||' ';
END desativaTV;
```

FIGURA 6.10 – Implementação do procedimento **desativaTV**

6.4 Controle de Exceções

Exceções são execuções de processos paralelos a um evento ou operação que foi executado, mas que por algum motivo não pode ser realizado. Geralmente, os SGBDs e aplicações em geral tratam uma eventual operação mal sucedida com a execução de uma mensagem informativa sobre o motivo do não prosseguimento da operação.

O tratamento de operações não permitidas durante a verificação de consistência temporal do BD, consiste na execução de eventos em forma de mensagens, informando o motivo pelo qual a operação foi abortada. A apresentação das mensagens de erros é possibilitada através do recurso do PL/SQL do Oracle *"Raise_application_error"*. O Oracle também utiliza-se deste recurso para a execução de seus eventos de erros. O PTV utiliza-se do intervalo disponível pelo Oracle para a criação de novas mensagens de erro que vai do número –20201 a –20207.

Estes eventos de operações não permitidas são ativados a partir do gatilho que controla a consistência temporal da tabela, acionado através da execução de uma função denominada *obtemMsgErr*. A figura 6.11 apresenta a implementação do procedimento de tratamento de exceções do PTV.

```
FUNCTION ObtemMsgErr(numErr NUMBER) RETURN VARCHAR2 IS
BEGIN

CASE NUMBER
   WHEN      20201 THEN RETURN 'TVI não pode ser maior que o TVF.';
   WHEN      20202 THEN RETURN 'Não é permitida operação de atualização sobre
                                atributos de tempo.';
```

FIGURA 6.11 – Implementação das Exceções

6.5 Mecanismos de Recuperação de Dados de Tempo

Nesta seção são analisados os mecanismos utilizados pelo TSC para facilitar a realização de consultas sobre o rótulo temporal, e, é apresentado o protótipo da função *PERIODO* , recurso implementado e inserido no PTV, com o objetivo de facilitar a criação de consultas sobre o rótulo temporal de tempo de validade.

6.5.1 *Time Series Cartridge*

Além de recursos para inserção e administração do rótulo temporal, o TSC fornece mecanismos para otimizar a recuperação dos dados através de funções. Em [ORA 99] são apresentados dois tipos de funções, as funções para séries de tempo e as funções de escala de tempo. O objetivo principal é facilitar e otimizar o acesso aos dados.

Com o auxílio das funções para séries de tempo do TSC, a consulta: "*Selecione a média salarial acumulativa da funcionária Maria durante o período de jun-1995 a jun-2001*" poderia ser recuperada utilizando a função para média acumulativa *cavg*, como mostra a figura 6.12.

```
SELECT to_char(tstamp) tstamp, value
FROM Folha_pgto fp,
     TABLE (CAST(ORDSYS.TimeSeries.ExtractTable(
             ORDSYS.TimeSeries.Cavg(fp.salario,to_date(
                '01-MAI-1995','DD-MON-YYYY'),
       to_date('01-MAI-2001','DD-MON-YYYY'))
       ) AS ORDSYS.ORDTNumTab)) t
WHERE fp.ID='1011';
```

FIGURA 6.12 – Exemplo do uso da função do TSC (cavg)

Quando o TSC está ativo são fornecidas algumas funções para acesso aos dados através do rótulo temporal de tempo de transação, como : *TSAvg* - retorna a média de um valor em uma série de tempo; *TSMax* - retorna o maior valor em uma série de tempo; *TSMedian* - retorna o elemento do meio em uma série de tempo. Outras funções e procedimentos podem ser vistos em [ORA 99].

6.5.2 *Pacote de Tempo de Validade*

Da mesma forma como no TSC, um recurso foi inserido no PTV com a finalidade de otimizar a recuperação dos dados de tempo. Este recurso consiste de uma função denominada *PERIODO* que é utilizada juntamente com a expressão *SELECT* do SQL.

A implementação da função *PERIODO* é baseada nas sugestões dos modelos relacionais temporais, mais especificamente na linguagem de consulta temporal estruturada TSQL2 [SNO 95], proposta com o objetivo de consolidar uma linguagem de consulta temporal. A figura 6.13 apresenta a descrição de uma consulta utilizando o TSQL2, bem como a descrição da mesma consulta em SQL padrão. A descrição da figura 6.13 corresponde à consulta:

"*Selecione o nome dos funcionários que receberam mais de R$500,00 no período de jun-1998 a jun-2000*".

```
-- Consulta convencional

SELECT func.nome
FROM funcionario func, folha_pgto fp
WHERE func.ID = fp.ID
      AND fp.salario > 500
      AND ((    fp.TvalidadeI >= '01/05/1999'
            AND fp.TvalidadeF <= '01/05/2000'   )
            OR ( fp.TvalidadeF = NULL
            AND  fp.TvalidadeI >= '01/05/1999'
            AND  fp.TvalidadeI <  '01/05/2000' )
           );

-- Consulta utilizando TSQL2

SELECT fun.nome
FROM funcionrio func, folha_pgto fp
WHERE func.ID = fp.ID
     AND fp.salário >500
     AND VALID(funcionario)
     CONTAINS  PERIOD(DATE '01/05/1999',DATE '01/05/2000');
```

FIGURA 6.13 – Consultas temporais

Apesar da função *PERIODO* ter assimilado os parâmetros de entrada e o resultante da instrução *CONTAINS PERIOD*, exemplificada na figura 6.13, a funcionalidade da função implementada não pode ser comparada ao sugerido no TSQL2, verificadas as limitações dos recursos do PL/SQL do ORACLE.

Os parâmetros de entrada da função *PERIODO* implementada são: a tabela, de onde será recuperada a série de tempo; uma data representando o início de um período; e uma data representando o final de um período. O resultante é uma série de tempo, ou seja, uma coleção de objetos de tempo de validade. A coleção de objetos de tempo é um montante de objetos de tempo válidos para o período especificado. Um objeto de tempo é formado pelo tempo de validade inicial e final. A figura 6.14 apresenta a implementação do tipo objeto de tempo e a coleção de objetos de tempo. O Oracle fornece dois modelos para trabalhar com coleções de tipos, o *VARRAY* e o *NESTED TABLE*. O *VARRAY* tem um tamanho fixo, devendo ser especificado no momento da criação da coleção, enquanto o *NESTED TABLE* é de tamanho variável. Desta maneira optou-se na implementação do modelo por *NESTED TABLE*.

```
-- Criacao do objeto de tempo " Tvalidade "
CREATE OR REPLACE TYPE Tvalidade AS OBJECT (
tvali
dadei DATE,
tvalidadef DATE);

-- Cricao da Coleção de objetos de tempo através de NESTED TABLE
CREATE OR REPLACE TYPE colecaoTV AS TABLE OF Tvalidade;

-- Criacao do Objeto Serie de Tempo
CREATE OR REPLACE TYPE SerieTempo AS OBJECT ( NomeTab varchar2(15),
```

FIGURA 6.14 – Implementação da criação de objetos e coleções de tempo de validade.

A função *PERIODO* consiste da criação de uma coleção de objetos de tempo através da execução de uma consulta ao BD. Na consulta são recuperados objetos de tempo que correspondam ao período fornecido nos parâmetros de entrada. A implementação desta função é apresentada na figura 6.15.

