INTERFACE DE CONSULTAS TF-ORM PARA ORACLE

MIRELLA MOURA MORO
NINA EDELWEISS – orientadora
INSTITUTO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
Av. Bento Gonçalves, 9500 – Bloco IV – Agronomia
Caixa Postal 15064 – CEP 91501 – Porto Alegre – RS
[mirella, nina]@inf.ufrgs.br

Resumo

O TF-ORM (*Temporal Functionality in Objects with Roles Model*) é um modelo de dados temporal orientado a objetos que utiliza o conceito de papéis para representar os diferentes comportamentos dos objetos. Como não existe um SGBD para o TF-ORM, optou-se por fazer o mapeamento do modelo para um BD relacional comercial. Com esse mapeamento pronto, o próximo passo é permitir que o usuário execute consultas TF-ORM sobre o BD. Assim como o modelo, a consulta também deve ser traduzida para uma linguagem que permita sua execução sobre um BD relacional, no caso para o SQL (*Structured Query Language*). O objetivo desse trabalho é apresentar uma ferramenta para interface do mapeamento de consultas TF-ORM no banco de dados Oracle. A consulta TF-ORM pode ser lida a partir de um arquivo texto ou escrita pelo usuário no próprio ambiente. O resultado é a consulta em SQL correspondente, a qual é gravada em outro arquivo e mostrada ao usuário. Após o mapeamento, a ferramenta permite sua execução sobre a base de dados desejada.

Palavras-chave: banco de dados, consulta temporal e relacional, mapeamento TF-ORM/SQL.

Abstract

TF-ORM (Temporal Functionality in Objects with Roles Model) is an object oriented temporal data model which uses role concepts to represent the different object behaviors. Because there is no DBMS for TF-ORM, we have chosen to make the mapping from the model to a relational database. Once this mapping is implemented, the next step is to allow the user to perform TF-ORM queries on the database. As the model, the query also needs to be mapped to a language which allows its running on a relational database, in this case to SQL (Structured Query Language). The objective of this work is to present an interface to the TF-ORM query mapping to an Oracle database. The TF-ORM query can be read from a text file or written by the user through the interface. The result is the correspondent SQL query which is recorded to another file and presented to the user. After the mapping is completed, the interface allows to execute the query on the chosen database.

Keywords: database, temporal and relational query, TF-ORM/SQL mapping.

1 Introdução

Bancos de dados temporais permitem armazenar todos os estados de um objeto, registrando sua evolução com o tempo [TAN 93]. Uma vez que a utilização de um modelo de dados temporal não implica, necessariamente, na utilização de um SGBD específico, pode-se usar bancos de dados comerciais através do mapeamento do modelo temporal para o banco em questão. Esse enfoque foi adotado para o modelo TF-ORM (*Temporal Functionality in Objects With Roles Model*) [EDE 94] e [EDE 98], modelo de dados orientado a objetos que utiliza o conceito de papéis para representar diferentes comportamentos dos objetos.

Tendo-se o mapeamento entre o modelo temporal e o banco de dados comercial, o próximo passo é permitir que o usuário faça consultas TF-ORM sobre a base de dados. Assim como o modelo, a consulta também deve ser traduzida para uma linguagem que permita sua execução sobre um BD relacional. A linguagem escolhida foi a SQL por ser a linguagem aceita pela maioria dos bancos de dados comerciais e por possuir padrão ANSI. O mapeamento da consulta TF-ORM para SQL foi realizado baseando-se em [CAR 97].

Esse trabalho apresenta uma ferramenta que realiza o mapeamento de consultas TF-ORM para o banco de dados Oracle. A consulta TF-ORM pode ser lida a partir de um arquivo texto ou escrita pelo usuário via teclado. A ferramenta então realiza as análises léxica, sintática e semântica da consulta e, se a mesma estiver correta, faz o mapeamento. Caso haja algum erro na consulta, o usuário é notificado e o processo é suspenso. O resultado do mapeamento é a consulta em SQL que é gravada em outro arquivo com extensão .SQL e mostrada ao usuário na tela. Também é permitida a execução da consulta SQL sobre a base de dados desejada previamente selecionada. Essa ferramenta foi projetada de modo que sua extensão para a utilização de bases de dados de outros fabricantes (Sybase, SQL Server, ...) fosse possível sem maiores dificuldades, uma vez que o mapeamento é feito para SQL ANSI.

