UNIVERSIDADE DO EXTREMO SUL CATARINENSE – UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

ROGER GOULART DO NASCIMENTO

FERRAMENTA PARA APOIO A MODELAGEM UTILIZANDO ENGENHARIA REVERSA EM BACOS DE DADOS RELACIONAIS POR MEIO DO MAPEAMENTO DE METADADOS CONTIDOS NO SCRIPT SQL

CRICIÚMA, NOVEMBRO DE 2009

ROGER GOULART DO NASCIMENTO

FERRAMENTA PARA APOIO A MODELAGEM UTILIZANDO ENGENHARIA REVERSA EM BACOS DE DADOS RELACIONAIS POR MEIO DO MAPEAMENTO DE METADADOS CONTIDOS NO SCRIPT SQL

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientador: Prof. MSc. Gustavo Bisognin

CRICIÚMA, NOVEMBRO DE 2009

ROGER GOULART DO NASCIMENTO

FERRAMENTA PARA APOIO A ENGENHARIA REVERSA EM BACOS DE DADOS RELACIONAIS POR MEIO DO MAPEAMENTO DE METADADOS CONTIDOS NO SCRIPT SQL

Submetido ao corpo docente do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Profa. MSc. Ana Claudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Prof. MSc. Gustavo Bisognin (UNESC)
Orientador

Prof. MSc. Daniel Pezzi da Cunha (UNICRUZ)

Esp. Marcelo Mazon (Depto de TI - UNESC)

AGRADECIMENTOS

Agradeço primeiramente aos meus pais por todo carinho e dedicação.

Agradeço também a minha irmã Sabrina e seu marido Ricardo por todo apoio e incentivo nos momentos mais difíceis sempre atenciosos compartilhando suas experiências comigo.

Muito Obrigado a todos que de alguma forma cooperaram para a realização deste trabalho!

RESUMO

O avanço dos sistemas computacionais se estendeu aos mais diversos setores nas ultimas décadas sob o advento de digitalizar tarefas cotidianas dentro de organizações, trazendo uma nova visão de como organizar os dados, como resposta a esta visão foram criados os banco de dados e sistemas para gerenciá-los. Porém tal evolução também trouxe o problema de como modelar e gerir os dados de forma eficiente e bem estruturada, com o intuito de sanar esta carência foram propostas diversas metodologias para o projeto de banco de dados. No entanto nem todos os desenvolvedores de soluções respeitam tais metodologias, em muitos casos são omitidos ou ignorados certas etapas do projeto, fato que gera grandes problemas em futuras manutenções. Tendo em vista o alto custo para o desenvolvimento de sistemas de base de dados, torna-se imprescindível a execução de manutenções, entretanto sem a devida modelagem esta tarefa pode vir a ser inviável. Dessa forma foi utilizado o processo de engenharia reversa em banco de dados com o intuito criar uma modelagem adequada, permitindo um auxílio a manutenção por meio da geração de um diagrama Entidade Relacionamento atualizado em relação ao banco de dados.

Palavras-chave: Banco de Dados, Engenharia Reversa, Engenharia de *Software*, SQL.

ABSTRACT

The advance of computer systems has been extended to several sectors in recent decades, in the advent of computerizing daily tasks within organizations. This advance generated a new vision of how to organize data. In response to this view, databases and systems to manage these databases were created. However, this development also brought the problem of how to model and manage data in an efficient and well structured way. In order to remedy this deficiency, various methodologies for database design have been proposed. Nevertheless, not all developers of solutions follow these methodologies. In many cases, some projects` steps are omitted or ignored, creating major problems in future maintenance. Considering the high cost of system development database, maintenance gets essential for companies. But, without proper modeling, this task can become impossible. Thus, it was used the reverse engineering process in a database in order to model systems. The main goal is to auxiliate the maintenance through the creation of an Entity Relationship Diagram updated on the database.

Keywords: database, reverse engineering, software engineering, SQL

LISTA DE ILUSTRAÇÕES

Figura 1. Ciclo de vida do Banco de Dados.....15

Figura 2. Representação de um Relacionamento.....25

Figura 3. Relacionamento de um-para-muitos (1:N).....26

Figura 4. Diagrama de um conjunto (1:N).....26

Figura 5. Relacionamento de muitos-para-muitos (N:N).....27

Figura 6. Diagrama de um conjunto (N:N).....27

Figura 7. Relacionamento de um-para-um (1:1).....28

Figura 8. Diagrama de um conjunto (1:1).....28

Figura 9. Diferença entre conectividade e cardinalidade.....29

Figura 10. G/E total..... 30

Figura 11. G/E parcial..... 31

Figura 12. Representação de um relacionamento de N:N.....40

Figura 13. Representação de uma especialização.....40

Figura 14. Representação de uma entidade.....41

Figura 15. Representação dos atributos.....42

Figura 16. Representação de um atributo identificador.....42

Figura 17. Caso de uso do sistema proposto.....45

Figura 18. Diagrama de classe do sistema proposto.....47

Figura 19. Diagrama de máquina de estados.....49

Figura 20. Obtenção do script SQL..... 51

Figura 21. Conjunto do Alfabeto Estipulado..... 52

Figura 22. Alfabeto estipulado..... 52

Figura 23. Relação de tabelas interpretadas..... 56

Figura 24. Relação de atributos interpretados..... 57

Figura 25. Relação de relacionamentos interpretados.. 57

Figura 26. Relação de identidades geradas..... 57

Figura 27. Relação de relacionamentos geradas..... 58

LISTA DE ABREVIATURAS E SIGLAS

BD	Banco de Dados
CASE	*Computer-Aided Software Engineering*
DBA	*Database Administrator*
DDL	*Data Definition Language*
DER	Diagrama Entidade Relacionamento
DER-E	Diagrama Entidade Relacionamento Estendido
ER	Entidade Relacionamento
ER-E	Entidade Relacionamento Estendido
ERBD	Engenharia Reversa em Banco de Dados
G/E	Generalização/Especialização
HD	*Hard Disk*
JDBC	*Java Database Connectivity*
JDK	*Java Development Kit*
MDC	*Multidimensional clustering*
OMT	*Object Modelling Technique*
SGBD	Sistema Gerenciador de Banco de Dados
SIG	Sistemas de Informação Geográfica
SQL	*Structured Query Language*
SWT	*Stadard Widget Toolkit*

SUMÁRIO

1 INTRODUÇÃO ……..….10

1.1 OBJETIVO GERAL ……...……...……11

1.2 OBJETIVOS ESPECIFICOS……...……...11

1.3 JUSTIFICATIVA……..……...……....…..11

1.4 ESTRUTURA DO TRABALHO……..…….....…12

2 PROJETO DE BANCO DE DADOS…….14

2.1 ETAPAS DO PROJETO DE BANCO DE DADOS……..17

2.1.1 Projeto Conceitual……...…….…..18

2.1.2 Projeto Lógico……...…..……19

2.1.3 Projeto Físico……...…..…….…….20

3 ENTIDADE RELACIONAMENTO……..22

3.1 ENTIDADES……..…22

3.2 ATRIBUTOS……..…22

3.2.1 Atributos Chave.……...…..……23

3.3 ENTIDADE FRACA……..…24

3.4 RELACIONAMENTOS…….24

3.4.1 Classificação e Cardinalidade de Relacionamentos Binários……...…..……25

3.4.1.1 Relacionamento Um-para-muitos…….26

3.4.1.2 Relacionamento Muitos-para-muitos……...….27

3.4.1.3 Relacionamento Um-para-um……...28

3.5 MODELO ENTIDADE RELECIONAMENTO ESTENDIDO…….29

3.5.1 Generalização / Especialização (G/E) ……...…….…..30

3.5.2 Herança.……..……...…..……31

3.5.3 Classe……..……...…..……32

3.5.3.1 Subclasse…….……...…32

3.5.3.2 Superclasse…….……...…32

4 ENGENHARIA REVERSA……..……...…..……..…34

4.1 ENGENHARIA REVERSA EM BANCO DE DADOS……...…34

4.2 METODOLOGIA UTILIZADA NA EXECUÇÃO DA ENGENHARIA REVERSA EM BANCO DE DADOS…….……...….35

4.2.1 Construção do Modelo ER……..……...…….……..…39

4.2.2 Identificação de Relacionamentos……...…….……..…..41

4.2.3 Definição de Atributos……..……...…….……..….41

4.2.4 Definição de identificadores de entidade……...…….....….42

5 TRABALHOS CORRRELATOS……..……...…..…….....43

6 SISTEMA DE ENGENHARIA REVERSA DE BANCO DE DADOS……...…..……...45

6.1 MODELAGEM DE SISTEMA…….……..…..….45

6.1.1 Modelagem de Casos de Uso……..……...…..…….....45

6.1.2 Modelagem de Classes……..……...…….……..….46

6.1.3 Modelagem de Máquina de Estados…..……...…..……..….49

6.2 COMPONENTES DO SISTEMA…..……..……...…50

6.2.1 Arquivo de Entrada (SQL) ..……..……...…..……..….51

6.2.1 Módulo de Controle……..……..……..……...…..……..….49

6.2.3 Módulo Entidade……..……..……..……..……...…..……...…….52

6.2.4 Módulo de Engenharia Reversa…..……..……..……...……...……...…….52

6.2.5 Módulo Gerador……..……..……..……..……...…..……...…….54

6.3 RECURSOS UTILIZADOS…….……..……...…54

6.3 ANÁLISE DOS RESULTADOS OBTIDOS…….……...….56

1 INTRODUÇÃO

Tendo em vista o grande avanço das sociedades orientadas a informação, gerenciar dados tornou-se fundamental para as organizações desenvolvedoras de soluções baseadas em sistemas computacionais. Dessa forma, os Sistema Gerenciador de Banco de Dados (SGBD) são amplamente utilizados para gerir a informação. Entretanto, o uso de banco de dados sem o devido conhecimento de modelagem de dados traz como problema a falta de organização entre os dados de forma eficiente e lógica (CHEN, 1990).

Devido a necessidade de uma modelagem de dados correta é justificado o uso do projeto de banco de dados, pois ele contempla as principais necessidades na projeção de um banco de dados tais como: modelagem conceitual, modelagem lógica e modelagem física.

No entanto, equipes de programadores costumam implementar bancos de dados sem o necessário planejamento e modelagem. Além disto, muitas vezes o projeto não é atualizado de acordo com as mudanças que ocorrem no banco de dados, fato que torna o modelo conceitual obsoleto.

O presente trabalho propõe um método para manter a modelagem na forma de Diagrama Entidade Relacionamento Estendido (DER-E) atualizada com as alterações feitas no banco de dados, tendo como diferencial, entre outras ferramentas comerciais, a adição da cardinalidade entre os relacionamentos dos DER-E bem como a composição semi-automática do modelo lógico e o apoio direto a engenharia reversa de bancos de dados de sistemas existentes proporcionando um ganho de produtividade em possíveis conversões de bancos de dados.

1.1 OBJETIVO GERAL

Elaborar uma ferramenta para apoio à modelagem utilizando engenharia reversa de banco de dados relacionais por meio do mapeamento de metadados a partir do *script Structured Query Language* (SQL) para múltiplas bases de dados.

1.2 OBJETIVOS ESPECIFICOS

a) mapear o *script* SQL para extração de metadados;
b) possibilitar a engenharia reversa de múltiplas bases de dados;
c) construir um analisador léxico que identifique entidades e relacionamentos com base no modelo Entidade Relacionamento (ER).

1.3 JUSTIFICATIVA

Com a popularização acentuada dos bancos de dados relacionais, a vasta implantação e utilização de banco de dados tornaram-se evidente em diversas organizações. A abordagem de modelagem Entidade Relacionamento (ER) passou a ser utilizada nas mais diversas linhas de produção de sistemas ou produtos relacionados a tecnologia da informação.

Como apoio direto a este paradigma do mercado, surgiram diversas ferramentas que permitem a criação de bancos de dados, normalmente em notação SQL, de forma simples e bastante rápida. Contudo, em uma grande parte dos casos, o planejamento é precário e muitas vezes não é realizado acarretando na ocorrência de erros de implantação causando um impacto direto nas futuras manutenções do sistema.

Este trabalho propõe a implementação de uma ferramenta para a conversão semi-automática do modelo lógico para o modelo conceitual no formato ER.

Os principais diferenciais desta ferramenta são:

a) a proposição de manter o modelo ER atualizado em relação ao banco de dados por meio da engenharia reversa;
b) desenvolvimento de um gerador de modelos ER;
c) indicação semi-automática de possíveis cardinalidades baseadas no modelo lógico especificado;
d) forma de tratamento dos dados para a conversão do modelo.

1.4 ESTRUTURA DO TRABALHO

O texto deste trabalho está estruturado em seis capítulos.

No Capítulo 2 é apresentado o Projeto de Banco de Dados (BD), onde são abordados os seguintes tópicos: a importância do projeto, possíveis problemas que ocorrem na falta de um planejamento adequado, o ciclo de vida de um BD e as três principais etapas do projeto de BD.

O Capítulo 3 corresponde à teoria do modelo Entidade Relacionamento, onde são definidos os conceitos de entidade e relacionamentos, e suas características tais como atributos, atributos chave, cardinalidade, conectividades, entre outras. Além disto, também é abordado o modelo Entidade Relacionamento Estendido.

No Capítulo 4 são apresentadas algumas metodologias de engenharia reversa em banco de dados e possíveis problemas decorrentes da falta de uma metodologia.