```
FUNCTION periodo(ST SerieTempo) return colecaoTV IS
i integer;
tvi DATE;
tvf DATE;
id  Varchar2(15);
colO ColecaoTV:=ColecaoTV(); -- Inicia uma colecao de objetos
TYPE Cursores is REF CURSOR;
c1 Cursores; -- variavel do tipo cursor
c2 Cursores; -- variavel do tipo cursor
c3 Cursores; -- variavel do tipo cursor
c4 Cursores; -- variavel do tipo cursor
BEGIN
if (periodoINI is not null) AND (periodoFIM is not null) then
        OPEN c3 FOR 'SELECT TvalidadeI,TvalidadeF,ID FROM '||tabela||' WHERE (
        ( TvalidadeI>='||''''||TO_char(periodoINI,'dd/mm/yyyy')||''''||' ) AND
        ( TvalidadeF <= '||''''||TO_char(periodoFIM,'dd/mm/yyyy')||''''||' ) AND
        ( TvalidadeF is not NULL ) ) OR
        ((TvalidadeI >= '||''''||TO_char(periodoINI,'dd/mm/yyyy')||''''||' ) AND
        ( TvalidadeI < '||''''||TO_char(periodoFIM,'dd/mm/yyyy')||''''||' ) AND
        ( TvalidadeF = NULL ))';
        LOOP
                FETCH c3 INTO tvi,tvf,id;
                EXIT WHEN c3%NOTFOUND;
                colO.extend;
                colO(colO.last) := Tvalidade(id,tvi,tvf);
                i:=i+1;
        END LOOP;
```

FIGURA 6.15 – Implementação da função “PERIODO”.

A função *PERIODO* deve ser utilizada em conjunto com a instrução *SELECT SQL*. Entretanto, não é possível associar um parâmetro da cláusula *WHERE*, por exemplo, com a coleção de objetos retornada pela função. Nesta hipótese um determinado dado está sendo comparado com um objeto, resultando uma incompatibilidade de tipos.

No entanto, o ORACLE fornece um recurso em forma de função chamado *CAST*, para que seja possível mapear um tipo de dado ou uma coleção de tipos de dados para um outra coleção de tipos de dados diferentes. Deste modo é possibilitada a comparação de uma seqüência de dados com uma coleção de objetos. A figura 6.16 apresenta a utilização da função *PERIODO* em conjunto com a função *CAST*, para a seguinte a consulta:

“*Selecione o nome dos funcionários que receberam mais de R$500,00 no período de jun-1998 a jun-2000*”.

```
SELECT func.nome
FROM funcionario func
WHERE func.salário > 500 AND
      func.id IN (
      SELECT per.id
      FROM TABLE(CAST(ptv.periodo('folha_pgto',
                        '01/06/1998','01/06/2000') as colecaoTV) per);
```

FIGURA 6.16 – Consulta utilizando a função PERIODO inserida no PTV.

6.6 Considerações Finais

Neste capítulo foram apresentados as estrutura do PTV e os procedimentos implementados para inserção, ativação e administração do rótulo de tempo de validade. Foi mostrada a metodologia para a execução do PTV, e a implementação dos procedimentos.

Apesar do PTV inserir e administrar o tempo de validade no BD relacional, não foram implementadas todas as características de BDT, devido às limitações de recursos do SGBD Oracle e a preocupação com a performance de operações sobre no BD. Em muitos casos a reação do PTV a operações de usuários sobre o BDT é a execução de exceções,

informando que a operação não pode ser executada. Entretanto, foi possível a inserção do tempo de validade, bem como manter a consistência dos rótulos temporais. As limitações de recursos na linguagem PL/SQL interferiram na implementação de procedimentos e funções, principalmente na função *PERIODO*, implementada para facilitar a recuperação de informações associadas ao rótulo temporal.

7 Estudo de caso

O PTV foi desenvolvido no intuito de facilitar e otimizar a implementação de sistemas que utilizam informações temporais. Para exemplificar sua utilização, neste capítulo é mostrado um estudo de caso, o sistema de controle sindical da Federação dos Trabalhadores na Agricultura no RS.

A Federação conta com aproximadamente 350 sindicatos filiados. Atualmente um sistema atemporal, com a base de dados em Oracle, armazena informações cadastrais bem como financeiras de cada sindicado. Entre as informações cadastrais estão armazenados dados sobre os componentes da diretoria (presidente, vice, tesoureiro, data de posse, duração do mandato, etc). O mandato de cada diretoria varia de dois a quatro anos dependendo do estatuto de cada sindicato.

Atualmente existe uma grande dificuldade para a recuperação dos dados referentes às diretorias, bem como de um histórico de duração de mandatos. Por exemplo, pode-se citar a seguinte consulta: "*Recuperar os componentes da diretoria que atuaram no período X* ". A dificuldade da realização desta consulta reside no fato de existirem apenas dois dados de atuação: a data de posse e a duração do mandato. A dificuldade aumenta quando o período solicitado não corresponde ao valor da data de posse. Atualmente estas consultas são realizadas por usuários especializados ou até mesmo por administradores do sistema.

7.1 Modelagem do Sistema

Devido à grande quantidade de informações existentes na modelagem do sistema, foram selecionadas para a execução do PTV somente as tabelas nas quais é relevante a inserção do tempo. Desta aplicação, as tabelas relacionadas são as seguintes: *Entidade* - armazena os dados dos sindicatos, tipo CGC, endereço, etc; *Diretoria* - armazena a nominata da diretoria do sindicato, bem como data de posse e duração do mandato; *Suplente* – armazena, se houver, os suplentes da diretoria.

Estas tabelas são criadas pelos seguintes comandos:

ENTIDADE

```
CREATE TABLE ENTIDADE
   (
    idStr   VARCHAR2(10),
    municipio     VARCHAR2(50),
    CGC           VARCHAR2(13),
    filiacao DATE,
    fundacao DATE
    ); ...
```

DIRETORIA

```
CREATE TABLE DIRETORIA
    (
    idStr   VARCHAR2(10),
    Npresidente   VARCHAR2(50),
    Nvice   VARCHAR2(50),
    Ntesoureiro VARCHAR2(50),
    Nsecretario VARCHAR2(50),
    dta_posse   DATE,
    duracao_mandato NUMBER,
    idSuplente  NUMBER
      ); ...
```

SUPLENTE