Esse trabalho está dividido da seguinte maneira: na segunda seção é apresentada resumidamente a consulta SQL; na terceira, estão o modelo TF-ORM e suas características gerais; a seção quatro apresenta a linguagem de consulta TF-ORM; a seção seguinte mostra o estudo feito para a realização das análises léxica e sintática da consulta TF-ORM; na sexta seção está a descrição do ambiente de mapeamento de consultas.

2 Consulta SQL

De acordo com o ANSI (*American National Standards Institute*), SQL é uma linguagem para SGBDs relacionais. Sentenças SQL são usadas para atualizar e recuperar informações de um banco de dados. Uma vez que o TF-ORM possui apenas uma linguagem de recuperação de dados, essa apresentação da linguagem SQL está restringida a seu comando de seleção. O modelo básico de uma instrução de consulta em SQL é [TOR 99]:

```
SELECT coluna1 [, coluna2, ...]
FROM   tabela1 [, tabela2, ...]
[WHERE condição];
```

Os nomes que estão na cláusula `SELECT` determinam quais colunas irão compor o resultado. O usuário pode escrever quantas colunas quiser ou colocar o operador "*" para selecionar todas. Pode-se também colocar expressões de soma, subtração, multiplicação e divisão no resultado das consultas, inserindo-as na cláusula `SELECT`. A cláusula `FROM` determina qual ou quais tabelas do banco de dados devem ser consideradas na consulta. A cláusula `WHERE` (opcional) especifica a quais condições as linhas retornadas devem obedecer. As condições podem conter uma série de operadores lógicos. Além desses, o operador `LIKE` permite que o usuário selecione somente as tuplas que seguem o padrão especificado. O SQL permite ainda uma série de facilidades, entre elas: junção de duas tabelas, aninhamento de consultas, funções sobre caracteres, números e datas, funções sob grupo de linhas (`COUNT`, `SUM`, `AVG`, `MAX`, `MIN`) e cláusulas especiais como `EXIST`, `GROUP BY`, `HAVING` e `ORDER BY`.

3 Modelo TF-ORM

TF-ORM (*Temporal Functionality in Objects with Roles Model*) [EDE 94] é um modelo de dados temporal orientado a objetos que utiliza o conceito de papéis para representar os diferentes comportamentos dos objetos. Esse modelo foi criado com o objetivo de armazenar não somente os valores atuais dos dados, mas toda a sua história – valores passados, atuais e previsões de valores futuros.

Dois diferentes conceitos de tempo podem ser identificados em uma aplicação segundo Snodgrass e Jensen, citado por [EDE 94]: *tempo de transação* no qual uma informação foi introduzida no banco de dados; *tempo de validade* no qual a informação modela a realidade. No modelo TF-ORM esse tempos são utilizados como rótulos associados às propriedades que descrevem as instâncias e os atributos que variam com o tempo. O tempo é modelado variando de forma discreta, estando linearmente ordenado.

Com o objetivo de separar os aspectos estáticos dos aspectos dinâmicos, o TF-ORM apresenta o conceito de papéis. O objeto continua a ser instância de uma única classe, mas poderá assumir ao longo de sua existência um ou mais papéis (comportamentos), inclusive diversas instâncias de um mesmo papel simultaneamente.

O modelo apresenta dois tipos de propriedades: **propriedade estática**, mantém seu valor no tempo; **propriedade dinâmica**, pode alterar seu valor no tempo, sendo que todos os valores devem ficar armazenados permitindo a recuperação do histórico de uma instância.

Cada **classe** é definida por um nome e papéis que sua instância pode assumir. São três tipos de classes: de recursos, de processos e de agentes. Cada objeto de classe possui um identificador único que é definido no instante de sua criação e armazenado na propriedade estática pré-definida *oId*. Todo objeto apresenta um **papel básico** que descreve as características iniciais de um instância e as propriedades globais que controlam sua evolução. A criação de um objeto de uma classe instancia automaticamente o seu papel básico que, ao contrário dos demais papéis, só pode ser instanciado uma vez.

Cada **papel** é representado por um nome, propriedades, estados possíveis para as instâncias desse papel, mensagens que podem ser enviadas ou recebidas, regras de transição de estado e restrições de integridade. Cada papel também possui um identificador único armazenado na propriedade estática pré-definida *rId*.