No Capítulo 5 são abordados os trabalhos correlatos que fazem uso da engenharia reversa em bancos de dados.

No Capítulo 6 é apresentado o sistema proposto, bem como a metodologia utilizada para sua construção e sua respectiva modelagem, também são apresentados os resultados obtidos e considerações finais.

2 PROJETO DE BANCO DE DADOS

Este capítulo apresenta o Projeto de Banco de Dados, onde são abordados os seguintes tópicos: a importância do projeto, possíveis problemas que ocorrem na falta de um planejamento adequado, o ciclo de vida de um BD e as três principais etapas do projeto de BD.

Um bom projeto é um ponto fundamental para o sucesso de um BD, visto que o projeto que irá possibilitar um melhor entendimento do problema ao qual se deseja resolver, satisfazendo as necessidades básicas do negocio através da modelagem de abstrações do mundo real.

A real importância de um planejamento é visível ao se analisar os problemas causados por um BD construído sem um devido projeto. Para Hernandez (2003, tradução nossa) a maioria dos problemas no nível de dados tal como falta de dados, dados incorretos, dados repedidos e informações imprecisas acontece pela falta de conhecimento da teoria de modelagem de BD por parte do desenvolvedor, que constrói projetos pobres sem respeitar a devida metodologia de planejamento.

Para entender melhor a metodologia para a construção de um banco de dados é preciso primeiramente compreender o ciclo de vida de um BD Muller (1999, tradução nossa). Na Figura 1 é apresentado o ciclo de vida de um BD.

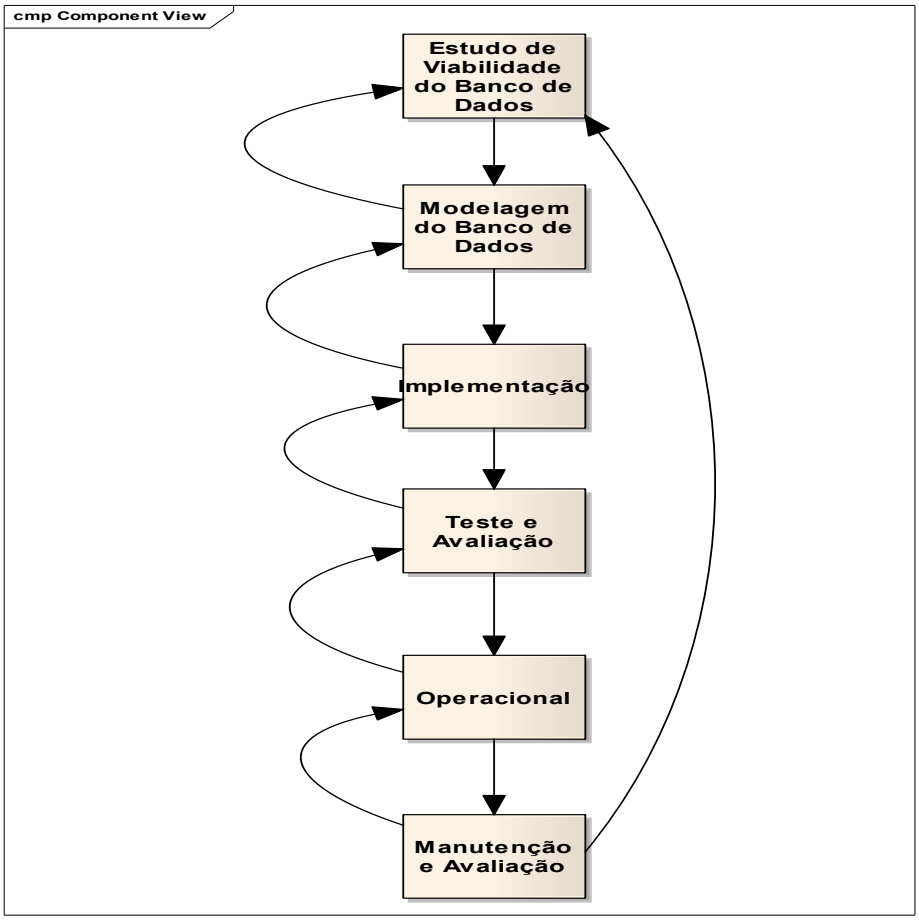


Figura 1. Ciclo d vida do Banco de Dados
Fonte: Adaptado de ROB, P; CORONEL, C (2005, tradução nossa)

O ciclo de vida de um BD é composto por pequenos ciclos redundantes como mostra a Figura 1. Basicamente cada fase completa o sentido de uma fase anterior, cada interação completa corresponde a um banco de dados concluído e caso em algum momento da implementação de uma fase se constate algum problema é possível retornar as fases anteriores e corrigir (MULLER, 1999, tradução nossa).

a) na fase de Estudo de Viabilidade do BD é analisada a situação atual da empresa, são definidos os problemas e restrições, bem como, são estabelecidos objetivos, limites e metas;
b) a Modelagem do BD é uma fase essencial, pois ela engloba as três principais etapas do projeto de BD: projeto conceitual, projeto lógico e projeto físico. O

primeiro passo é criar um esquema conceitual, tendo como resultado um diagrama ER; no segundo passo é escolhido um SGBD compatível com as necessidades do negócio; no terceiro passo é criado um esquema lógico com base na tradução do esquema conceitual, sendo compatível com o SGBD escolhido; no quarto passo é desenvolvido o esquema físico utilizando os dados mapeados no esquema lógico, são definidos estruturas de armazenamento e fluxo de acesso;

c) durante a fase de Implementação, o *hardware*, o *software* do SGBD, e os aplicativos necessários para seu funcionamento e manipulação são instalados. O real BD é construído nesta fase, onde são implementados diretamente no SGBD as tabelas, relacionamentos, visões e níveis de acesso. Também são tratadas outras questões tais como: performance, segurança, integridade e controle de concorrência (ROB; CORONEL 2005, tradução nossa);

d) a fase de Teste e Avaliação efetua inúmeros testes que avaliam performance, integridade, acesso concorrente e questões de segurança. É relevante comentar que os testes de dados não devem se limitar apenas a situações ideais onde todos os dados de entrada são reais. A modelagem de testes deve abranger o máximo possível de situações ao qual o sistema poderá ser apresentado futuramente (MULLER, 1999, tradução nossa). Caso ocorra alguma falha na bateria de testes, e falha for referente a fase de Modelagem de Banco de Dados, então será corrigido ou aperfeiçoado os projetos conceitual e lógico. Caso a falha corresponda a Fase de Implementação poderá ser necessário atualizar e/ou substituir o *hardware* e *software* do SGBD;

e) a fase operacional representa o inicio do uso do BD. Normalmente problemas que não foram previstos ou encontrados na fase de teste começam a aparecer,

alguns destes podendo necessitar grandes alterações e outros nem tanto. Porém independente do tamanho do problema, ele deverá ser tratado na fase seguinte (MULLER, 1999, tradução nossa);

f) para a fase de manutenção e avaliação são estipuladas metas com base nos problemas encontrados na fase operacional. Entretanto, esta fase não se limita a resolução de problemas, podendo ocorrer manutenções preventivas para *backup*, manutenção corretiva para recuperação de dados e por fim, dentre elas a mais importante, a manutenção adaptativa para melhorar o sistema, aumentando a performance e/ou adicionando entidades, atributos, relacionamentos, entre outros.

Segundo Rob e Coronel (2005) a fase de manutenção e avaliação pode ser facilmente implementada caso a modelagem do BD seja flexível e toda a modelagem esteja atualizada.

Após uma melhor compreensão do ciclo de vida de um BD é possível então um maior aprofundamento no projeto de banco de dados, pois se trata de uma fase de extrema relevância para o desenvolvimento de um BD.

2.1 ETAPAS DO PROJETO DE BANCO DE DADOS

As três etapas de projeto que compõem a modelagem do banco de dados são fundamentais para seu futuro sucesso. Nelas são definidas todas as estruturas conceituais, lógicas e físicas. Conforme abordado anteriormente, a projeção do BD permite uma identificação prévia de erros na estrutura de dados. Para que isto ocorra de forma correta seguindo o ciclo de vida do BD, todas as etapas devem ser executadas seguindo a devida teoria da modelagem de banco de dados.

A seguir são abordadas as etapas de projeto conceitual, projeto lógico e projeto físico.

2.1.1 Projeto Conceitual

No projeto conceitual ocorre a modelagem de dados, criando uma abstração de alto nível da estrutura dos dados para representar itens do mundo real (ROB; CORONEL 2005, tradução nossa). Nesta fase do projeto é criado um modelo de diagrama para representar entidades, relacionamento, atributos e restrições. Este modelo de dados pode ser representado por diferentes diagramas, tal como: diagrama hierárquico, diagrama de rede e diagrama ER.

Segundo Rob e Coronel (2005) a projeção do modelo conceitual é definida pela criação do diagrama tendo como base a analise dos dados e requisitos definidos na etapa anterior de Estudo de Viabilidades do BD, seguido pela Normalização da Modelagem e Verificação do modelo de dados. Conforme Heuser (2002) modelo mais comumente utilizado para representar banco de dados é o diagrama ER.

A etapa de modelagem compreende alguns passos para a construção de um diagrama ER. Primeiramente ocorre a identificação, analise e refinamento das regras de negócios. Com o resultado deste primeiro passo são definidas as principais entidades e seus relacionamentos, campos e chaves. No segundo passo ocorre a normalização das tabelas, onde são tratadas dependências e redundâncias. Em seguida com o resultado da normalização é desenvolvido um diagrama inicial e verificado junto ao usuário final se todos os requisitos estão contemplados. Caso o resultado final não atinja os objetivos almejados o diagrama ER é refeito(ROB; CORONEL 2005, tradução nossa).

Com isso, após a construção e verificação do diagrama ER é concebido um modelo de dados final que possibilita a construção de um projeto lógico coerente visto que as

etapas de projeto no nível conceitual permitem uma maior abstração dos requisitos necessários para a projeção de um banco de dados.

2.1.2 Projeto Lógico

O projeto lógico traduz o esquema definido no projeto conceitual em um modelo interno de baixo nível, mapeando os objetos em um modelo especifico para um determinado SGDB. Nesta etapa são é definido visão (*View*), gatilhos (*Triggers*), tabelas (*Tables*) e nível de acesso (*Access authorities*).

a) *view* é qualquer relação que não seja parte do modelo lógico, mas que se torna visível a um usuário como uma relação virtual (SILBERSCHATZ; KORTH; SUDARSHAN, 2006). A *View* tem como objetivo restringir alguns detalhes do modelo lógico do banco de dados, pois não é interessante que todos os usuários vejam o modelo lógico por uma questão de segurança;

b) *trigger* é uma instrução que o sistema executa automaticamente quando ocorre alguma alteração no banco de dados (SILBERSCHATZ; KORTH; SUDARSHAN, 2006). Para definir a ocorrência de um *Trigger* é necessário especificar um evento. no qual será testado uma condição. Caso a condição seja satisfeita o *Trigger* entrará em execução e cumprirá as ações especificadas;

c) *table* é um conjunto não ordenado de linhas, onde cada linha é um conjunto de campos, cada campo é identificado por um nome de campo e o conjunto de campos das linhas de uma tabela com o mesmo nome formam uma coluna (HEUSER, 2001);

d) *access authorities* tem por finalidade restringir o acesso aos dados. Cada usuário deve ter seu nível de acesso - *update* ou *read*. O nível de *update* é o

> mais avançado e deve ser restrito ao *Database Administrator* (DBA), pois permite que o usuário faça alterações diretas no banco de dados. O nível de *read* é menos restrito, pois usuários simples não necessitam ter visão de todos os dados, mas apenas aos dados que lhe são pertinentes (OLSON, 2008, tradução nossa).

Em resumo, o projeto lógico traduz o projeto conceitual que é independente de algum SGBD para um modelo dependente de um determinado SGBD, definindo os tipos de domínio de dados, as tabelas necessárias e as restrições de acesso por meio de *Rolers* (ROB; CORONEL 2005, tradução nossa).

2.1.3 Projeto Físico

O projeto físico é um processo que seleciona o armazenamento e o acesso de dados do BD.

O armazenamento é definido em função do tipo de dispositivo de *hardware*, tipo de método de acesso aos dados e do SGBD (ROB; CORONEL 2005, tradução nossa). No entanto, o projeto físico não se limita apenas a definir a localização do armazenamento dos dados nos dispositivos, mas sim, a desempenhar um papel fundamental para maximizar a performance do sistema.

Para selecionar o armazenamento de dados e possibilitar uma melhor performance, o esquema físico do BD faz uso da seleção de índices, particionamento e clusterização de dados (LIGHTSTONE; TEOREY; NADEAU, 2007, tradução nossa).

Os índices são as posições nas quais os dados são armazenados em disco. A indexação é uma forma de permitir um acesso mais rápido aos valores de uma coluna e/ou concatenação de colunas (HARRINGTON, 2002). Eles podem variar em índices ordenados

baseados em uma ordem classificada dos valores, ou índices de *hash* baseados em uma distribuição uniforme de valores por um intervalo de baldes (*buckets*), sendo os baldes um valor atribuído por uma função *hash* (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

O particionamento em relação ao esquema físico de BD é um método de reduzir a sobrecarga em algum componente de *hardware*, tal como um *hard disk* (HD), particionando (dividindo) os dados em vários HD. Isto influencia no balanceamento da sobrecarga no sistema prevenindo a ocorrência de "gargalos" (LIGHTSTONE, 2007, tradução nossa).