```
CREATE TABLE SUPLENTE
   (
    idSuplente   Number,
    hierarquia    Number,
    Nome    VARCHAR2(50)
    );
```

Outros atributos e tabelas fazem parte do modelo apresentado. Entretanto, não são apresentadas por não serem necessárias para que a execução do PTV possa ser exemplificada e entendida. A figura 7.1 mostra todo o diagrama de classes do estudo de caso. As tabelas relevantes à inserção do tempo, estão em destaque.

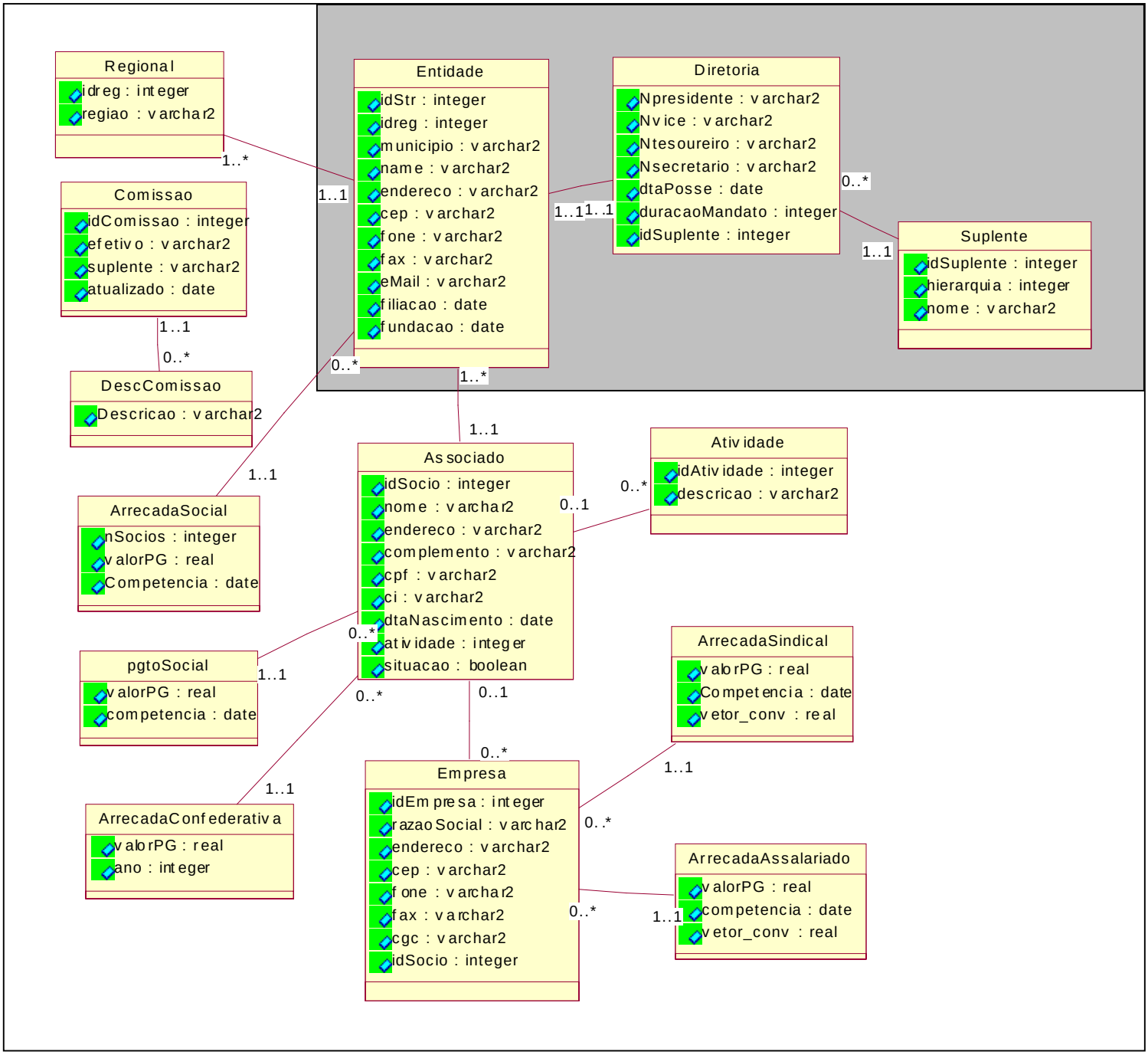

Regional
idreg : integer
regiao : varchar2
Entidade
idStr : integer
idreg : integer
municipio : varchar2
name : varchar2
endereco : varchar2
cep : varchar2
fone : varchar2
fax : varchar2
eMail : varchar2
filiacao : date
fundacao : date
Diretoria
Npresidente : varchar2
Nvice : varchar2
Ntesoureiro : varchar2
Nsecretario : varchar2
dtaPosse : date
duracaoMandato : integer
idSuplente : integer
Suplente
idSuplente : integer
hierarquia : integer
nome : varchar2
Comissao
idComissao : integer
efetivo : varchar2
suplente : varchar2
atualizado : date
DescComissao
Descricao : varchar2
Associado
idSocio : integer
nome : varchar2
endereco : varchar2
complemento : varchar2
cpf : varchar2
ci : varchar2
dtaNascimento : date
atividade : integer
situacao : boolean
Atividade
idAtividade : integer
descricao : varchar2
ArrecadaSocial
nSocios : integer
valorPG : real
Competencia : date
pgtoSocial
valorPG : real
competencia : date
ArrecadaConfederativa
valorPG : real
ano : integer
ArrecadaSindical
valorPG : real
Competencia : date
vetor_conv : real
Empresa
idEmpresa : integer
razaoSocial : varchar2
endereco : varchar2
cep : varchar2
fone : varchar2
fax : varchar2
cgc : varchar2
idSocio : integer
ArrecadaAssalariado
valorPG : real
competencia : date
vetor_conv : real
1..*
1..1
0..*
1..1
1..1
0..*
1..1
0..*
1..1
0..*
1..*
1..1
0..*
0..1
0..*
1..1
0..*
0..1
0..*
1..1
0..*
1..1

FIGURA 7.1 – Diagrama de Classes do Estudo de Caso

7.2 Inserção do Pacote de Tempo de Validade

Antes do procedimento de inserção dos atributos temporais, é necessária uma análise do modelo para verificar as tabelas que conterão os atributos e o controle temporal. Analisado o modelo apresentado e o objetivo proposto, é identificada a necessidade da inserção do tempo somente na tabela *diretoria.* Entretanto, por necessidade ou otimização do modelo, a tabela deverá sofrer algumas alterações:

1. o nome do atributo *idStr*, referente a identificação do sindicato, deverá ser renomeado para *id*, conforme requerido pelo PTV;
2. o atributo *dta_posse*, referente à data de posse da diretoria, será definido como rótulo temporal, a fim de usufruir das funções e procedimentos oferecidos pelo TSC;