Além das propriedades estáticas pré-definidas *oId* e *rId*, o modelo TF-ORM apresenta também propriedades dinâmicas pré-definidas: *object_instance* (intervalos de tempo em que a instância está ativa); *end_object* (instante de tempo em que a instância foi descontinuada); *role_instance* (eventuais suspensões e reativações da instância do papel); *end_role* (tempos de transação e validade do final da vida da instância de um papel).

4 Linguagem de Consulta TF-ORM

A linguagem de consulta do modelo TF-ORM baseia-se na linguagem SQL e apresenta extensões para suportar os aspectos temporais [EDE 94]. A estrutura geral do comando SQL é mantida: cláusulas de especificação (`SELECT`), de identificação (`FROM`) e de busca (`WHERE`). A linguagem TF-ORM apresenta duas formas alternativas para a consulta:

```
SELECT  <cláusula de especificação>
FROM    <cláusula de identificação>
[ { WHERE | WHEN }  <cláusula de busca> ]
[ AS ON <cláusula de instante temporal> ]
```

A cláusula de especificação define a saída da consulta: de dados, temporal ou mista. As saídas temporal e mista são obtidas apenas com a cláusula `WHEN`, pois a cláusula `WHERE` recupera apenas os dados da base atual (válidas hoje). Quando a cláusula de especificação contiver uma data de transação, uma data de validade ou período, a saída será temporal. Essas informações temporais estão associadas a cada uma das propriedades dinâmicas do modelo, já as propriedades estáticas são válidas enquanto a instância estiver ativa, por isso não possuem data associada. Na cláusula de identificação são especificados as classes e papéis sobre os quais se deseja recuperar informações. Na cláusula de busca estão as condições que devem ser satisfeitas pelos objetos recuperados, podendo ser identificada pelas cláusulas `WHERE` ou `WHEN`.

A cláusula `WHEN` permite a recuperação de toda a história do banco de dados (presente, passado e futuro). Nesse caso, duas situações podem ocorrer: a cláusula de busca não possui nenhuma restrição temporal, ou seja, serão analisadas todas as informações do banco de dados; a cláusula de busca contém as palavras `DATE`, `PERIOD`, `TRANSACTION_DATE` e/ou `TRANSACTION_PERIOD` e serão analisadas apenas as informações que satisfazem a restrição temporal da cláusula de busca.

A cláusula de instante temporal (`AS ON`) define a data correspondente a uma história passada, mudando o referencial de tempo da consulta para o momento definido e utilizando como base as informações conhecidas naquele instante. O modelo TF-ORM possui também operadores temporais que podem ser utilizados na cláusula de busca e determinam o momento do tempo no qual a expressão deve ser avaliada. Além desses, existem predicados e funções temporais que podem ser utilizados na cláusula de busca.

5 Análises Léxica e Sintática

Um compilador é uma ferramenta que lê um programa escrito em uma linguagem (*fonte*) e o converte para um programa equivalente em outra linguagem (*alvo*). Como parte importante do processo, o compilador reporta ao seu usuário a presença de erros. A principal função da ferramenta desse trabalho é servir de "tradutor" de consultas TF-ORM para SQL. Nos termos de compilação, a ferramenta recebe o arquivo *fonte* em TF-ORM e o traduz para o arquivo *alvo* em SQL ou retorna os erros encontrados no arquivo *fonte*.

O processo de compilação consumia bastante tempo até Lesk e Johnson publicarem seus artigos sobre o Lex e o Yacc – [LES 75] e [JOH 75]. Esses dois utilitários simplificaram enormemente a tarefa de construir um compilador [NIE 98]. Essas ferramentas geram um componente de análise sintática tendo como base uma gramática especificando a sintaxe da linguagem. O Lex gera código para o analisador léxico, ou *scanner*, e o Yacc gera código C para o analisador sintático, ou *parser*.

Novas pesquisas têm sido feitas nessa área desde então. Um dos resultados é o PRECCX (*PREttier Compiler Compiler eXtended*), um tradutor que converte um *script* de definição de gramática em código C ANSI [BRE 94]. O código de saída pode ser compilado em qualquer compilador ANSI. Ele prolonga as facilidades do Yacc do Unix e dos outros equivalentes para PC, incluindo aqueles baseados no Bison da GNU. Entre elas: *lookahead* infinito em vez do *lookahead* de um *token* do Yacc; compilação e lincagem incrementais de *scripts* arranjados em módulos separados; suporte para descrições de BNFs extendidas; definição de gramática parametrizadas, permite gramáticas dependente de contexto; variáveis que servem para a gramática como parâmetros, permitindo "macros" para substituir construções gramaticais repetitivas; permite o uso de colchetes para representar segmentos opcionais da gramática. Outra vantagem do PRECCX é o tamanho do arquivo de entrada aceito: o Yacc tem limitação de 255 caracteres enquanto o PRECCX aceita arquivos maiores.