Clusterização multidimensional ou *Multidimensional clustering* (MDC) é uma técnica que agrupa "clusteriza" os dados em dimensões, os dados podem ser agrupados de acordo com alguma especificação dimensional, por exemplo, tempo e espaço. O *cluster* é designado a utilizar o conhecimento antecipado de possíveis sobrecargas de dados (LIGHTSTONE, 2007, tradução nossa).

O projeto de banco de dados representa a essência da modelagem de um BD relacional, porém a estrutura lógica baseada no conceito de Entidade Relacionamento é apresentada no Capítulo seguinte.

3 ENTIDADE RELACIONAMENTO

Segundo Heuser (2001) a técnica de modelagem de dados mais conhecida e utilizada é a abordagem ER. Nesta técnica o modelo ER é representado normalmente por um Diagrama ER (DER), esta abordagem foi criada por Chen (1970).

3.1 ENTIDADES

Para Machado e Abreu (1996) entidades são abstrações de objetos existentes no mundo real, porém essas abstrações são delimitadas de acordo com o problema que se almeja resolver, ou seja, entidades são um conjunto de representações de um determinado negócio o qual se tem interesse em manter registros.

O propósito de se armazenar registros em um banco de dados é fundamentado na necessidade de rever os dados mais tarde, desta forma é necessário que haja uma distinção entre as entidades de forma que possibilite recuperar apenas a entidade exata requisitada (HARRINGTON, 2002).

3.2 ATRIBUTOS

Atributos são as propriedades especificas que descrevem uma entidade, eles podem ser classificados de acordo com o modelo ER em atributos: simples, composto, monovalorados, multivalorados e derivados (ELMASRI; NAVATHE, 2002).

Atributos simples são aqueles que seus valores não podem se subdividir, por exemplo, idade, sexo e estado matrimonial. Para facilitar consultas usualmente atributos compostos são transformados em atributos simples (ROB; CORONEL, 2005, tradução nossa).

Atributos compostos podem ter seus valores subdivididos, por exemplo, nome e numero e telefone. Desta forma o atributo nome pode ser subdivido em nome, primeiro sobrenome e segundo sobrenome; o telefone pode ser subdividido em ddd, código da região e numero da linha.

Atributos multivalorados são aqueles que podem ter muitos valores, por exemplo, um atributo ‘certificados’ uma pessoa pode ter muitos certificados ou nenhum.

Atributos monovalorados possuem no máximo um valor associado a uma entidade, por exemplo, idade é um atributo de valor único da pessoa.

Atributos derivados dependem estritamente de outro atributo, o exemplo clássico desta propriedade é o calculo da idade de uma entidade pessoa, onde o campo ‘idade’ é calculado com base na subtração da data atual pelo campo de ‘DataDeNascimento’, não necessitando manter o campo ‘idade’ armazenado no BD pois seu valor é calculado em tempo de execução.

Heuser (2001) define que os atributos possuem cardinalidades de forma análoga a uma entidade em um relacionamento, estas cardinalidades têm por objetivo definir quantos valores de um atributo podem estar associados a uma entidade a qual ele pertence. Portanto é definido que para um atributo monovalorado a cardinalidade mínimo/máxima é (1/1) e para um atributo multivalorado a cardinalidade mínimo/máxima é (0/N).

3.2.1 Atributos Chave

Atributo chave tem como principal função estabelecer relações entre as entidades de um banco de dados relacional (HEUSER, 2001). Os principais tipos de chaves em um banco de dados relacional são: chave primária (*primary key*), chave estrangeira (*foreing key*) e chave concatenada ou chave composta (*composite key*).

Chave primária é um identificador que possibilita a distinção entre uma ocorrência de entidade das demais ocorrências da mesma entidade por meio de um atributo ou combinação de atributos (HEUSER, 2001). Para Powell (2006, tradução nossa) é fundamental que exista uma unicidade para identificar cada ocorrência de uma entidade, pois só desta maneira é possível que uma busca retorne apenas a exata ocorrência da entidade solicitada.

Chave estrangeira é uma copia da chave primária criada em outra tabela com o objetivo de estabelecer uma ligação entre as tabelas relacionadas, por meio deste mecanismo é possível criar uma relação dentro de um banco de dados relacional.

Chave concatenada pode ocorrer quando mais de uma coluna ou combinações de coluna possui poder de unicidade como chave primária, neste caso apenas uma coluna é escolhida como chave primária as demais são classificadas como chave alternativa.

3.3 ENTIDADE FRACA

Uma entidade fraca possui tal dependência que não pode existir em um banco de dados a menos que uma instância relacionada de outra entidade esteja presente e relacionada a ela (HARRINGTON, 2002). A entidade fraca não possui atributos chaves próprios, seus atributos chave são herdados da entidade da qual dependente e agregados as seus atributos próprios (ELMASRI; NAVATHE,2002) .

3.4 RELACIONAMENTOS

Relacionamentos são associações entre entidades (DATE 2004). Os relacionamentos podem variar de acordo com seu grau de relacionamento, os mais comuns são aqueles que envolvem duas entidades (binários), porem existe relacionamentos que

envolvem apenas uma entidade (unários) ou mais de duas entidades (GARCIA-MOLINA; ULLMAN; WIDOW, 2001, tradução nossa). Existem três tipos de relacionamentos possíveis entre entidades: um-para-muitos (1:N), muitos-para-muitos (N:N) e um-para-um (1:1), o relacionamento é representado através de um losango que conecta as entidades relacionadas através de uma linha de relacionamento, o nome do relacionamento normalmente é representado por um verbo dentro do losango conforme Figura 2 (ROB; CORONEL 2005, tradução nossa).

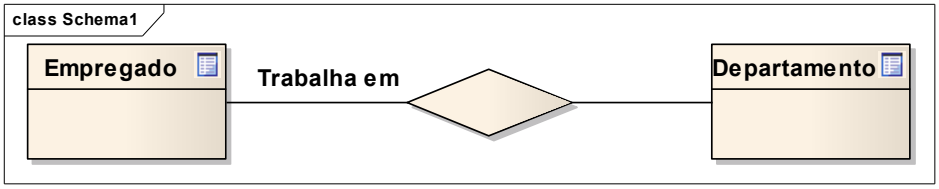


Figura 2. Representação de um Relacionamento
Fonte: Adaptado de RAMAKRISHNAN, R; GEHRKE, J (2000, tradução nossa)

Na figura 2 existem duas entidades: ‘Empregado’ e ‘Departamento’, representadas pelos dois retângulos; ligados por uma linha com um losango chamado de ‘Trabalha em’ representando um relacionamento.

3.4.1 Classificação e Cardinalidade de Relacionamentos Binários

Segundo Silberschatz, Korth e Sudarshan (2006) as cardinalidades são utilizadas para representar relacionamentos binários, porém podem também representar outros tipos de relacionamentos. Para um conjunto de relacionamento binário R entre o conjunto de entidades E1 e E2, as cardinalidades correspondem a uma das seguintes: um-para-muitos, muitos-para-muitos e um-para-um.

3.4.1.1 Relacionamento Um-para-muitos

Um-para-muitos (1:N) significa que uma entidade E1 é associada a qualquer numero de entidades (zero ou mais) em E2.Entretanto, uma entidade em E2 pode ser associada a no máximo uma entidade em E1, conforme Figura 3 (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

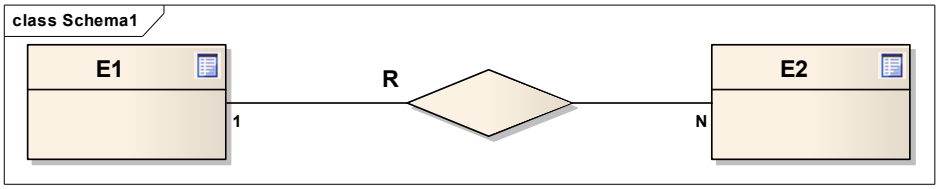


Figura 3. Relacionamento de um-para-muitos (1:N)
Fonte: Adaptado de SILBERSCHATZ, A; KORTH, H; SUDARSHAN, S (2006)

É possível na Figura 4 visualizar um diagrama de conjuntos que expressa as possíveis relações de um-para-muitos.

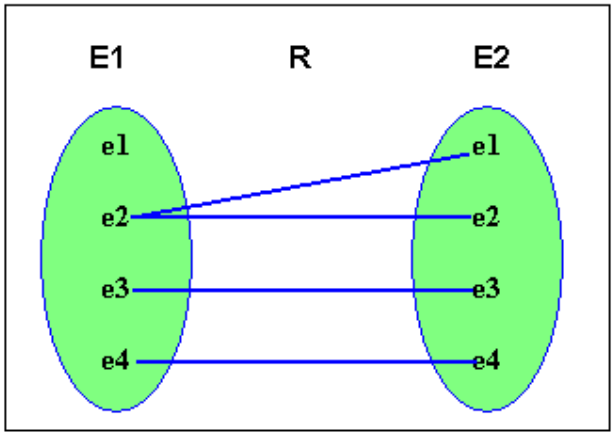


Figura 4. Diagrama de um conjunto (1:N)
Fonte: Adaptado de MANNINO (2007)

Conforme Figura 4, dada uma entidade E1 que possui um relacionamento R formando um par ordenado (E1, R) e uma entidade E2 que possui um relacionamento R formando um par ordenado (E2, R), podem adquirir os seguintes valores:

a) Cardinalidade mínima (E1, R) = 0;

b) Cardinalidade máxima (E1, R) = N;

c) Cardinalidade mínima (E2, R) = 1;

d) Cardinalidade máxima (E2, R) = 1.

3.4.1.2 Relacionamento Muitos-para-muitos

Muitos-para-muitos (N:N) uma entidade E1 é associada a qualquer numero de entidades (zero ou mais) em E2, e uma entidade E2 é associada a qualquer numero de entidades (zero ou mais) em E1, conforme Figura 5 (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

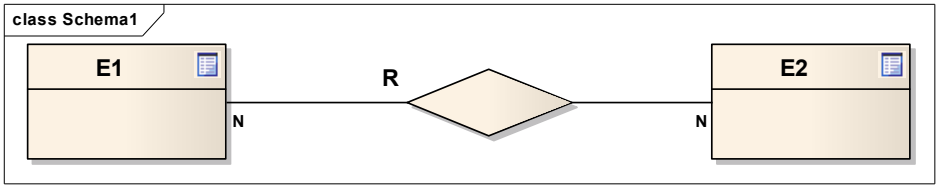


Figura 5. Relacionamento de muitos-para-muitos (N:N)
Fonte: Adaptado de SILBERSCHATZ, A; KORTH, H; SUDARSHAN, S (2006)

Na Figura 6 é apresentado um diagrama de conjuntos que expressa as possíveis relações de muitos-para-muitos.

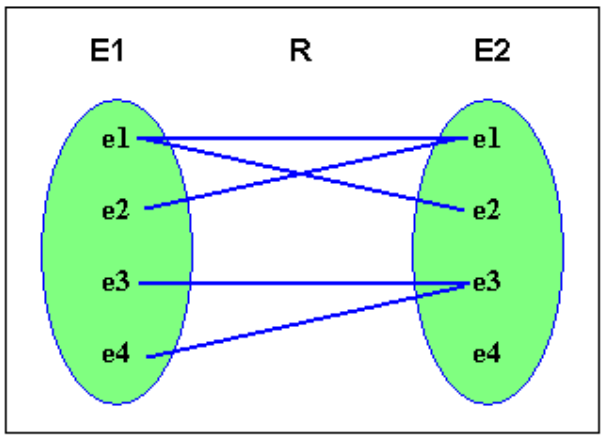


Figura 6. Diagrama de um conjunto (N:N)
Fonte: Adaptado de MANNINO (2007)

Conforme Figura 6, dada uma entidade E1 que possui um relacionamento R formando um par ordenado (E1, R) e uma entidade E2 que possui um relacionamento R formando um par ordenado (E2, R), podem adquirir os seguintes valores:

a) Cardinalidade mínima (E1, R) = 0;

b) Cardinalidade máxima (E1, R) = N;

c) Cardinalidade mínima (E2, R) = 0;

d) Cardinalidade máxima (E2, R) = N.

3.4.1.3 Relacionamento Um-para-um

Um-para-um (1:1) uma entidade E1 é associada a no máximo uma entidade em E2, uma entidade em E2 é associada a no máximo uma entidade em E1, conforme Figura 7 (SILBERSCHATZ; KORTH; SUDARSHAN, 2006).

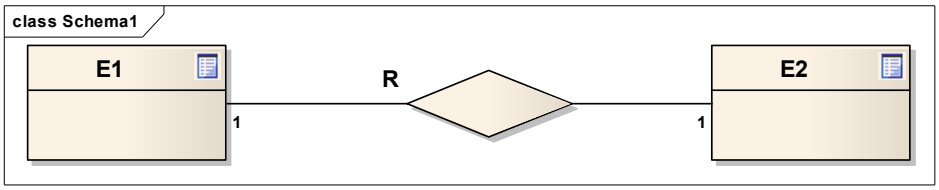


Figura 7. Relacionamento de um-para-um (N:N)
Fonte: Adaptado de SILBERSCHATZ, A; KORTH, H; SUDARSHAN, S (2006)

Com o intuito de melhor representar esta relação de (1:1), é apresentado na Figura 8 um diagrama de conjuntos que expressa as possíveis relações de muitos-para-muitos.