3. o atributo *duracao_mandato*, que se refere à duração do mandato, não terá mais utilidade, visto que serão inseridos o tempo de validade inicial e final, devendo ser suprimido.

O primeiro procedimento a ser realizado é a inserção dos atributos de tempo na tabela *diretoria*. Optou-se pelo uso do PTV em conjunto com o TSC para que sejam utilizadas as características de tempo de validade e tempo de transação, desta forma a tabela deverá conter o atributo *tstamp*, requerido pelo TSC. A figura 7.2 apresenta a inserção dos atributos de tempo de validade e do rótulo temporal, através da execução da função *insereAtributoTV*. A execução do TSC é mostrada nas figuras 7.7 e 7.8.

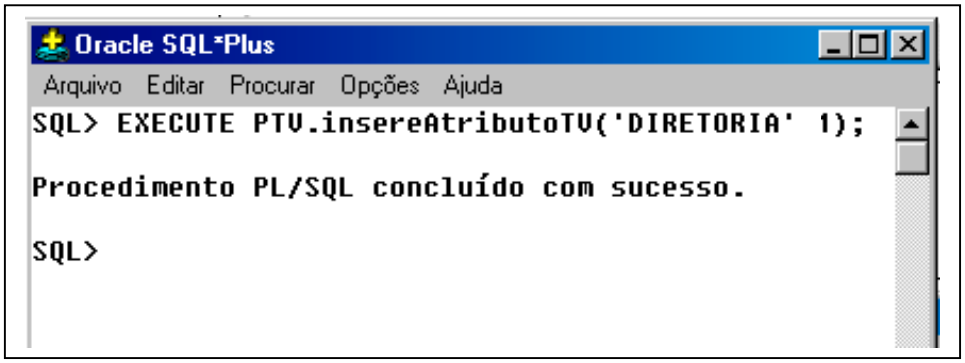


FIGURA 7.2 – Oracle. Exemplo da função **insereAtributoTV()**.

Após a criação dos atributos de tempo, é possível implementar a primeira alteração sugerida após a análise do modelo. Como o Oracle não permite que um nome de atributo seja renomeado, os valores do atributo *"idStr"* deverão ser transportados para o novo

atributo ”*id*”, e finalmente o atributo “*idStr*” poderá ser suprimido. A figura 7.3 apresenta este procedimento implementado no Oracle.

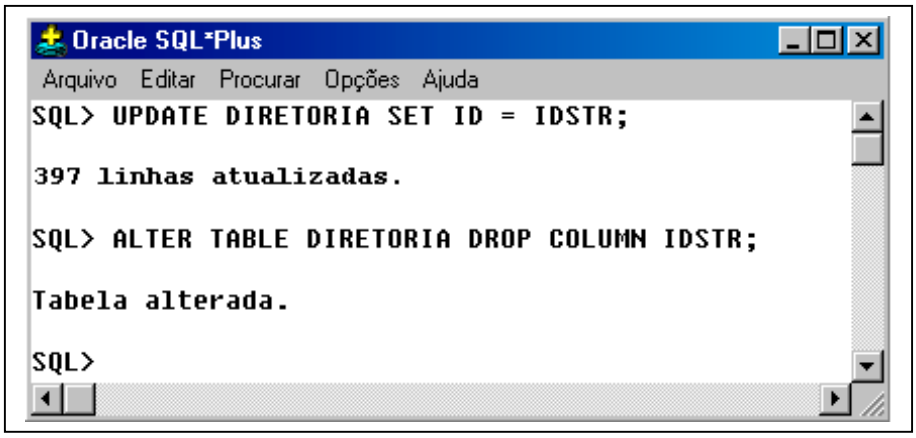


FIGURA 7.3 – Oracle. Reformulação do atributo de identificação.

Conforme sugerido, os valores do atributo “*dta_posse”* serão transferidos para o atributo temporal *tstamp*, sendo o atributo “*dta_posse*” suprimido da mesma forma como ocorreu para a situação da figura 7.3. A figura 7.4 apresenta esta implementação no Oracle.

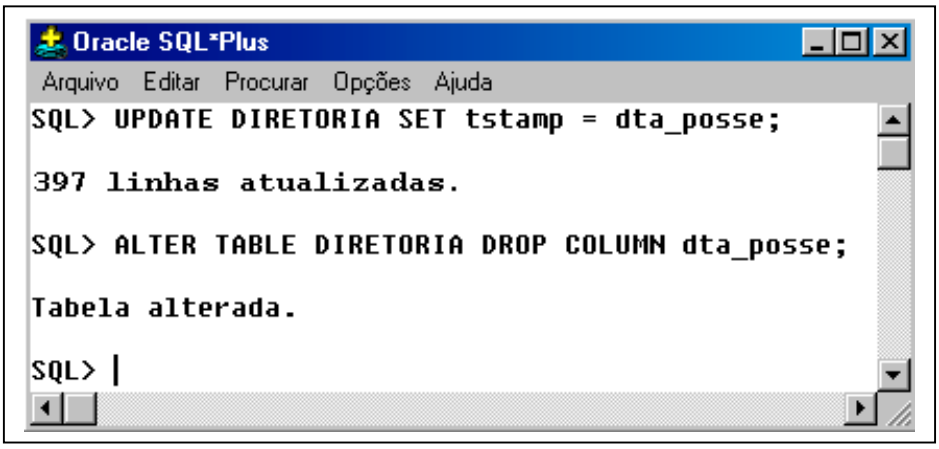


FIGURA 7.4 – Oracle. Organização do rótulo temporal.

Com a inserção dos atributos de tempo de validade inicial e final, o armazenamento do atributo “*duracao_mandado”* não é necessário. Porém, antes se suprimi-lo, deve-se utilizá-lo para a inserção dos valores dos atributos de tempo de validade. Isso só deve ser feito para atributos que tenham associados explicitamente o tempo de validade. Depois de

ativado o PTV, o controle da validade é feito por ele automaticamente. É definido que os valores do tempo de validade inicial serão os mesmos do rótulo temporal, que neste modelo está representando a data de posse da diretoria. Os valores do tempo de validade final serão representados pelo valor do rótulo temporal acrescidos da duração do mandato. Estes procedimentos são implementados no Oracle e apresentados na figura 7.5.

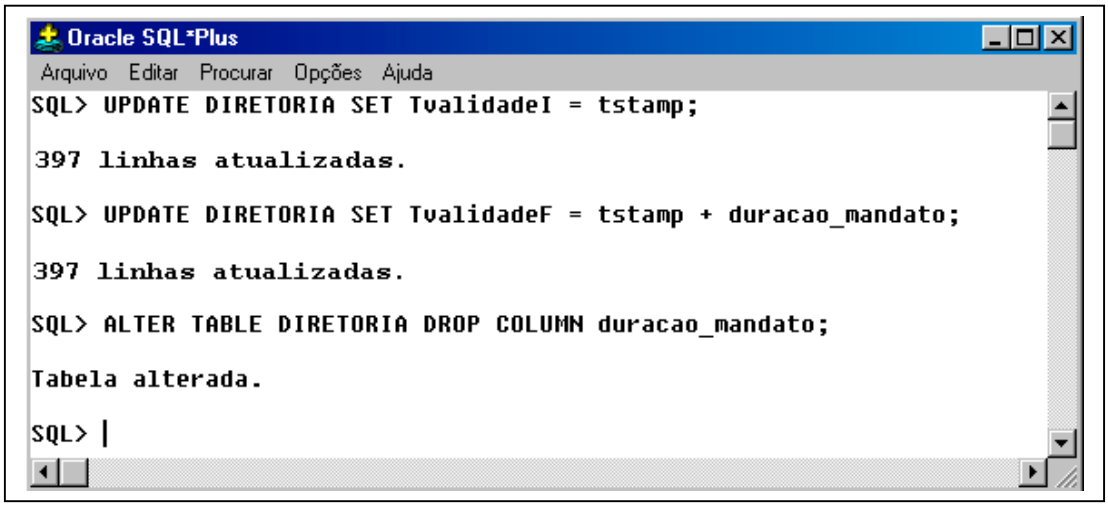


FIGURA 7.5 – Oracle. Inserção dos valores do tempo de validade.

Depois de realizados os procedimentos listados nesta seção, o mecanismo de controle temporal do PTV pode ser ativado, conforme mostra a figura 7.6. A partir deste procedimento qualquer operação realizada sobre a base de dados envolvida, será interceptada e administrada pelo PTV.