A desvantagem do PRECCX em relação ao Yacc é que, até o presente momento (versão 2.42), não há equivalência para as declarações de precedência e associatividade existentes no Yacc. Em vez disso, é necessário codificar explicitamente a ordem requerida.

6 Descrição do Ambiente

Implementação: a interface do sistema foi implementada utilizando-se a ferramenta para desenvolvimento de aplicações Delphi. A versão de banco de dados utilizada na implementação foi o *Personal Oracle 7.3 for Windows 95*, pois possibilita o manuseio de uma base de dados relacional com a flexibilidade, a escalabilidade e as características de multiusuário providas pelo Oracle 7.3, com a vantagem de não exigir muito do sistema.

Características: a principal função da ferramenta é servir de interface para consultas TF-ORM em um banco de dados Oracle. Adicionalmente, a ferramenta pode apenas verificar se a consulta TF-ORM está correta léxica, sintática e semanticamente, como também executar a consulta sobre a base de dados e mostrar o resultado em forma de tabela. O ambiente realiza o mapeamento baseado no banco de dados desejado, que pode ser qualquer um que esteja registrado no ODBC com driver Oracle. A consulta TF-ORM pode ser lida a partir de um arquivo texto ou escrita pelo usuário via teclado.

A ferramenta também informa se o arquivo contem erros e, na maioria das vezes, relata o tipo e onde ele ocorreu. O resultado do mapeamento, a consulta em SQL, é gravado em outro arquivo com extensão .SQL e mostrado ao usuário na tela. Com o arquivo SQL, o usuário pode executar a consulta na base de dados e a ferramenta retorna os resultados.

Desde o princípio, fundamentou-se que a consulta SQL resultante do mapeamento deveria seguir o padrão ANSI [CAR 97] para poder ser executada em qualquer banco de dados relacional comercial. Desse modo, a linguagem utilizada na consulta TF-ORM deve ser aceita pelo banco de dados evitando confusões com o conjunto de caracteres para suporte de linguagem adotado pelo mesmo. Assim, a consulta TF-ORM de entrada não pode apresentar acentuação ou a letra "ç" em qualquer uma das cláusulas.

Durante a realização do mapeamento, a ferramenta faz consultas às tabelas do sistema do banco de dados que o usuário está usando. Como os nomes das tabelas de sistema, bem como seus atributos, variam entre os produtos e nessa versão foi usada uma base Oracle para realizar o mapeamento, o usuário deve escolher uma base de dados que também seja Oracle para poder realizar a tradução de TF-ORM para SQL com sucesso. Isso é uma exigência, sendo que a ferramenta nem permite a conexão a um banco de dados que não tenha driver Oracle configurado no ODBC.

Para evitar problemas de versões de arquivos diferentes, estipulou-se que o arquivo com a consulta TF-ORM deve estar salvo antes de realizar o mapeamento. Caso o usuário faça qualquer modificação no arquivo, ele deverá salvá-lo (com o mesmo nome ou não) para poder realizar a tradução ou a verificação da consulta. A figura 1 mostra como ficou a interface final.