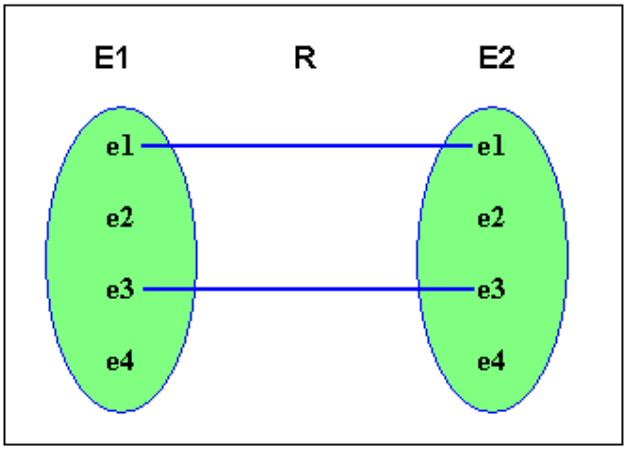


Figura 8. Diagrama de um conjunto (1:1)
Fonte: Adaptado de MANNINO (2007)

Conforme Figura 8, dada uma entidade E1 que possui um relacionamento R formando um par ordenado (E1, R) e uma entidade E2 que possui um relacionamento R formando um par ordenado (E2, R), podem adquirir os seguintes valores:

a) Cardinalidade mínima (E1, R) = 0;

b) Cardinalidade máxima (E1, R) = 1;

c) Cardinalidade mínima (E2, R) = 0;

d) Cardinalidade máxima (E2, R) = 1.

É importante diferenciar que o Diagrama Entidade Relacionamento (DER) usa o termo conectividade para classificar os tipos de relacionamentos e cardinalidade para representar o numero especifico de ocorrências de uma entidade associada a outra entidade, esta distinção é representada na Figura 9. O modelo de Chen (1970) representa uma cardinalidade no formato (X,Y), onde X é o numero mínimo em que uma entidade pode ocorrer em uma relacionamento e Y é o numero máximo em que uma entidade pode ocorrer em uma relacionamento (ROB; CORONEL, 2005, tradução nossa).

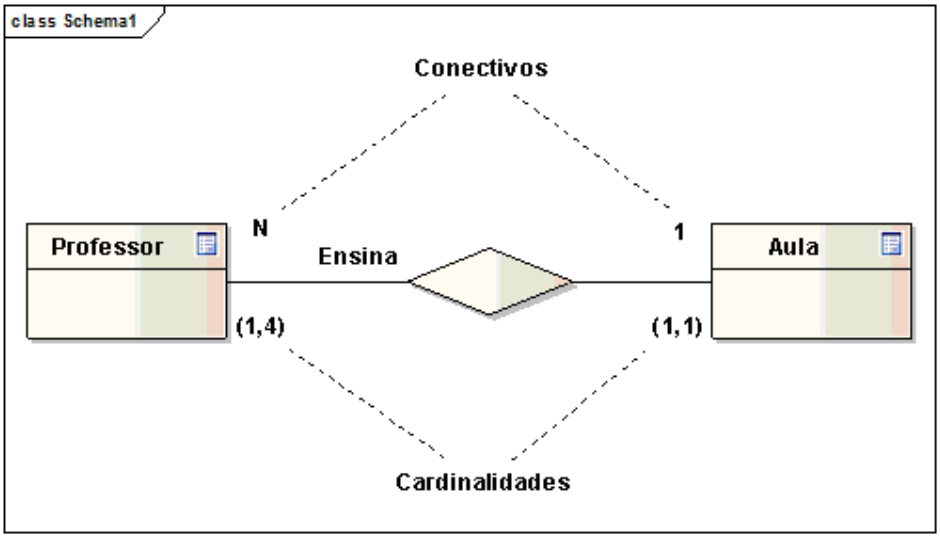


Figura 9. Diferença entre conectividade e cardinalidade
Fonte: Adaptado de ROB, P; CORONEL, C (2005, tradução nossa)

3.5 MODELO ENTIDADE RELECIONAMENTO ESTENDIDO

O modelo ER-E foi concebido devido a necessidade de uma melhor modelagem da abstração dos dados em relação ao mundo real, visto que o modelo ER definido por Chen

(1970) já não engloba algumas características que surgiram nos últimos anos com a sofisticação de aplicações empresariais (ELMASRI ; NAVATHE, 2002).

A evolução do modelo ER para ER-E possibilitou modelares as seguintes abstrações: Generalização/Especialização (G/E), herança, subclasse, superclasse.

3.5.1 Generalização / Especialização (G/E)

Conforme Heuser (2001) as propriedades de uma entidade podem variar não apenas de acordo com relacionamentos e atributos, mas também através do conceito de G/E. Este conceito depende diretamente do mecanismo de herança de propriedades, visto que a G/E pode variar em duas formas: total ou parcial.

a) Numa G/E total cada ocorrência de uma entidade genérica corresponde a pelo menos uma ocorrência da entidade especializada conforme Figura 10;

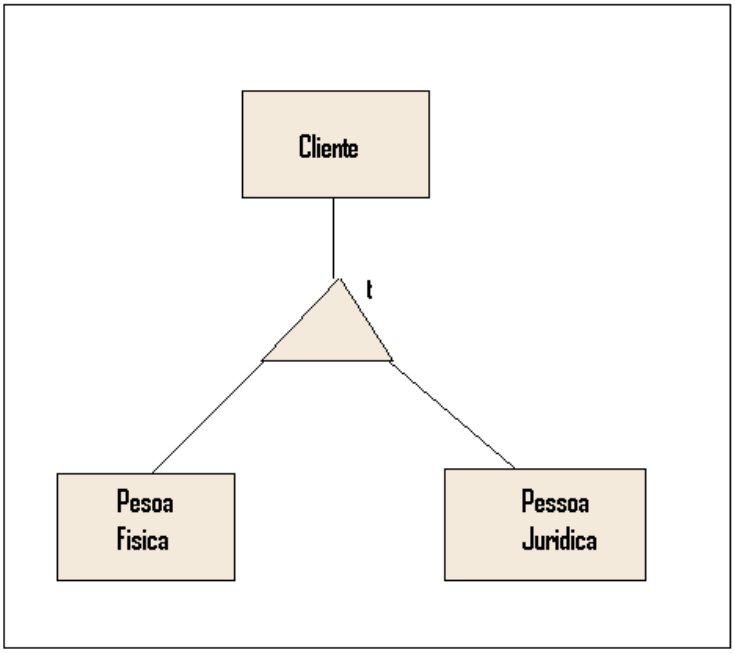


Figura 10. G/E total
Fonte: Adaptado de Heuser (2001)

Na Figura 10 o **t** indica que todo cliente é uma pessoa jurídica ou física.

b) Já em uma G/E parcial nem toda ocorrência de uma entidade genérica corresponde em uma entidade especializada.

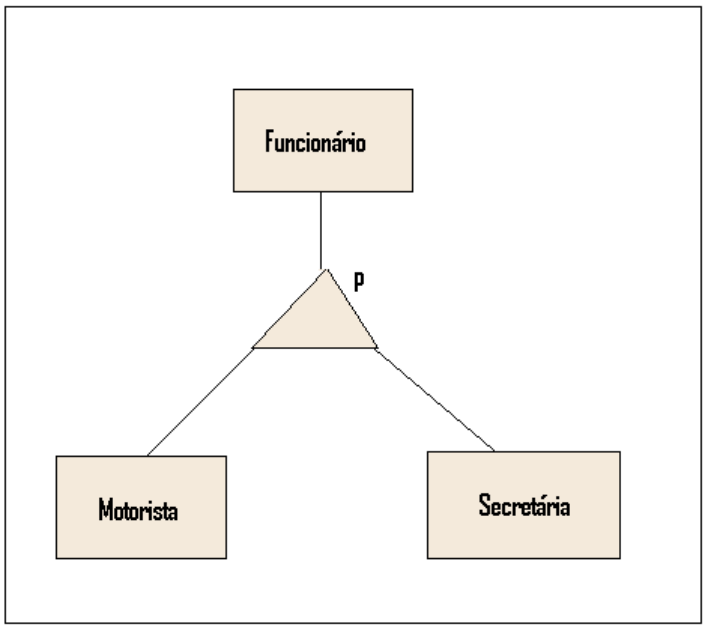


Figura 11. G/E parcial
Fonte: Adaptado de Heuser (2001)

Na Figura 11 **p** indica que nem todo funcionário é motorista ou secretária.

Além as duas formas supracitadas uma G/E pode ser uma disjunção exclusiva se cada elemento da classe genérica é mapeado para pelo menos um elemento das subclasses; e/ou uma sobreposição se algum elemento da classe genérica que é mapeado para duas ou mais subclasses diferentes.

3.5.2 Herança

Herança é uma abstração que modela os atributos e relacionamentos que uma entidade pai possui transcrevendo-as para suas entidades filhos. Basicamente herdar propriedades significa que cada ocorrência da entidade especializada possui, além de seus próprios campos, relacionamentos, restrições e G/E, também as propriedades da ocorrência da entidade genérica correspondente (HEUSER, 2001).

3.5.3 Classe

A abstração de orientação a objetos classifica os objetos de acordo com suas semelhanças e diferenças, objetos que compartilham características comuns são agrupados em classes. Onde classe é uma coleção de objetos similares que dividem estrutura (atributos) e comportamento (relacionamentos) (ROB; CORONEL, 2005, tradução nossa).

Classes são organizadas em classe hierárquica, semelhante a uma arvore *upside-down* onde cada classe possui somente uma classe pai.

3.5.3.1 Subclasse

Desta forma, uma entidade filho que herda os atributos e relacionamentos de uma entidade pai, é definida como um subclasse da entidade pai. Uma subclasse nada mais é do que uma especialização de uma entidade genérica.

3.5.3.2 Superclasse

Neste caso, a entidade pai representa uma superclasse ao qual as entidades filhas são subclasses. É possível trançar um paralelo onde uma entidade genérica representa uma superclasse e suas entidades filhos são especializações que herdam seus atributos.

Após uma melhor compreensão da teoria de modelagem de BD e análise da estrutura lógica do DER é possível visualizar que o projeto de BD não é uma tarefa corriqueira, pois exige um alto nível de abstração. Desta forma a engenharia reversa vem com

o objetivo de auxiliar tanto o desenvolvimento quanto a manutenção de projetos, no Capítulo subseqüente são apresentadas a definição e metodologia de engenharia reversa.

4 ENGENHARIA REVERSA

Engenharia reversa tem seu foco na análise e extração de código fonte de sistemas já implementados, tendo como propósito alcançar a modelagem do projeto em alto nível de abstração. Para Pressman (2000) o nível de abstração de um processo de engenharia reversa depende de quão detalhado o esquema final deva ser, considerando que o modelo ER é um esquema de alto nível de abstração, pois representa estrutura e fluxo de dados.

4.1 ENGENHARIA REVERSA EM BANCO DE DADOS

Engenharia reversa em banco de dados (ERBD) lida com a tarefa de compreender bases de dados legados e extrair especificações de um esquema conceitual a partir de seu código fonte (CHIANG; BARRON; STOREY; 1997). ERBD requer a analise do catálogo do BD; *data mining* do BD; interação com usuários avançados, programadores e analistas que possuem o devido conhecimento do sistema e alguma informação adicional sobre o código fonte das aplicações (PEDRO-DE-JESUS; SOUSA, 1999).

Hainaut (2002) descreve alguns problemas específicos usualmente encontrados no processo de ERBD, tais como:

a) debilidade na modelagem do SGBD: o modelo técnico provido pelos SGBD pode expressar somente um pequeno subgrupo de estruturas e restrições projetadas no esquema conceitual;
b) estruturas implícitas: alguns construtores não são declarados na *Data Definition Languange* (DDL) do BD intencionalmente, fato que dificulta sua identificação;

c) modelagem inadequada: nem todos os BD são construídos por projetistas experientes e bem capacitados, em muitos casos desenvolvedores sem o devido conhecimento da teoria e metodologia de BD constroem estruturas pobres ou até mesmo erradas;

d) estruturas obsoletas: algumas partes do BD são abandonadas e ignoradas pela aplicação atual.

Devido aos problemas citados acima é evidenciado a necessidade do uso de uma metodologia para apoiar a ERBD, visto que apenas o uso de engenharia reversa não garante que certas abstrações contidas nos projetos sejam contempladas, a adoção de uma metodologia com alto nível de abstração é primordial para que se alcance como um resultado final o esquema conceitual mais próximo da realidade.