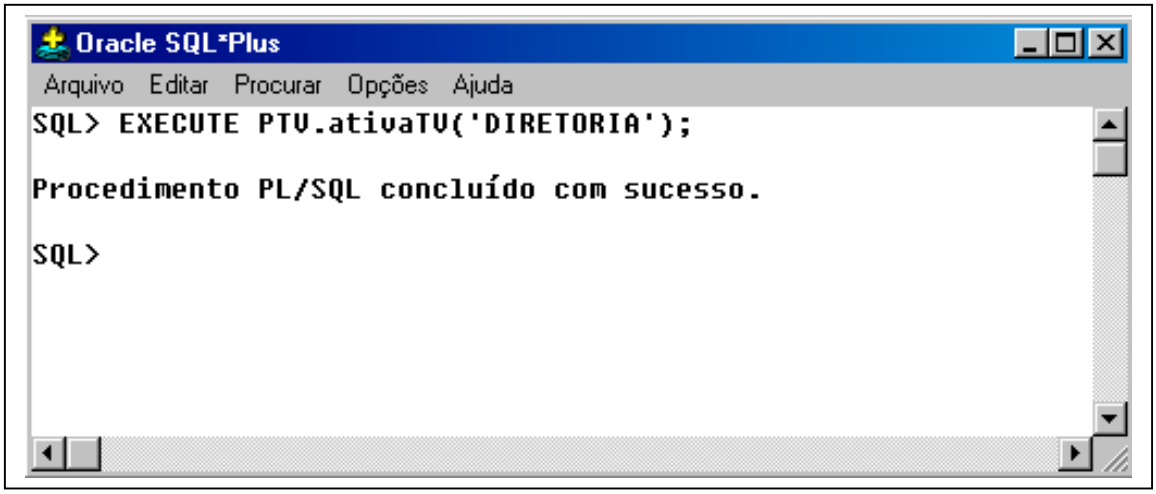


FIGURA 7.6 – Oracle. Ativação do PTV.

No modelo do estudo de caso é proposta a utilização do PTV em conjunto com o TSC. Para a execução do TSC sobre uma base de dados já populada, é necessário que :

- para cada tabela envolvida exista um atributo de identificação e um índice primário;
- exista o atributo do rótulo temporal;
- exista a tabela de metadados, com o atributo para o nome do calendário.

Verificadas as exigências para a inserção do TSC em uma base de dados populada, verifica-se que no modelo do estudo de caso, o identificador e o rótulo temporal são representados pelos atributos "*id*" e "*tstamp*" respectivamente. Desta forma, é necessária a criação de uma tabela de metadados. A figura 7.7 apresenta a criação desta tabela.

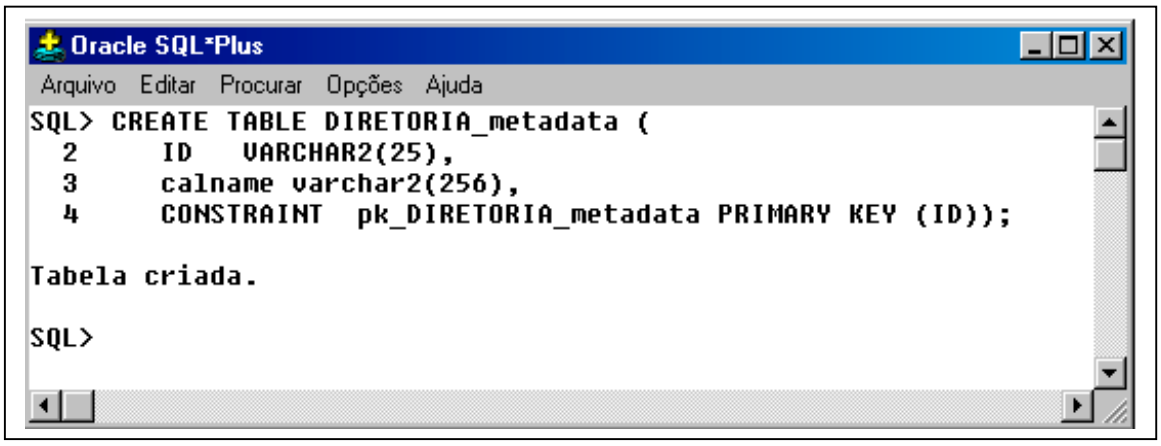


FIGURA 7.7 – Oracle. Criação da tabela de metadados.

Depois de implementados os requisitos, pode ser executado o procedimento de inserção do TSC através da criação de um esquema. Inicialmente deve-se especificar um nome para o esquema de tempo da base de dados do estudo de caso. A figura 7.8 mostra como verificar a existência de um esquema já existente. A execução do exemplo apresentado na figura resulta em um erro, pois o nome do esquema que será criado não existe.

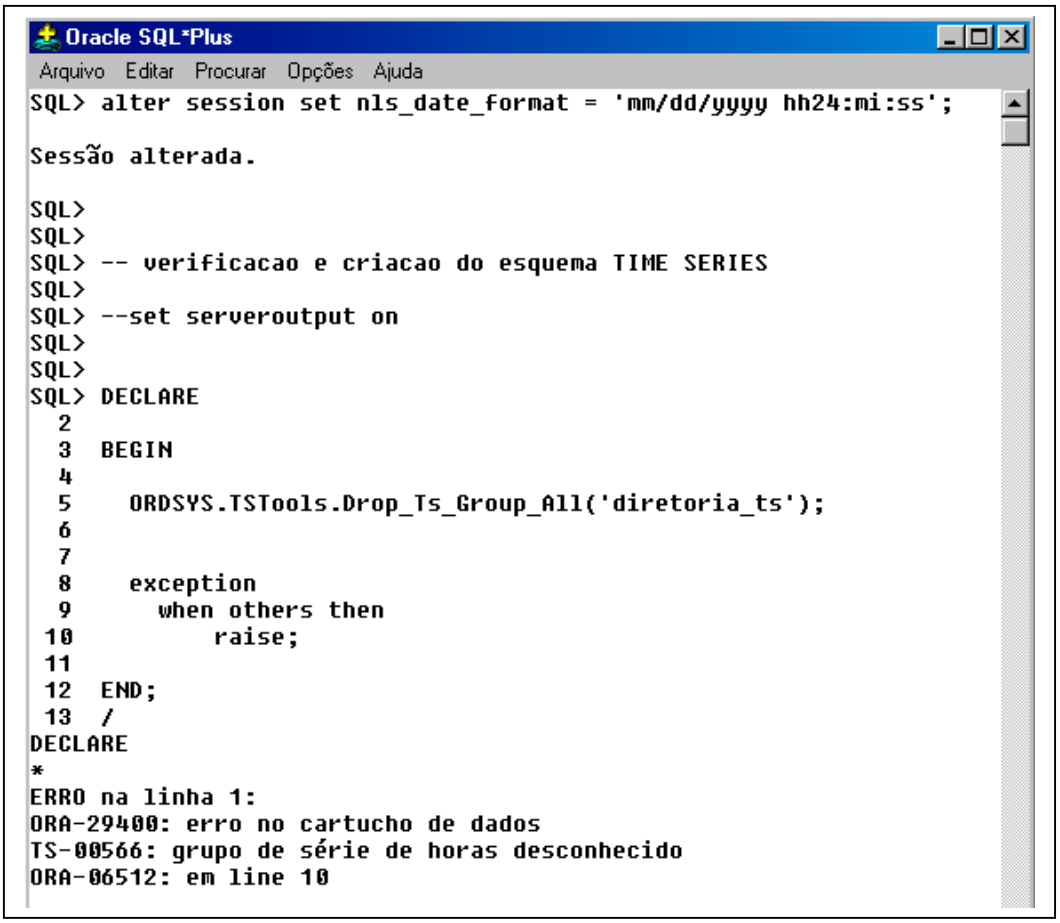


FIGURA 7.8 – Oracle. Verificação se o nome do esquema de série de tempo existe (TSC).

Não existindo o esquema de série de tempo desejado, é possível executar os procedimentos para a criação do novo esquema do TSC. O procedimento necessário é implementado e mostrado através da figura 7.9.

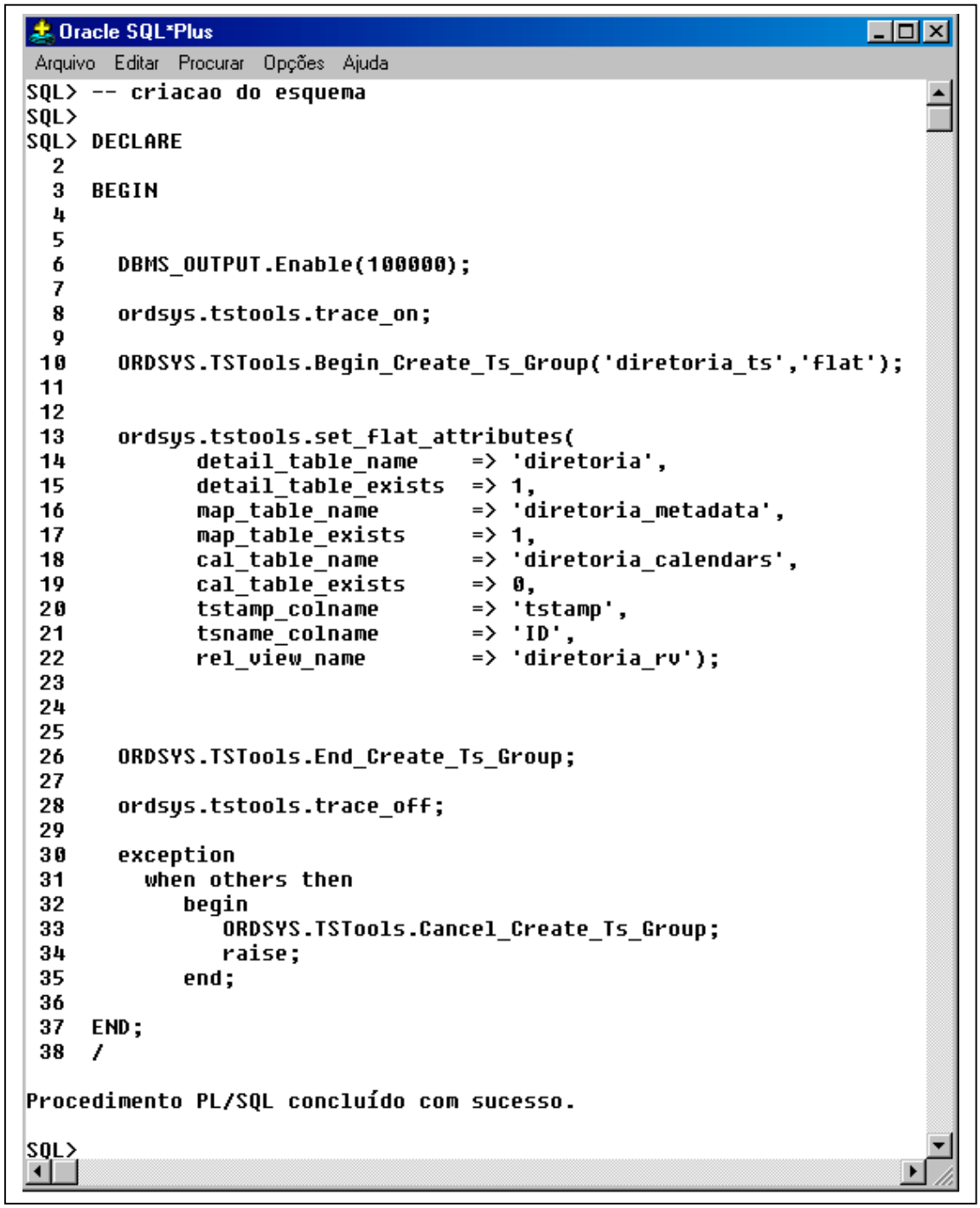

Oracle SQL*Plus
Arquivo Editar Procurar Opções Ajuda
SQL> -- criacao do esquema
SQL>
SQL> DECLARE
2
3 BEGIN
4
5
6 DBMS_OUTPUT.Enable(100000);
7
8 ordsys.tstools.trace_on;
9
10 ORDSYS.TSTools.Begin_Create_Ts_Group('diretoria_ts','flat');
11
12
13 ordsys.tstools.set_flat_attributes(
14 detail_table_name => 'diretoria',
15 detail_table_exists => 1,
16 map_table_name => 'diretoria_metadata',
17 map_table_exists => 1,
18 cal_table_name => 'diretoria_calendars',
19 cal_table_exists => 0,
20 tstamp_colname => 'tstamp',
21 tsname_colname => 'ID',
22 rel_view_name => 'diretoria_rv');
23
24
25
26 ORDSYS.TSTools.End_Create_Ts_Group;
27
28 ordsys.tstools.trace_off;
29
30 exception
31 when others then
32 begin
33 ORDSYS.TSTools.Cancel_Create_Ts_Group;
34 raise;
35 end;
36
37 END;
38 /
Procedimento PL/SQL concluído com sucesso.
SQL>

FIGURA 7.9 – Oracle. Criação de um esquema de série de tempo (TSC).