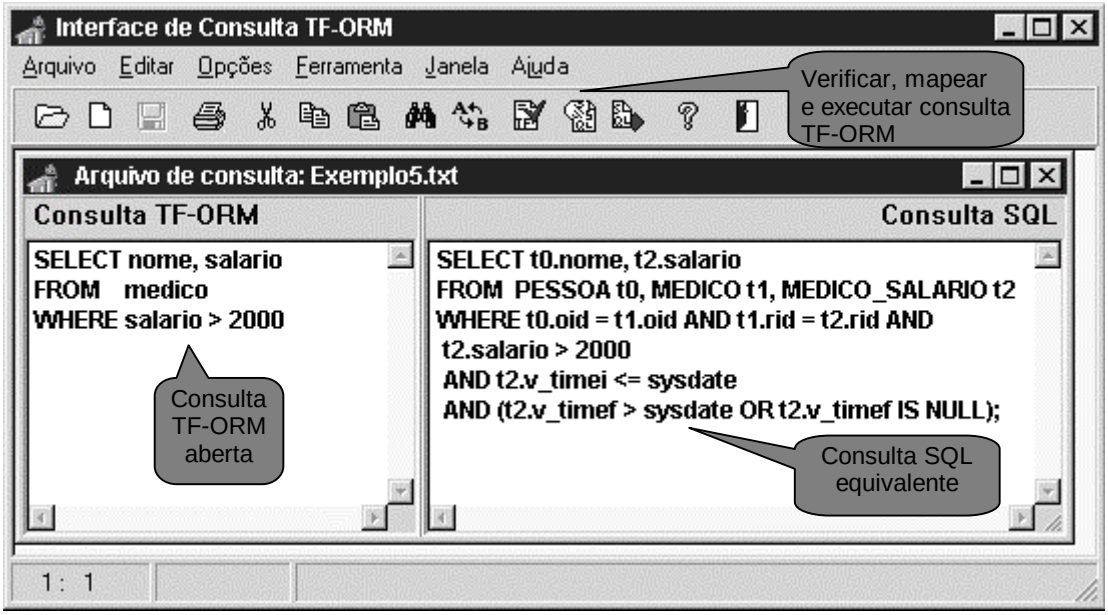

Figura 1 – Interface de consultas TF-ORM para Oracle

A tela principal apresenta um menu, uma barra de botões com as principais opções do menu e uma área para as consultas. A maioria dos comandos do menu apresenta teclas de atalho. Os botões de Verificação e Mapeamento, bem como seus comandos no menu, são habilitados quando há alguma consulta TF-ORM aberta. Já o botão e o comando de Executar a Consulta é habilitado quando existe uma consulta SQL aberta, como mostra a figura 1.

Funcionamento: ao entrar na ferramenta, o usuário deve escolher qual base de dados Oracle será considerada durante a verificação ou mapeamento da consulta TF-ORM. O usuário pode abrir um arquivo contendo a consulta TF-ORM ou digitá-la na parte esquerda da ferramenta. Quando a consulta aberta pelo usuário já possui uma consulta SQL equivalente gravada, ou seja, com o mesmo nome da consulta TF-ORM com extensão .SQL, a ferramenta também a mostra no espaço da direita (figura 1).

Caso o usuário queira apenas verificar se o arquivo contém uma consulta TF-ORM, após o clique no menu ou no botão de Verificação da Consulta, a ferramenta realiza as análises léxica, sintática e semântica. Após, pode-se realizar o mapeamento, sendo que em ambos os casos uma janela é aberta mostrando o andamento do processo. Caso o mapeamento ocorra com sucesso, a ferramenta apresenta ao usuário a consulta SQL resultante. Desse modo, o comando e o botão de Execução da Consulta ficam habilitados. O usuário tem a opção de salvar a consulta SQL no menu Arquivo. Ele pode salvar com o mesmo nome do arquivo da consulta TF-ORM ou com outro. A consulta SQL também pode ser executada no banco de dados. Após sua execução, a ferramenta mostra uma tela com as tuplas resultantes da consulta.

Verificações e erros: a primeira verificação que a ferramenta faz é se a consulta TF-ORM apresenta as cláusulas `SELECT`, `FROM` e `WHERE` ou `WHEN`. A segunda verificação é feita pelo compilador gerado pelo PRECCX com as análises léxica e sintática, retornando a linha e o símbolo onde ocorreu o erro. A terceira verificação é a análise semântica executada pela própria ferramenta. Essa análise verifica se as tabelas e atributos da consulta existem no banco de dados. Caso o erro seja no parâmetro de alguma função ou predicado, a tela de andamento apresenta a propriedade que não pertence à base de dados e o nome da função na qual é mencionada. Em todos os casos de erro, o mapeamento é cancelado.

Gramática: a gramática usada como entrada no PRECCX foi escrita baseada em [CAR 97]. Foi feito um estudo para adequá-la ao formato de entrada do compilador. Entre as modificações e ajustes estão: modificação dos operadores de comparação de ≥, ≤ e ≠ para >=, <= e <>, seguindo as normas do padrão ANSI para SQL; visto que os BDs comerciais aceitam tabelas e colunas cujos nomes começam com sublinhado ("_"), mudou-se o identificador para que aceite como primeiro caracter uma letra ou um sublinhado; exclusão da recursividade à esquerda; redefinição de *property value* para aceitar números inteiros.