4.2 METODOLOGIA UTILIZADA NA EXECUÇÃO DA ENGENHARIA REVERSA EM BANCO DE DADOS

Nos últimos anos foram abordadas diversas metodologias para ERBD, sendo algumas especificas para determinados BD e outras de uso comum a BD relacionais. Pedro-de-Jesus e Sousa (1999) apresentam sete metodologias para ERBD:

a) método definido por Chiang (1994): necessita como entrada as instancias dos dados e o esquema relacional, incluindo chaves primárias. É suposto que esteja na 3FN, os atributos são nomeados de forma consistente impedindo que ocorram erros na definição de campos chave. Basicamente é dividido em três etapas: na primeira são classificados relacionamentos e atributos, tendo como base o esquema relacional e as chaves primárias; na segunda ocorre a geração e verificação de instancias de dados inclusos de forma heurística fundamentados

na classificação de relacionamentos e atributos; na terceira são identificados componentes do ER-E a partir de uma lista de regras. Após a execução das três etapas tem-se como saída um DER-E;

b) método definido por Johannesson (1998): necessita como entrada o esquema relacional, dependências funcionais e de inserção. É suposto que esteja na 3FN. É fundamentado em quatro tapas: divisão do esquema relacional que corresponde a mais de um objeto; adição de um esquema relacional extra para manipular a ocorrência de certos tipos de dependências de inserção; relações de um mesmo tipo de objeto são quebradas em esquemas e aplicação de um mapeamento de esquema relacional convertendo para um esquema conceitual;
c) método definido por Markowitz (1990): necessita como entrada o esquema relacional, dependências de inserção de chave e integridade de referências a restrições. Não assume que esteja na 3FN. Composto por cinco etapas: transformação de um esquema relacional para uma forma apropriada para identificar estruturas de objetos ER-E; verificação do esquema relacional, dependências funcionais e de inserções obtidas após a tradução, com o objetivo de avaliar se satisfazem o conjunto de probidades; determinação do tipo de interação-objeto para cada dependência de inserção; eleição de um esquema ER-E candidato utilizando algumas regras de mapeamento e verificação da qualidade do esquema ER-E gerado;
d) método definido por Navathe (1992): necessita como entrada o esquema relacional, assume que esteja na 3FN e requer coerência nos nomes dos atributos, sem chaves estrangeiras ambíguas ou homônimas. Demanda três etapas: relações são processadas e classificadas com auxilio humano; os relacionamentos classificados são mapeados com base nos atributos chave e os

relacionamentos que não se encaixam nas classificações acima são tratados um a um. Este método apresenta coma saída um esquema ER-E;

e) método definido por Petit (1996): necessita como entrada um esquema relacional, com restrições únicas e não nulas. Não assume que esteja na 3FN. Formado por quatro etapas: propõe dependências de inserção utilizando esquema relacional, instancias do BD e união de consultas com base no código; sugere também dependências funcionais de campos não chave utilizando esquema relacional e chaves candidatas; utiliza o algoritmo de decomposição de normalização para alcançar um esquema na 3FN e por fim converte o esquema relacional para um esquema ER-E;

f) método definido por Blaha e Premerlani (1994; 1998): necessita como entrada um esquema relacional e dados. Não assume que esteja na 3FN. Esta metodologia possui quatro etapas: prepara inicialmente um modelo de objeto para representar cada relação como uma classe experimental; conta com o apoio do usuário para encontrar as chaves candidatas através de algumas metodologias descritas; o usuário deve determinar as chaves estrangeiras tendo como base algumas informações e o refinamento do esquema *Object Modelling Technique* (OMT) é progressivo contando com o auxilio o usuário que é orientado por meio de dados de algumas consultas. Apresenta como saída um esquema OMT;

g) método definido por Signore (1994): necessita como entrada esquema relacional e código. Não assume que esteja na 3FN. Concebido em três etapas: primeiramente são definidas as chaves primárias, no final desta etapa é cada relação deve ter uma chave primária ou pelo menos uma chave candidata; na segunda etapa são detectadas restrições de chaves iguais utilizando comandos

de SQL; por fim é obtido o esquema conceitual por meio da conceituação dos indicadores de esquema, chave primária, SQL e indicadores "procedurais". Como saída é obtido um modelo conceitual.

As metodologias citadas acima diferem nos propósitos e nas respectivas saídas, algumas consideram o nível de entrada mínimo e lidam com linguagens já não tão usuais atualmente, é relevante comentar que estas metodologias foram estipuladas para utilizar ERBD em bases legadas onde realmente não havia informações suficientes tais como: chave primária, chave secundária, restrições e dependências. Porém com adoção da SQL como padrão possibilitou que muitos destes problemas fossem resolvidos. Além disto, a ERBD não possui como único propósito oferecer manutenção para BD legados, a engenharia reversa também pode apoiar construção e manutenção de bases de dados relativamente atuais.

Neste sentido uma metodologia mais próxima da realidade atual é proposta por Heuser (2001), onde ele define quatro etapas:

a) construção do modelo ER;
b) identificação de relacionamentos;
c) definição de atributos;
d) definição das chaves de identificação.

A fim de facilitar a compreensão do processo de ERBD Heuser (2001) demonstra cada etapa por meios de um BD para um sistema acadêmico, desta forma também é proposto neste trabalho como exemplo um BD para um sistema bancário:

agencia(nome-agencia, cidade_agencia, ativo)

cliente(id-cliente, nome_cliente, rua_clinte, cidade_cliente)

conta(numero-conta, saldo)

conta_poupança(numero-conta, taxa_juros)

conta_corrente(numero-conta, quantia_saque_descobrto)

emprestimo(numero-emprestimo, quantia)

agencia_conta(numero-conta, nome-agencia)

agencia_emprestimo(numero-emprestimo, nome-agencia)

depositante(id-cliente, numero-conta)

tomador(id-cliente, numero-emprestimo)

4.2.1 Construção do Modelo ER

Primeiramente na construção de um modelo ER são definidas para cada tabela uma classificação, uma tabela pode ser classificada em: uma entidade, um relacionamento de N:N ou uma entidade especializada A restrição que define qual classificação uma tabela pertence é a chave primária, Heuser (2001) definiu uma regra para cada uma das três possíveis classificações.

A tabela que possui uma chave primária composta por mais do que uma chave estrangeira representa um relacionamento de N:N entre as entidades correspondentes as tabelas referenciadas pelas chaves estrangeiras (HEUSER, 2001). No exemplo do sistema bancário duas tabelas se enquadram nesta classificação, a tabela ‘depositante’ e a tabela ‘tomador’. A tabela ‘depositante’ possui como chave primária ‘id-cliente’ e ‘numero-conta’, sendo estes dois atributos chaves estrangeiras das respectivas tabelas ‘cliente’ e ‘conta’, o mesmo ocorre na tabela ‘tomador’ onde sua chave primária corresponde a duas chaves estrangeiras ‘id-cliente’ e ‘numero-emprestimo’ referentes às entidades ‘cliente’ e ‘emprestimo’. Ambos os exemplos são classificados como um relacionamento conforme Figura 12.

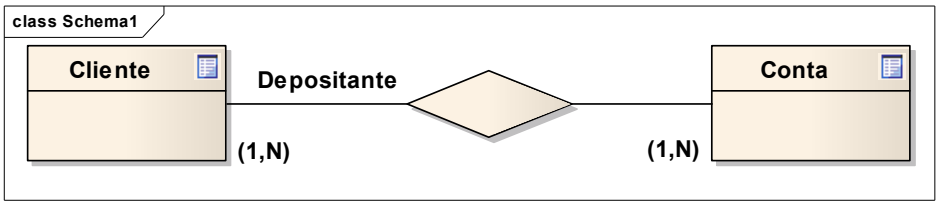


Figura 12. Representação de um relacionamento de N:N

A tabela que toda chave primária é uma chave estrangeira simboliza uma entidade que compõe uma especialização (HEUSER, 2001). Conforme o exemplo do sistema bancário as tabelas ‘conta_poupança’ e ‘conta_corrente’ possuem a chave primária ‘numero-conta’ que também é chave estrangeira referente à tabela ‘conta’, sendo assim ‘conta_poupança’ e ‘conta_corrente’ representam uma especialização da tabela ‘conta’, visto que para cada ocorrência de uma ‘conta_poupança’ ou ‘conta_corrente’ é necessário que exista um registro da tabela ‘conta’ conforme Figura 13.

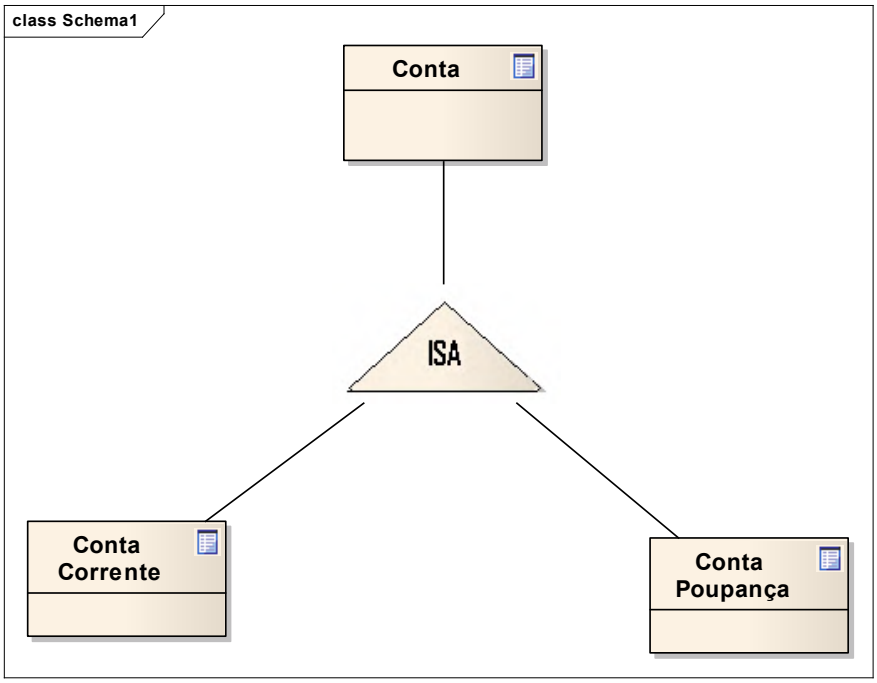


Figura 13. Representação de uma especialização

Para os demais casos que não se enquadram em uma das regras citados acima, as tabelas são consideradas entidades (HEUSER, 2001). No caso do sistema bancário a tabela ‘agencia’ possui uma chave primária ‘nome_agencia’ que não é chave estrangeira de outra

tabela, portanto não se encaixa em nenhuma regra anterior sendo classificada como uma entidade conforme Figura 14.

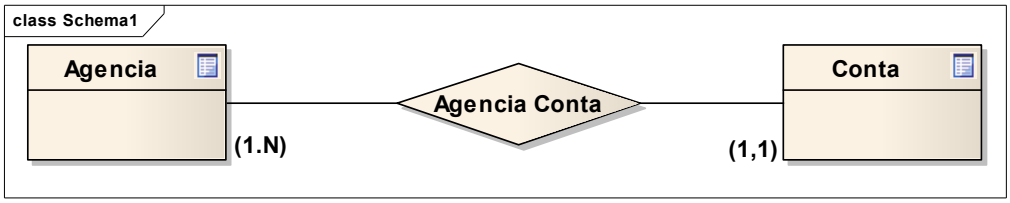


Figura 14. Representação de uma entidade

4.2.2 Identificação de Relacionamentos

Como os relacionamentos de N:N já foram identificados acima, visto que são classificados de acordo com a primeira regra e a segunda regra define as especializações, restam apenas os relacionamentos de 1:1 e 1:N que são identificados pela terceira regra. Diferente dos relacionamentos de N:N não é possível determinar se um relacionamento possui a cardinalidade de 1:1 ou 1:N, pois estas cardinalidades são definias baseado nas regras negocio, como no processo de engenharia reversa nem sempre se tem informações sobre as regras de negocio é possível verificar estas cardinalidades analisando os possíveis conteúdos no BD.

4.2.3 Definição de Atributos

Nesta etapa são definidos os campos que representam atributos. Para Heuser (2001) os campos que não são chaves estrangeiras são considerados atributos das tabelas correspondentes. Conforme esta definição são representados os atributos na Figura 15.

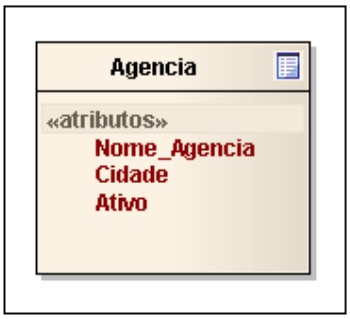


Figura 15. Representação dos atributos

4.2.4 Definição de identificadores de entidade

Nesta ultima etapa são definidos os campos que identificam as entidades e relacionamentos. Todo campo que faz parte da chave primaria e não é chave estrangeira simboliza um atributo identificador da entidade ou relacionamento, e todo campo que faz parte da chave primaria e é chave estrangeira simboliza um identificador externo da entidade ou relacionamento. Conforme esta definição são representados os atributos identificadores na Figura 16.

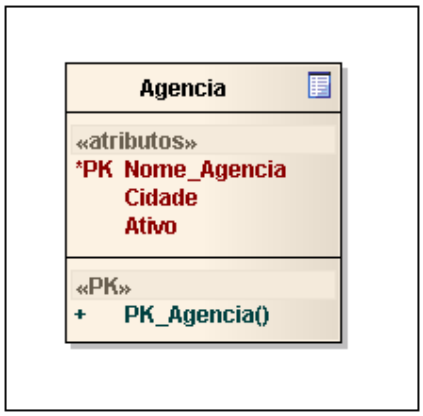


Figura 16. Representação de um atributo identificador

Aplicando esta ultima etapa a todas as tabelas tem se com resultado um DER-E. Esta metodologia de ERBD é amplamente utilizada, no capitulo subseqüente são apresentados dois trabalhos que utilizam ERBD definido por Hesuer em suas ferramentas.