A remodelagem do sistema do estudo de caso não prevê a necessidade da utilização de calendários de tempo, pois os dados do sistema podem ser inseridos a qualquer momento. No entanto, para melhor exemplificar a utilização do TSC foi inserido, no esquema *diretoria_ts*, um calendário com granularidade de ano, conforme é apresentado na figura 7.10.

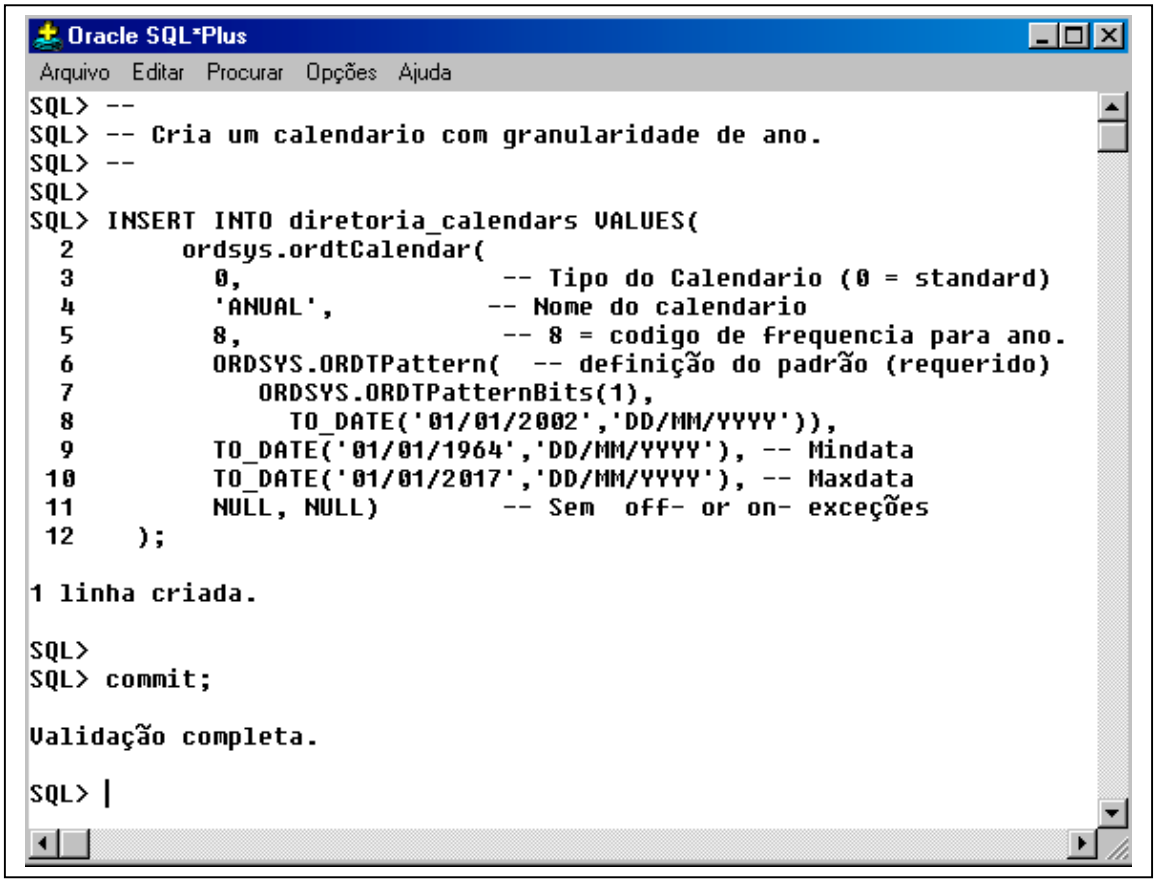


FIGURA 7.10 – Oracle. Criação do calendário para a série de tempo (TSC).

Para exemplificar a reação do PTV a uma operação do usuário, a figura 7.11 simula a execução de uma inserção no BD, e a figura 7.12 mostra qual foi a reação do mecanismo de administração temporal do PTV. Primeiramente a figura 7.11 mostra a execução no SQL/PLUS de uma consulta sobre o BDT e apresenta o seu resultado. Em seguida, apresenta uma operação de inserção. O resultado pode ser verificado na figura 7.12

Nos dados resultantes da consulta apresentada na figura 7.11 nota-se que existe um intervalo temporal sem informações, referente ao intervalo de mandato do presidente “EURICO PEIXOTO DE ASSIS” ao presidente “JUAREZ DA ROSA CANDIDO”. Presume-se que o usuário não atualizou o BDT ao termino do último mandato e fez a

atualização em outro instante de tempo através da operação de inserção como mostra no decorrer da figura 7.11.

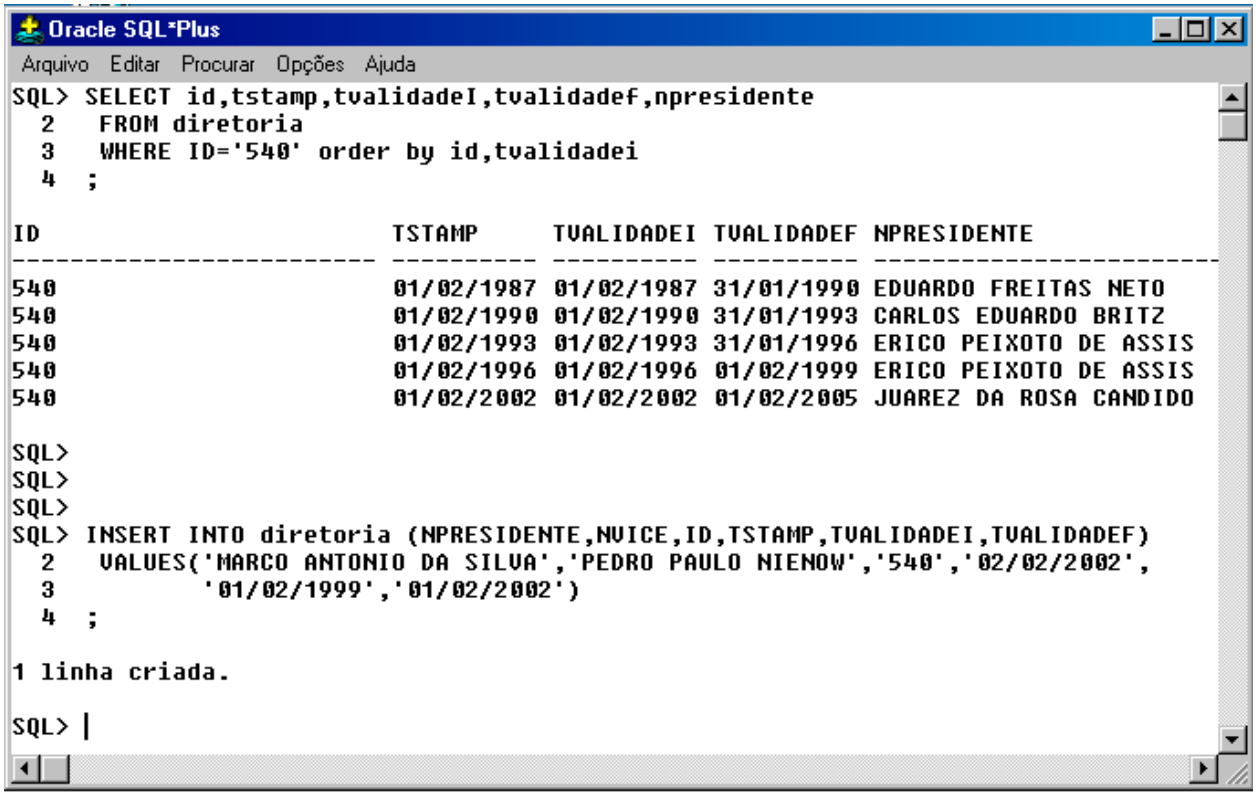
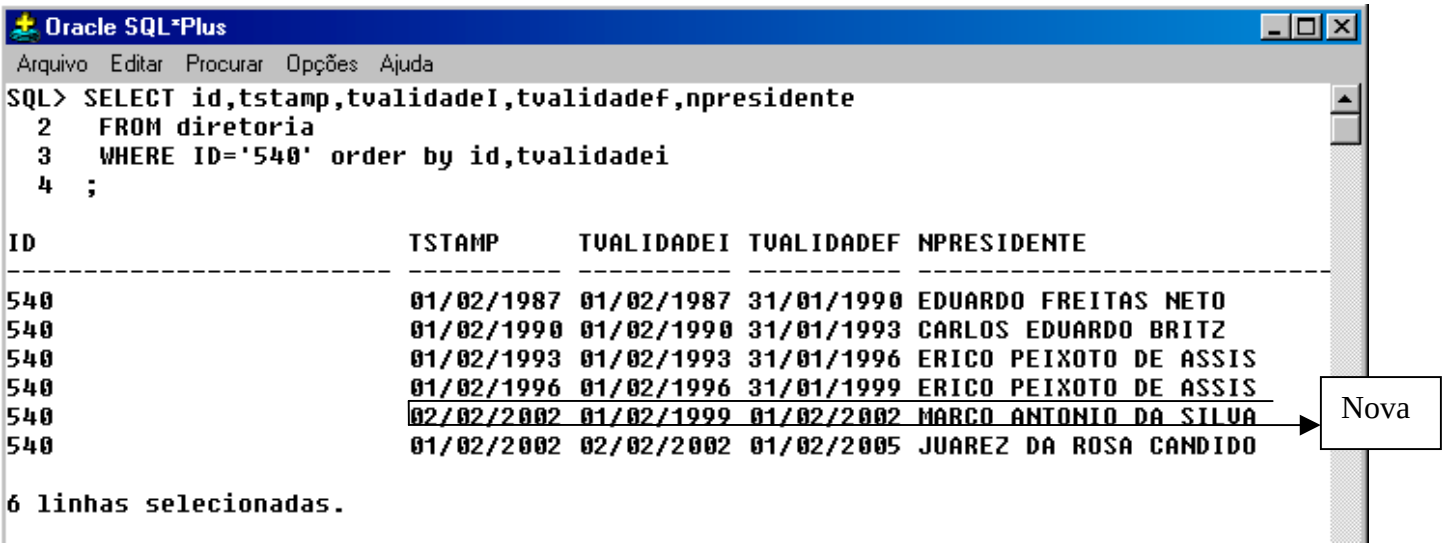


FIGURA 7.11 – Operação de inserção no BDT



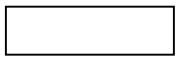


FIGURA 7.12 – Reação do PTV a uma operação de inserção

A reação do PTV à operação apresentada na figura 7.11 é mostrada na figura 7.12. Para esta situação, o PTV alterou os valores de *TVF* da tupla anterior e de *TVI* da tupla que sucede a nova. Os valores foram alterados conforme as regras implementadas, diminuindo uma unidade temporal para o valor de *TVF* e aumentado de uma unidade temporal para o valor de *TVI*.

A figura 7.13 mostra a execução no Oracle de uma consulta ao BD utilizando a função *PERIODO*, apresentada na seção 6.5.2. Neste exemplo deseja-se: "*Recuperar os componentes da diretoria que terminam o mandato em dez/2002 e que iniciaram o mandato no período que vai de jan/1999 à dez/2000*".

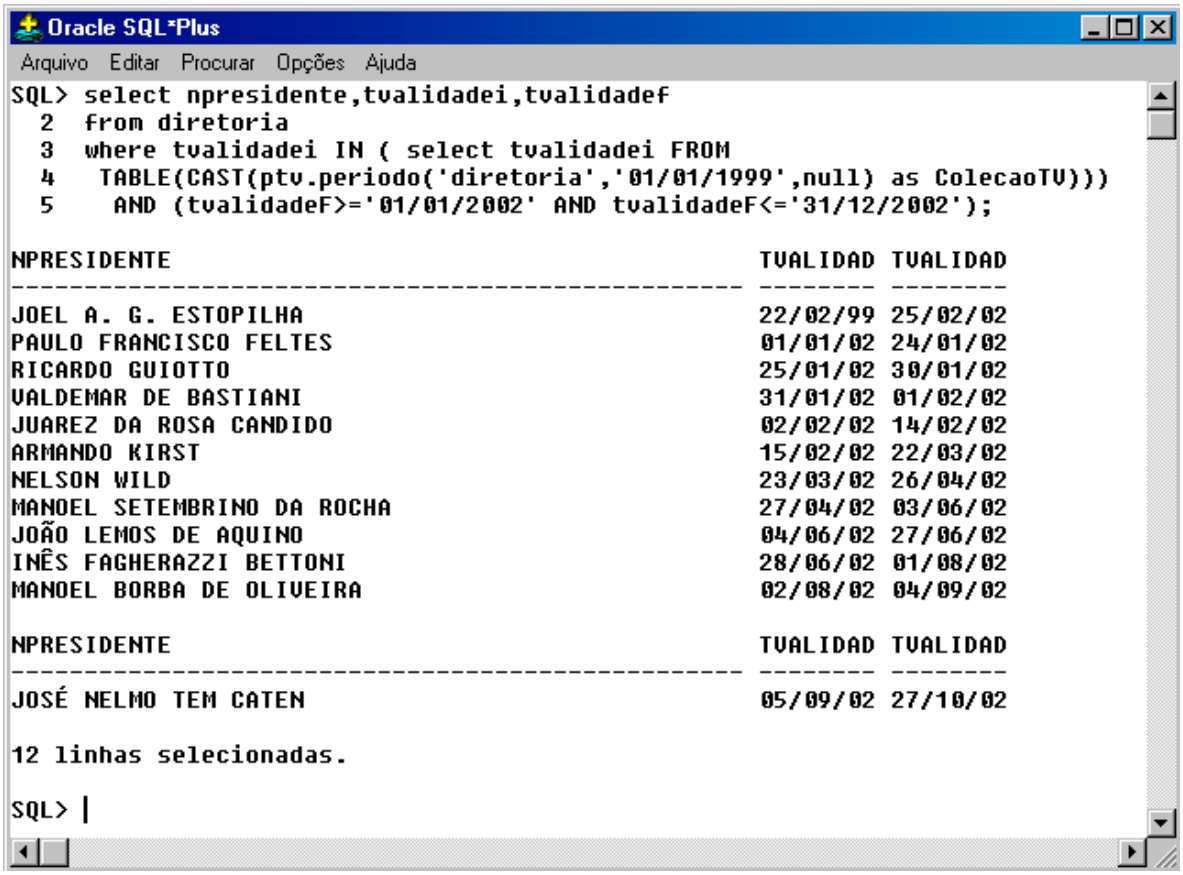


FIGURA 7.13 – Oracle. Execução da função PERIODO

8 Conclusão

O SGBD Oracle na versão 8i permite que informações possam ser associadas a um ponto no tempo, de forma similar a um banco de dados de tempo de transação. Entretanto, não houve mudanças na estrutura do SGBD para que o tempo pudesse ser inserido no BD. O mecanismo de controle temporal utiliza-se dos recursos já conhecidos como funções, procedimentos, gatilhos e objetos, para inserir, administrar e otimizar a consulta de informações temporais. A forma como o TSC é empregado, a complexidade do uso, a limitação de recursos e a falta de conceitos temporais o tornam vulnerável a falhas tanto por parte do usuário como da ferramenta.