7 Conclusão

Esse trabalho apresentou uma ferramenta de interface de consultas TF-ORM para base de dados Oracle. A consulta TF-ORM pode ser lida a partir de um arquivo texto ou escrita pelo usuário via teclado. O resultado é a consulta SQL correspondente, a qual é gravada em outro arquivo e mostrada ao usuário. Após o mapeamento, a ferramenta permite sua execução sobre a base de dados desejada.

Como principal resultado desse trabalho, tem-se a consulta TF-ORM mapeada para SQL/ANSI, que pode ser executada em qualquer BD relacional comercial que aceite o padrão. No caso específico do protótipo implementado, o SGBD TF-ORM deve estar mapeado para um banco de dados Oracle. Como resultado adicional, tem-se a possibilidade de apenas verificar se a consulta TF-ORM é válida. Um compilador gerado pelo PRECCX executa as

análises léxica e sintática da consulta TF-ORM. Se não ocorrerem erros, a ferramenta realiza a análise semântica da consulta. Uma tela de mensagens é apresentada com o andamento da verificação e com os erros encontrados na consulta.

Para trabalhos futuros estão a extensão do protótipo para que aceite mais de uma classe e papel sendo consultados. Isso implicaria no estudo da sintaxe do TF-ORM para aceitar tal modificação. Também seria interessante modificar a ferramenta para que aproveitasse mais dos recursos do ambiente de desenvolvimento Delphi e do SGBD Oracle. Como sugestão, estão a possibilidade de ter um ambiente cliente/servidor, com acesso remoto e multiusuário.

Além disso, o protótipo foi planejado de modo que extensões para outras bases de dados, como Sybase e SQL Server, pudessem ser feitas sem maiores dificuldades. Todas as consultas às tabelas do sistemas são realizadas logo que o usuário seleciona o BD. As informações que interessam, nome das tabelas e de suas colunas, são armazenadas em variáveis globais. O único trabalho que se teria para mudar de base, seria o de adaptar o nome das tabelas e atributos do sistema que a ferramenta consulta.

Bibliografia

[AHO 86] AHO, Alfred V.; SETHI, Ravi. **Compilers – Principles, Techniques, and Tools**. Stanford: Addison-Wesley Publishing Company, 1986. 796 p.

[BRE 94] BREUER, Peter T.; BOWEN, Jonathan. P. **PRECCX User Manual**. Oxford, Reino Unido, 1994. Disponível por WWW em http://www.comlab.ox.ac.uk/archive/redo/precc/user-manual.html (12/04/1999).

[CAR 97] CARVALHO, Tanisi. P. **Implementação e Consultas para um Modelo de Dados Temporal Orientado a Objetos**. Porto Alegre: CPGCC – UFRGS, 1997. Dissertação de Mestrado.

[EDE 94] EDELWEISS, Nina**. Sistemas de Informações de Escritórios: um modelo para Especificações Formais**. Porto Alegre: CPGCC - UFRGS, 1994. Tese de doutorado.

[EDE 98] EDELWEISS, N. Bancos de Dados Temporais: Teoria e Prática. In: XVII Jornada de Atualização em Informática; XVIII Congresso Nacional da Sociedade Brasileira de Computação, 1998, Recife. **Anais**... Recife: Sociedade Brasileira de Computação, 1998. p.225-282.

[JOH 75] JOHNSON, Stephen C. **Yacc: Yet Another Compiler Compiler**. Technical Report 32, New Jersey, United States of America. Bell Laboratories, Murray Hill, 1975. Disponível por WWW em http://members.xoom.com/thomasn/y_yacc.pdf (12/04/1999).

[LES 75] LESK, M. E.; SCHMIDT, E. **Lex – A Lexical Analyzer Generator**. Technical Report 39, New Jersey, United States of America. Bell Laboratories, Murray Hill, 1975. Disponível por WWW em http://members.xoom.com/thomasn/y_lex.pdf (12/04/1999).

[NIE 98] NIEMANN, Thomas. **A Guide to Lex & Yacc**. Portland, Oregon. Disponível por WWW em http://members.xoom.com/thomasn/y_man.htm (12/04/1999).

[TAN 93] TANSEL, Clifford G.; et al. (editores). **Temporal Databases - theory, design and implementation**. Redwood City: The Benjamin/Cummings Publishing Company, Inc., 1993.

[TOR 99] TORRES, Frank. **SQL Tutorial & Interpreter w/ Live Practice Database**. Disponível por WWW em http://torresoft.netmegs.com/ (10/04/1999).