5 TRABALHOS CORRRELATOS

A utilização de engenharia reversa é um processo para alcançar a compreensão da estrutura e interrelação de um sistema. Um de seus objetivos é criar representações com o intuito de documentar sistemas (DAVIS; AIKEN 2000, tradução nossa).

Dessa forma muitas pesquisas e publicações têm como assunto o uso de engenharia reversa. Uma das grandes áreas abrangidas neste contexto é a engenharia reversa de banco de dados, tendo como uma de suas finalidade gerar uma modelagem adequada.

Matté (2002) define um conjunto de procedimentos de engenharia reversa de bancos de dados espaciais que auxiliam na criação e manutenção de modelos conceituais para aplicações de Sistemas de Informação Geográfica (SIG) a partir de dados já existentes, contribuindo para o aumento da qualidade dos SIG, utiliza a metodologia definida por Heuser.

Outra metodologia diferenciada em engenharia reversa de BD é utilizada por Brognoli (2008), onde os metadados contidos no dicionário de dados, compatível com dois SGBD: Firebird e Oracle, são extraídos por meio de uma conexão direta utilizando um *driver Java Database Connectivity* (JDBC), após a extração gera-se, com base nos metadados mapeados um esquema de diagrama ER de forma semi-automática contando com o auxilio do usuário final para definir alguns relacionamentos, utiliza a metodologia definida por Heuser.

Nobre e Souza (2005) abordam engenharia reversa de banco de dados fazendo uso do mapeamento de *script* SQL, especificamente do SGBD Microsoft SQL Server. Com base nos dados obtidos no mapeamento realiza-se o reconhecimento das entidades e relacionamentos com o intuito de produzir uma modelagem na forma de um diagrama ER, utiliza a metodologia definida por Heuser.

Em Ávila e Mello (2008) é apresentada uma ferramenta de apoio ao processo de normalização de tabelas baseado na analise de dados contidos na extração de um arquivo XML de um banco de dados, com o intuito de produzir um modelo relacional normalizado.

6 SISTEMA DE ENGENHARIA REVERSA DE BANCO DE DADOS

O sistema proposto nesta pesquisa é uma ferramenta de auxílio a modelagem de banco de dados relacionais que utiliza engenharia reversa por meio da extração de metadados contidos no *script* SQL.

6.1 MODELAGEM DE SISTEMA

O sistema foi modelado com base na metodologia de Engenharia de *Software*, por meio de diagramas é possível uma melhor compreensão da estrutura lógica e fluxo de informação do sistema. A seguir são apresentados três modelos de diagramas: caso de uso, classe e máquina de estados. Cada um deles representa um nível diferente de abstração facilitando a compreensão do sistema como um todo.

6.1.1 Modelagem de Casos de Uso

O diagrama de caso de uso ilustra as interações entre o ator e o sistema, representando a visão que o usuário possui em relação ao sistema conforme Figura 17.

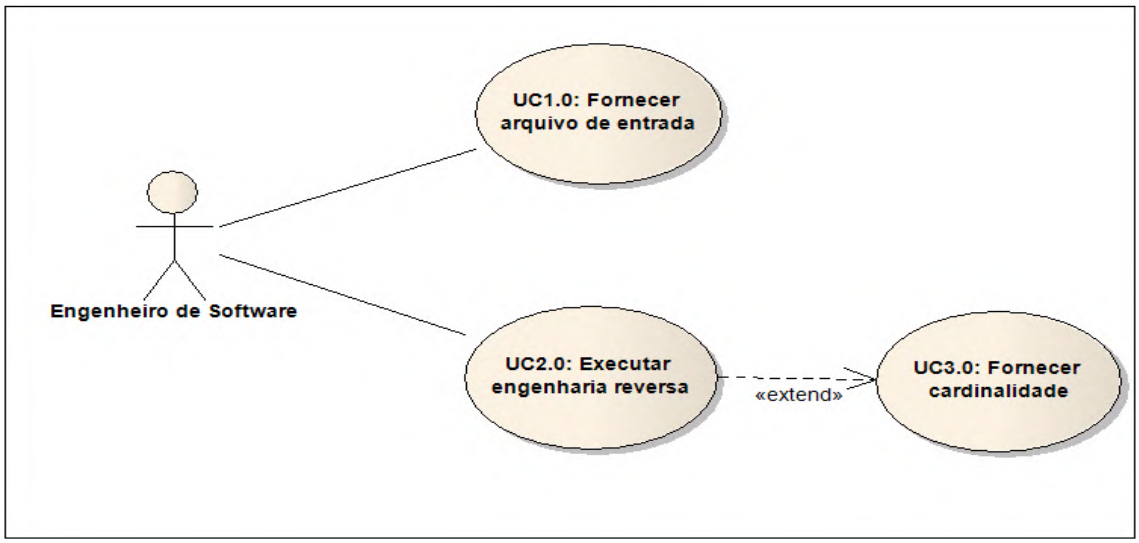


Figura 17. Caso de uso do sistema proposto

a) Engenheiro de Software: este ator refere-se ao engenheiro de software responsável pela utilização do sistema;

b) UC1.0 Fornecer arquivo de entrada : este caso de uso refere-se ao fornecimento de um arquivo de entrada para a geração do modelo ER , ER-E do banco de dados;

c) o ator fornece o caminho de destino do arquivo por meio da caixa de texto “caminho”;

d) o caso de uso se encerra;

e) UC2.0 Executar engenharia reversa: este caso de uso inicia-se quando o ator pressiona o botão "Executar Engenharia Reversa";

f) o ator presiona o botão “Executar Engenharia Reversa” <<extend UC3.0 Fornecer cardinalidade>> ;

g) o sistema apresenta o diagrama resultante da engenharia reversa;

h) o caso de uso se encerra;

i) UC3.0 Fornecer cardinalidade: este caso de uso refere-se ao fornecimento da cardinalidade dos relacionamentos entre as entidades apresentadas com base na análise léxica do arquivo de dados fornecido no UC1.0;

j) o sistema apresenta a interface T3;

k) o ator seleciona a cardinalidade;

l) o caso de uso retorna ao passo 2 do UC2.0 Executar engenharia reversa;

6.1.2 Modelagem de Classes

O diagrama de classe ilustra a estrutura lógica do sistema, demonstra como cada classe se comporta em relação as demais, transmitindo uma abstração de alto nível.

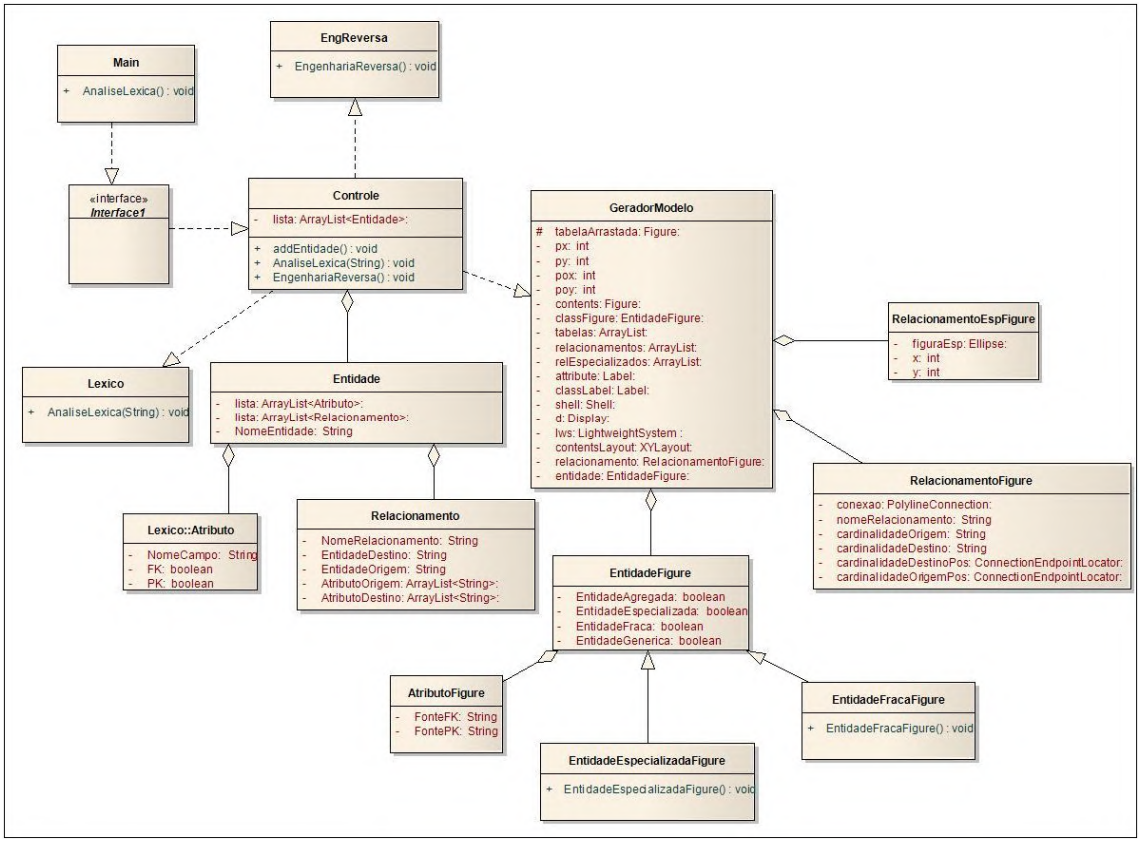


Figura 18. Diagrama de classe do sistema proposto

Conforme a Figura 18 segue uma breve descrição de cada classe.

a) interface: representa a interface gráfica principal, onde internamente e executada a classe Main;

b) controle: distribui as operações do sistema entre seus módulos;

c) lexico: interpreta o arquivo de entrada fazendo a leitura dos metadados caractere a carctere, os metadados são identificados e categorizados em Entidades, Atributos e Relacionamentos;

d) engReversa: esta classe representa o modulo de Engenharia Reversa onde são identificadas as Entidades que serão transformadas em Tabelas e as que serão transformadas em Relacionamentos, também são identificadas as cardinalidades dos Relacionamentos e os tipos de Relacionamentos: especialização, agregação, entidade fraca, entidade genérica;

e) entidade: representa o modelo de entidade, uma entidade por suas vez só existe caso existam atributos e relacionamentos integrados a ela;

f) atributo: esta classe representa o modelo de Atributo, um atributo pode ter um nome e ser uma chave estrangeira e/ou primária;

g) relacionamento: representa o modelo de Relacionamento, um relacionamento é uma restrição que possui um nome, uma Entidade de origem e destino, e um ou mais Atributos de origem e destino;

h) geradorModelo: esta classe gera o diagrama ER com base na estrutura de Entidades definidas pelo modulo lexico erRelacionamentos definidos pelo modulo engReversa. O diagrama é composto por Entidades renderizadas pela classe entidadeFigure e suas classes filhas, já os Relacionamentos renderizados pela classe relacionamentoFigure;

i) entidadeFigure: representa a interface gráfica de Entidade do diagrama ER;

j) atributoFigure: representa a interface gráfica de Atributos que compõe a interface de Entidade do Diagrama ER;

k) relacionamentoFigure: renderiza as conexões que compõe os relacionamentos entre as entidades;

l) relacionamentoEspFigure: renderiza a conexão referente a uma especialização;

m)entidadeEspecializadaFigure: constrói a interface gráfica de uma entidade especializada;

n) entidadeFraca: constrói a interface gráfica de uma entidade fraca.

6.1.3 Modelagem de Máquina de Estados

Por meio do diagrama de máquina de estados é possível representar as mudanças sofridas por uma determinada classe ou componente em tempo de execução. No caso do sistema proposto é relevante demonstrar graficamente como ocorre a interpretação dos comandos SQL na classe lexica conforme segue Figura 19.