O PTV foi desenvolvido no intuito de ser utilizado independente ou em paralelo com o TSC, de maneira a permitir a administração de informações temporais como acontece em banco de dados bitemporais. Os conceitos utilizados pelos pacotes são diferentes, enquanto o TSC visa oferecer recursos para facilitar a recuperação de informações associadas ao rótulo temporal de tempo de transação, o PTV busca oferecer mecanismos para que a consistência temporal do banco de dados seja preservada, pois o controle de consistência temporal em BD de tempo de validade é bem maior do que quando é utilizado somente o tempo de transação.

Para implementar o PTV foram analisados os conceitos relativos a BDT e na medida do possível, implementados. As regras para a consistência em BDTs não puderam ser implementadas como um todo, impossibilitando a execução de todas as funcionalidades de BD de tempo de validade. Apesar da dificuldade da implementação do tempo de validade em decorrência da limitação de recursos do SGBD relacional, o objetivo foi alcançado. Foi possível visualizar o limite da implementação de um BD de tempo de validade sobre um SGBD relacional.

De uma forma geral, o PTV pode ser considerado uma ferramenta de apoio à inserção e ao controle de consistência de informações de tempo de validade no SGBD Oracle.

8.1 Perspectivas de Trabalhos Futuros

O objetivo principal deste trabalho era utilizar o PTV em conjunto com o TSC, de forma a possibilitar o controle bitemporal do BD relacional. Mesmo que os pacotes possam ser executados em conjunto, o TSC não corresponde às necessidades de um BD de tempo de transação. Desta forma poderia ser criado um pacote para a inserção e o controle do

tempo de transação no SGBD relacional para que, em conjunto com o PTV, possam efetivamente implementar um BD bitemporal sobre um SGBD relacional.

Outro aspecto que pode ser aprofundado é o estudo e a implementação de recursos de consulta sobre informações temporais em SGBD relacionais. O PTV inclui o protótipo da função *PERIODO* que poderia ser aperfeiçoado, bem como expandido.

Bibliografia

[EDE 94] EDELWEISS, N; OLIVEIRA, J. P. M. de. **Modelagem de aspectos temporais de sistemas de informação**. Recife: Universidade Federal de Pernambuco, 1994. Livro texto da Escola de Computação, 9., 1994,Recife.

[EDE 98] EDELWEISS, N. Bancos de Dados Temporais: Teoria e Prática. In: JORNADA DE ATUALIZAÇÃO EM INFORMÁTICA, 17., 1998, Belo Horizonte. **Anais...**. Belo Horizonte : SBC, 1998. p. 225-282.

[ETZ 98] ETZION, O.; JAJODIA, S.;SRIPADA, E. (Ed.). **Temporal Databases:** Research and Practice. Berlin: Springer – Verlag, 1998.

[HÖP 2001] HÖPPNER, F. Discovery of Temporal Patterns - Learning Rules about the Qualitative Behaviour of Time Series**.** In**:** EUROPEAN CONFERENCE ON PRINCIPLES AND PRACTICE OF KNOWLEDGE DISCOVERY IN DATABASES, 5., 2001. **Proceedings**. [S.l.]: Springer, 2001. p 192-203.

[HUB 2000] HÜBLER, P.N. **Definição de um Gerenciador para o Modelo de Dados TF-ORM**. 2000. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.

[HUB 2000a] HÜBLER, P. N.; EDELWEISS, N. Gerenciamento de Diferentes Tipos de Dados Temporais Implementados em BDs Convencionais. In: CONFERENCIA LATINOAMERICANA DE INFORMATICA, CLEI, 26.,2000, México. **Memórias.** Mexico: Tec. De Monterrey, 2000. 1 CD-ROM.

[JEN 98] JENSEN, C.S. et al. The Consensus Glossary of Temporal Database Concepts. In: ETZION, O. JAJODIA, S.; SRIPADA, S (Ed.). - **Temporal Databases**: Research and Practice. Berlin: Springer, 1998. p.367-405.

[JEN 99] JENSEN, C.S. **Temporal Database Management**. 1999. Dr. Techical the sis. Departament of computer Science, Aalborg University, Aalborg. Disponível em: <http://www.cs.auc.dk/~csj/Thesis/>. Acesso em: 30 out. 2001.

[LAS 2001] LAST, M; KLEIN, Y; KANDEL, A. Knowledge discovery in Time Series Databases. **IEEE Transactions on Systems, Man, and Cybernetics**, New York, v. 31, n. 1, p160-169, Feb. 2001.

[ORA 99] ORACLE 8i: time series user´s guide, Release 8.1.5 A67294-01. Disponível em: <http://technet.oracle.com/doc/inter.815/a67294/toc.htm> Acesso em: 30 out. 2001.

[ÖZS 95] ÖZSOYOGLU, G.; SNODGRASS, R. T. Temporal and real-time databases: a survey. **IEEE Transactions on Knowledge and Data Engineering**, New York, v.7, n.4, p.513, Aug. 1995.

[PER 2000] PERNG, C.; WANG, H.; ZHANG, S.;PARKER,S. A New Model for Similarity-based Pattern Querying in Time Series Databases. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, ICDE, 2000. **Proceedings**, [S.l:s.n.], 2000.

[SNO 85] SNODGRASS, R. T.; AHN, I. A taxonomy of Time in Databases. In: ACM-SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, SIGMOD, 1985, Austin. **Proceedings…** New York: ACM, 1985. p. 235-246.

[SNO 95] SNODGRASS, R: **The TSQL2 Temporal Query Language.** [S.l.]: Kluwer Publishers, 1995.

[SNO 2000] SNODGRASS, R. T. **Developing Time-Oriented Database Aplications in SQL**. San Francisco: Morgan Kaufmann, 200.

[TAN 93] TANSEL, C.G. et al. (Ed.). **Temporal Databases :** theory, design and implementation. Redwood City: Benjamin/Cummings, 1993.

[ZAN 97] ZANIOLO, C. et al. **Advanced Database Systems**. San Francisco: Morgan Kaufmann, 1997