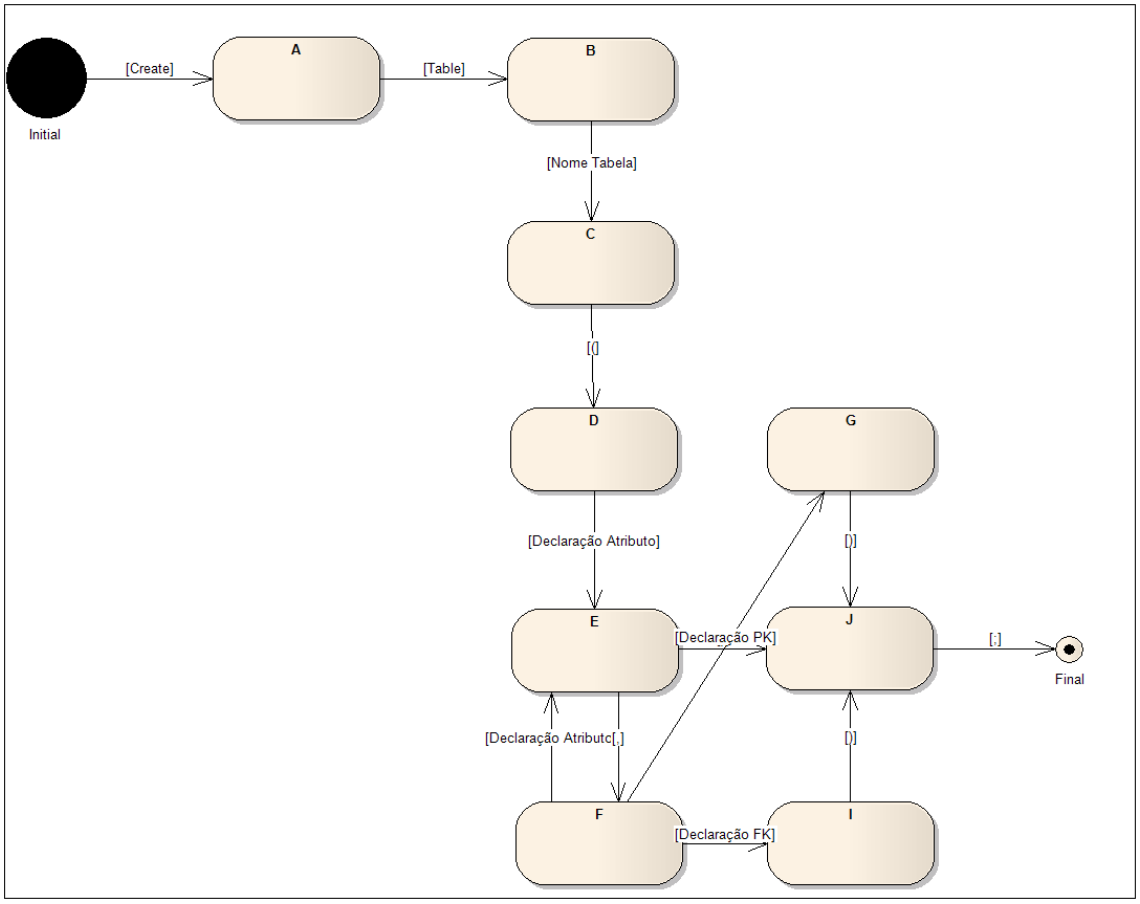


Figura 19. Diagrama de máquina de estados

Conforme Figura 19 é possível definir o seguinte fluxo de estados:

a) o diagrama inicia quando lê o comando create, no estado **A** ele aguarda pelo próximo comando;

b) ao ler o comando table reconhece a sentença create table e muda para o estado **B** indicado a criação de uma tabela;

c) recupera o nome da tabela e muda para o estado **C**;

d) interpreta o símbolo "(" abre parêntese, avança para o estado **D** indicando o inicio da estrutura de declaração da tabela;

e) identifica a declaração de atributo, segue para o estado **E** onde irá recuperar os atributos, caso o próximo comando lido seja “)” fecha parêntese indica o fim da estrutura de declaração da tabela muda para o estado **I**;

f) lê o comando “,” virgula e passa para o estado **F**, representando o fim da declaração de um atributo e inicio de uma nova declaração podendo retornar ao estado **E** ou ir para o estado **G** ou **H**;

g) identifica a declaração de chave primária e muda para o estado **G** onde recupera a chave primária, caso o próximo comando lido seja “)” fecha parêntese indica o fim da estrutura de declaração da tabela;

h) identifica a declaração de chave estrangeira e muda para o estado H onde recupera a chave estrangeira, caso o próximo comando lido seja “)” fecha parêntese indica o fim da estrutura de declaração da tabela muda para o estado **I**;

i) o estado **I** representa o fim da interpretação de declaração da tabela, demais seqüências de comandos são ignorados com exceção do “;” ponto e virgula que referencia o estado final do diagrama;

6.2 COMPONENTES DO SISTEMA

Os componentes do sistema são os módulos e entradas que compõe a arquitetura necessária para efetuar a ERBD. Sendo composto pelo Arquivo de Entrada e por quatro módulos: Módulo de Controle, Módulo Léxico, Módulo Entidade, Módulo de Engenharia Reversa e Módulo gerador de Visão.

6.2.1 Arquivo de Entrada (SQL)

Neste trabalho é utilizado o *backup* de um BD no formato *script* SQL como arquivo de entrada para que o Módulo Léxico faça uma análise das entidades e relacionamentos conforme Figura 20.

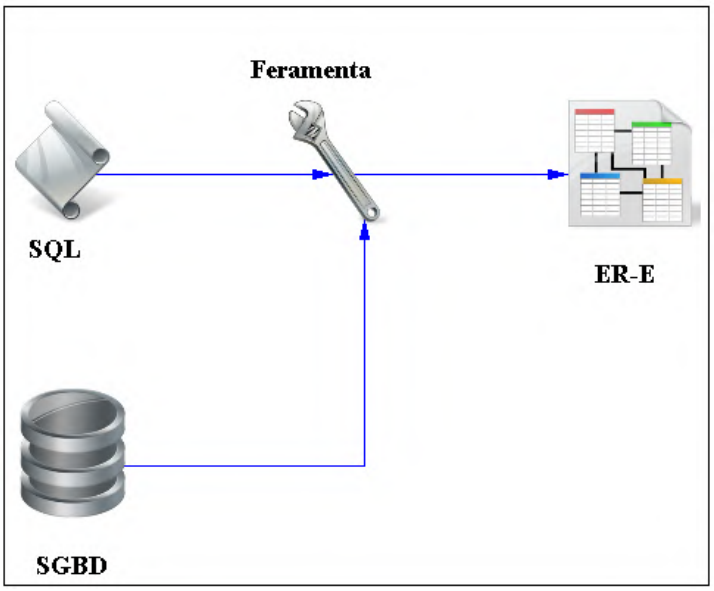


Figura 20. Obtenção do *script* SQL

O SQL teve sua primeira padronização oficial no ano de 1986 pela ANSI por meio do Comitê para banco de dados X3H2 (MANZANO, 2002). Este fato garantiu uma maior heterogeneidade nos SGBD e possibilitou um grande avanço no uso de *script* SQL, destacando-se a exportação do BD para *backup*. Dentre os metadados contidos no *script* SQL relevantes ao Módulo Léxico estão:

a) nome de Tabelas;

b) nome de Atributos;

c) nome de Relacionamentos;

d) definições de restrições.

Uma definição mais específica dos metadados interpretados pelo léxico é apresentada no Módulo Léxico.

6.2.1 Módulo de Controle

O Módulo de Engenharia Reversa gerencia as operações solicitadas pelo usuário final por meio de sua *interface* e distribui o processamento entre seus módulos. Por exemplo, o usuário final por meio da *interface seleciona* o caminho do arquivo de entrada e pressiona o botão 'executar Engenharia Reversa', ao pressionar o botão o Módulo de Controle passa o caminho do arquivo para o Módulo Léxico, ao termino do processo de interpretação do Módulo Léxico os dados obtidos são passados para o Módulo Entidades onde são armazenados.

6.2.2 Módulo Léxico

O Módulo Léxico atua sobre o arquivo de entrada atendendo a solicitação do Módulo de Engenharia Reversa, extraindo as entidades seus respectivos atributos e relacionamentos. Depois de mapeados os dados obtidos são traduzidos para um esquema de dados onde são categorizados.

Conforme Price e Toscani (2001) o objetivo principal de um analisador léxico é identificar seqüências de caracteres que constituem unidades léxicas *tokens*.

Por meio de uma analise de caractere a caractere são identificados às palavras reservadas, delimitadores, símbolos, identificadores contidas no alfabeto estipulado. O alfabeto estipulado não representa o conjunto de todos os comandos SQL e sim um subconjunto dos comandos SQL conforme Figura 21.

Figura 21. Conjunto do Alfabeto Estipulado

Como pode ser visto na Figura acima existe um grande conjunto que engloba todos os comandos de SQL, e dentro deste conjunto existe um subconjunto que representa o alfabeto estipulados nesta pesquisa conforme Figura 22.

Comando SQL	Símbolos SQL
CREATE	(
TABLE	)
PRIMARY	,
FOREING	;
KEY	'
ALTER	
CONSTRAINT	
REFERENCES	

Figura 22. Alfabeto estipulado

6.2.3 Módulo Entidade

No Módulo Entidade os dados são organizados e armazenados em uma classe de acordo com suas características mapeadas no Módulo Léxico. Em seguida os relacionamentos são armazenados na mesma classe, porém não serão analisados neste momento, a analise de relacionamentos ocorrerá no Modulo de Engenharia Reversa.

A partir dos dados mapeados para cada tabela é gerada uma lista de atributos, para cada atributo é especificado seu nome e seu tipo de definição: chave primária, chave

secundária e/ou não possui definição. Com base nos dados supracitados a classe Entidade é populada permitido a analise de sua estrutura.

6.2.4 Módulo de Engenharia Reversa

O Módulo de Engenharia Reversa faz uma analise prévia da estrutura de relações entre as tabelas analisando os atributos que possuem a definição de *primary key* e *foreing key* atribuindo a cada entidade uma ou mais relações. Com base nessas relações são definidas as tabelas e conexões no Módulo Gerador.

6.2.5 Módulo Gerador de Visão

Neste módulo é gerado o diagrama ER com base na estrutura de entidades e relacionamentos definidos previamente nos módulos anteriores. As entidades são renderizadas por meio de um container gráfico onde os atributos são adicionados, depois são criadas as conexões que representam os relacionamentos entre os containers.

6.3 RECURSOS UTILIZADOS

Para o desenvolvimento da ferramenta de ERBD foi utilizada a tecnologia Java, dentre os motivos da escolha desta tecnologia estão: o fato de Java ser livre e o incentivo de desenvolvimento e distribuição de *software* livre que a comunidade JavaTools representa.

O Ambiente de Desenvolvimento Integrado (IDE) utilizado foi o Netbeans 6.5.1, sua preferência deve-se ao fato de ser gratuito e um ambiente extremamente completo representando um grande auxílio ao desenvolvimento, disponível em http://www.netbeans.org/downloads/index.html. Além do Netbeans é necessário o Java Development Kit (JDK) 5.0 disponível em http://java.sun.com/javase/downloads/index_jdk5.jsp.

A interface geradora de modelos utiliza a biblioteca Draw2d para renderizar o diagrama ER, disponível em www.eclipse.org/gef. Também foi utilizado a biblioteca Standard Widget Toolkit (SWT) para gerar a janela onde o diagrama ER é renderizado disponível em http://www.eclipse.org/swt/.

Foram utilizados os SGBD Firebird e Oracle. O Firebird é um banco de dados relacional gratuito que oferece muitos recursos do padrão SQL e é compatível com diversos sistemas operacionais, a versão utilizada foi Firebird 2.0 disponível em http://www.firebirdsql.org/index.php?op=files (FIREBIRD, 2009) . Já o SGBD Oracle é um dos mais utilizados no mundo, sendo referência no mercado a pelo menos 30 anos, apesar de ser pago distribui uma versão *express* gratuitamente, também é compatível com diversos sistemas operacionais, a versão utilizada foi Oracle Database XE 10g Release 2 (10.2) disponível em http://www.oracle.com/technology/software/products/lite/index.html.

Para fazer a extração do *script* SQL foi utilizado as ferramentas SQL Manager for Firebird e Oracle versão trial disponível em http://sqlmanager.net/, para converter as bases foi utilizado a ferramenta Enterprise Convert trial disponível em http://www.spectralcore.com/fullconvert/.

6.3 ANÁLISE DOS RESULTADOS OBTIDOS

Com o intuito de validar o sistema proposto foram definidos os seguintes parâmetros de autenticação:

a) numero de tabelas identificadas corretamente;

b) numero de atributos identificados corretamente;

c) numero de relacionamentos identificados corretamente;

d) numero de entidades geradas no diagrama ER corretamente;

e) numero de relacionamentos gerados no diagrama ER corretamente.

Os teste foram rodados utilizando as bases Cinemas e Compras Web retiradas da apostila Banco de Dados I do professor Daniel Pezzi, vide Apêndices A e B.

6.3.1 Numero de Tabelas Identificadas Corretamente

O objetivo deste teste é verificar se o interpretador de SQL do simulador esta identificando as entidades para os SGBDs selecionados conforme Figura 23.

	BD Cinemas	BD ComprasWeb
Firebird	9	10
Oracle	9	10

Figura 23. Relação de tabelas interpretadas

6.3.2 Numero de Atributos Identificados Corretamente

O objetivo deste teste é verificar se o interpretador de SQL do simulador esta identificando os atributos para os SGBDs selecionados conforme Figura 24.

	BD Cinemas	BD ComprasWeb
Firebird	36	38
Oracle	36	38

Figura 24. Relação de atributos interpretadas

6.3.4 Numero de Relacionamentos Identificados Corretamente

O objetivo deste teste é verificar se o interpretador de SQL do simulador esta identificando os relacionamentos para os SGBDs selecionados conforme Figura 25.

	BD Cinemas	BD ComprasWeb
Firebird	8	10
Oracle	8	10

Figura 25. Relação de relacionamentos interpretadas

6.3.5 Numero de Entidades Geradas Corretamente

O Objetivo deste teste é verificar se a geração o simulador implementado esta gerando as entidades corretamente no diagrama ER para os SGBDs selecionados conforme Figura 26.

	BD Cinemas	BD ComprasWeb
Firebird	9	10
Oracle	9	10

Figura 26. Relação de identidades geradas

6.3.6 Numero de Relacionamentos Gerados Corretamente

O Objetivo deste teste é verificar se a geração o simulador implementado esta gerando os relacionamentos corretamente no diagrama ER para os SGBD selecionados conforme Figura 27.

	BD Cinemas	BD ComprasWeb
Firebird	8	10
Oracle	8	10

Figura 27. Relação de identidades geradas

Após feito os teste supracitados é possível verificar que o simulador se comportou da mesma maneira para ambos SGBD, os testes foram bem sucedidos nas duas bases especificadas gerando o diagrama ER corretamente, vide Apêndice C e D.

Com o termino da fase de teste pode se verificar que o objetivo geral de implementar uma ferramenta de apoio a engenharia reversa utilizando *script* SQL foi alcançado. O objetivo especifico do item **a** foi devidamente contemplado conforme os testes os dados do *script* SQL foram devidamente extraídos, já o objetivo especifico do item **b** foi parcialmente alcançado, pois foi possível utilizar dois SGBD distintos, no entanto era de grande interesse que se viabiliza-se para no mínimo quatro SGBD utilizando o mesmo algoritmo e por fim o objetivo do item **c** também foi alcançado com a implementação do analisador léxico.

CONCLUSÃO

Projetar BD tornou-se fundamental para seu sucesso, destacando-se a modelagem do projeto conceitual, que permite uma visão completa da estrutura lógica do banco de maneira simples, mas num alto nível de abstração de dados.

Esta pesquisa se subdividiu com base no cronograma proposto para melhor atender aos objetivos enunciados. Primeiramente ocorreu o estudo do projeto de banco de dados, tendo como grande contribuição o ciclo de vida do banco de dados, em seguida foi desenvolvido um estudo aprofundado no modelo Entidade Relacionamento, na terceira etapa foi elaborado uma pesquisa com ênfase na metodologia de Engenharia Reversa em Bancos de Dados, por fim foi elaborado o estudo do SQL bem como a implementação da ferramenta proposta.

O levantamento bibliográfico bem refinado possibilitou trazer diversas metodologias de engenharia reversa pouco conhecida no ambiente acadêmico, porem também evidenciou a necessidade de uma gama maior de pesquisas na área de ERBD em nosso país.

Foram encontrados diversos desafios no decorrer da pesquisa, dentre eles destaca-se os problemas com o gerador de modelos, devido a escassa documentação da biblioteca gráfica SWT, com poucos exemplos e abstrações diferenciadas de modelagem. Para resolver esta falha foram utilizados alguns tutorias e consultas a fórum e comunidades de programadores Java, no entanto a solução encontrada foi adaptar o modelo utilizado por Brognoli (2008), pois se trata de uma solução bem documentada de código aberto tendo como um de seus objetivos cooperar com o meio acadêmico.

Um fato positivo que possibilitou o desenvolvimento desta pesquisa foi o uso do script SQL, graças a existência do padrão SQL foi viável construir um analisador léxico que

interprete o código SQL em distintos SGBD, visto que caso não existi-se nenhum padrão e cada SGBD possui-se um código diferente inviabilizaria esta pesquisa.

Porém é relevante comentar que o padrão SQL não é amplamente respeitado, cada distribuidor de SGBD cria seu próprio padrão tendo como base o padrão definido pela ANSI, gerando diversos dialetos que dificultam a analise do script SQL. Outro problema encontrado é relacionado as ferramentas front, muitas delas tem sua própria estrutura de SQL, criando um dialeto diferenciado compatível apenas com determinado SGBD.

Como resultado da implementação da ferramenta proposta obteve-se compatibilidade com os SGBD Firebird e Oracle, devido aos problemas relacionados com o padrão SQL alguns SGBD podem vir a ser incompatíveis. Dessa forma é proposto como trabalho futuro a criação de um módulo de formatação do arquivo de entrada que seja compatível com a lógica definida no diagrama de máquina de estados da Figura 19.

Também são indicados os seguintes trabalhos futuros:

a) desenvolvimento de uma ferramenta de ERBD que utilize a extração de dados por XML;
b) desenvolvimeto de um comparador de metadados contidos no *script* SQL para comparar bases de dados;

REFERÊNCIAS

AVILA, M. L. ; MELLO, R. S. . **Uma Ferramenta de Apoio à Normalização de Tabelas Relacionais baseada na Análise de Dados**. In: IV Escola Regional de Banco de Dados, 2008, Florianópolis. ERBD 2008 - IV Escola Regional de Banco de Dados, 2008.

BROGNOLI, Samuel Nicolau. **FERRAMENTA DE APOIO A ENGENHARIA REVERSA DE BANCOS DE DADOS RELACIONAIS**. 2008. 87 f. Monografia (Bacharelado) - Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Cricíuma, 2008.

CHEN, Peter. **Modelagem de dados**: a abordagem entidade-relacionamento para projeto lógico. São Paulo: Makron Books, 1990.

CHIANG, Roger H. L.; BARRON, Terence M.; STOREY, Veda C. **Reverse engineering of relational databases**: Extraction of an EER model from a relational database. In: DATA & KNOWLEDGE ENGINEERING, 1, 1994, Amsterdam. Reverse engineering of relational databases. Amsterdam: Elsevier Science Publishers, 1996. v. 21, p. 57 - 77.

DATE, C. J. **Introdução a sistemas de bancos de dados**. 4. ed. Rio de Janeiro: Campus, 2004.

DAVIS, K. H.; AIKEN, P. H. Data **Reverse Engineering**: A Historical Survey. In: WORKING CONFERENCE ON REVERSE ENGINEERING, 2000. Procedings... [S.1: s.n],2000.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de bancos de dados**: Fundamentos e aplicações. 3. ed. Rio de Janeiro: LTC, 2002.

GARCIA-MOLINA, Hector; ULLMAN, Jeffrey D.; WIDOM, Jennifer. **Database System Implementation.** New Jersey: Prentice Hall, 1999.

HAINAUT, Jean-luc. **Introduction to Database Reverse Engineering**. Namur: Libd, 2002.

HARRINGTON, Jan L. **Projetos de bancos de dados relacionais.** Rio de Campus: Campus, 2002.

HERNANDEZ, Michael J. **Database design for mere mortals**: a hands-on guide to relational database design. Boston: Addison-wesley, 2003.

HEUSER, Carlos Alberto. **Projeto de banco de dados**. 4. ed. Porto Alegre: Sagra Luzzatto, 2001.

JOHANNESSON, Paul. **A Method for Transforming Relational Schemas Into Conceptual Schemas**. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 10, 1994, Washington. International Conference on Data Engineering. Washington: IEEE Computer Society, 1994. p. 190 - 201.

LIGHTSTONE, Sam; TEOREY, Toby; NADEAU, Tom. **Physical Database Design**: The Database Professional's Guide to Exploiting Indexes, Views, Storage, and More. San Francisco: Morgan Kaufmann, 2007.

MARKOWITZ, V; MAKOWSK, J. **Identifying extended entityrelationship object structures in relational schemas**. IEEE Transactions on Software Engineering, 16(8), Aug. 1990. p. 777 - 790.

MACHADO**,** Felipe Nery Rodrigues; ABREU, Maurício Pereira de. **Projeto de banco de dados**: uma visão prática. 8. ed. São Paulo: Érica, 2002**.**

MANZANO, José Augusto N. G. **Estudo dirigido**: SQL**.** São Paulo: Érica, 2002.

MANNINO, Michael V. **DATABASE DESIGN, APPLICATION DEVELOPMENT, AND ADMINISTRATION**. 2. ed. New York: Mcgraw-hill, 2007.

MATTÉ, Lia Cláudia. **Conjunto de Procedimentos de Engenharia Reversa para o Projeto de Banco de Dados Espaciais**. 2002. 107 f. Dissertação (Mestre) - Curso de Ciência da Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2002.

MULLER, Robert J. **Database Design for Smarties**: Using UML for Data. San Francisco: M Organ Kaufmann, 1999.

NOBRE, Miguel K.; SOUZA, Marcelo Caon. **Ferramenta de apoio a Engenharia Reversa de um Banco de Dados Relacional**. 2005. 54 f. Monografia (Bacharelado em Sistemas de Informação) Universidade Federal de Santa Catarina, Florianópolis.

OLSON, Jack E. **Database Archiving**: How to Keep Lots of Data for a Very Long Time. New York: Morgan Kaufmann, 2008.

PEDRO-DE-JESUS, Maria de Lurdes; SOUSA, Pedro Manuel Antunes. **Selection of Reverse Engineering Methods for Relational Databases**. In: SOFTWARE MAINTENANCE AND REENGINEERING, 3, 1999, Amsterdam. Selection of Reverse Engineering Methods for Relational Databases. Washington: Ieee Computer Society, 1999. p. 194 - 197.

PETIT, Jean-marc et al. **Towards the Reverse Engineering of Denormalized Relational Databases**. In: INTERNATIONAL CONFERENCE ON DATA ENGINEERING, 12, 1996, Washington. Towards the Reverse Engineering of Denormalized Relational Databases. Washington: Ieee Computer Society, 1996. p. 218 - 227.

POWELL, Gavin. **Beginning Database Design**. Indianapolis: Wiley, 2006.

PREMERLANI, William J; BLAHA, Michael R.. **An approach for reverse engineering of relational databases**. In: COMMUNICATIONS OF THE ACM, 5, 1994, New York. An approach for reverse engineering of relational databases. New York: Acm, 1994. v. 37, p. 42 - ff.

PRESSMAN, Roger S. **Software Engineering**: APRACTITIONER'S APPROACH. 5. ed. New York: Mcgraw-hill, 2000.

PRICE, Ana Maria de Alencar; TOSCANI, Simão Sirineo. **Implementação de linguagens de programação: compiladores.** Porto Alegre: Sagra Luzzatto, 2000.

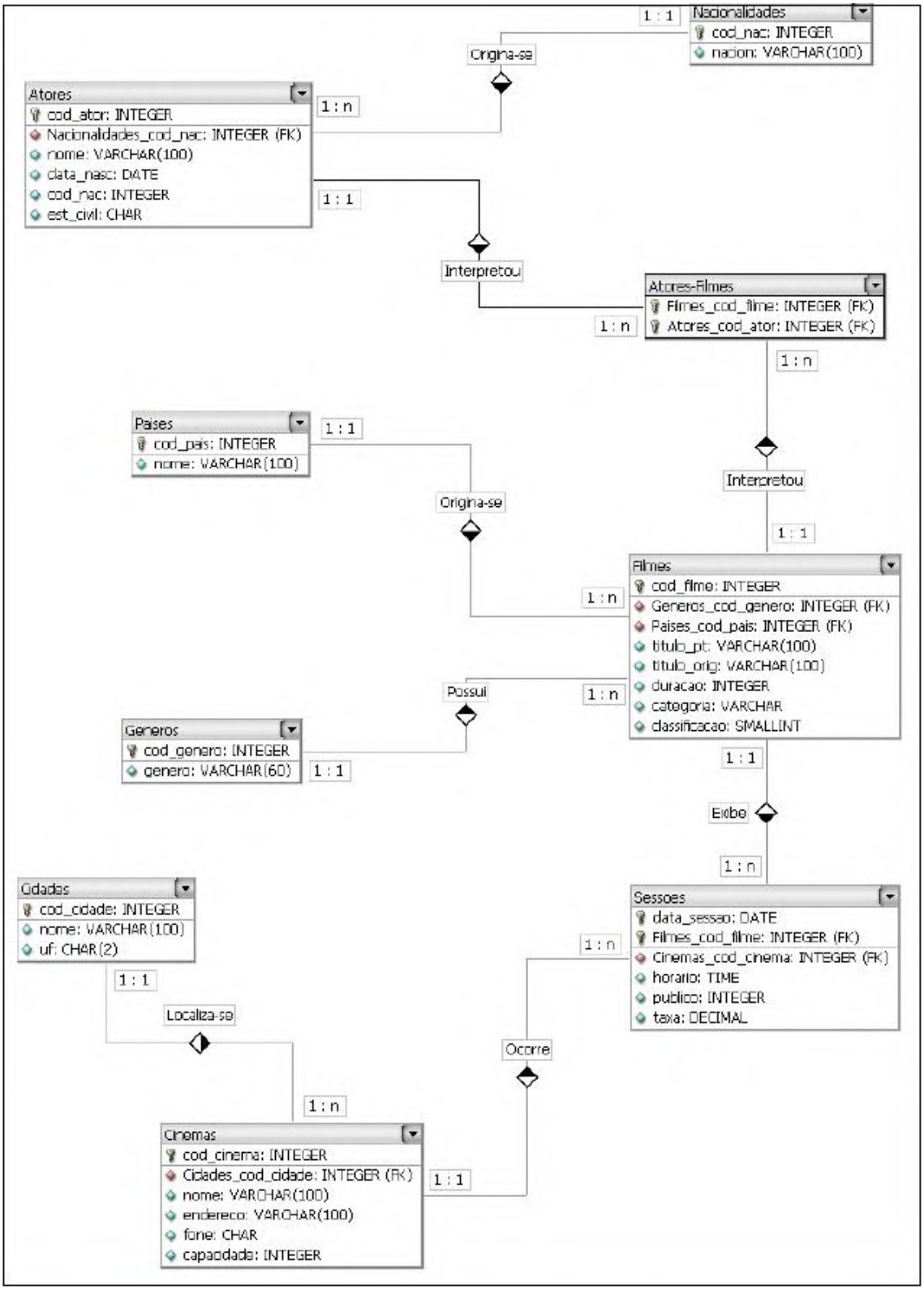
RAMAKRISHNAN, Raghu; GEHRKE, Johannes. **Database Management Systems.** 2. ed. New York: Mcgraw-hill, 2000.

ROB, Peter; CORONEL, Carlos. **Database Systems**: Design, Implementation and Management. 6. ed. Boston: The Wadsworth, 2005.

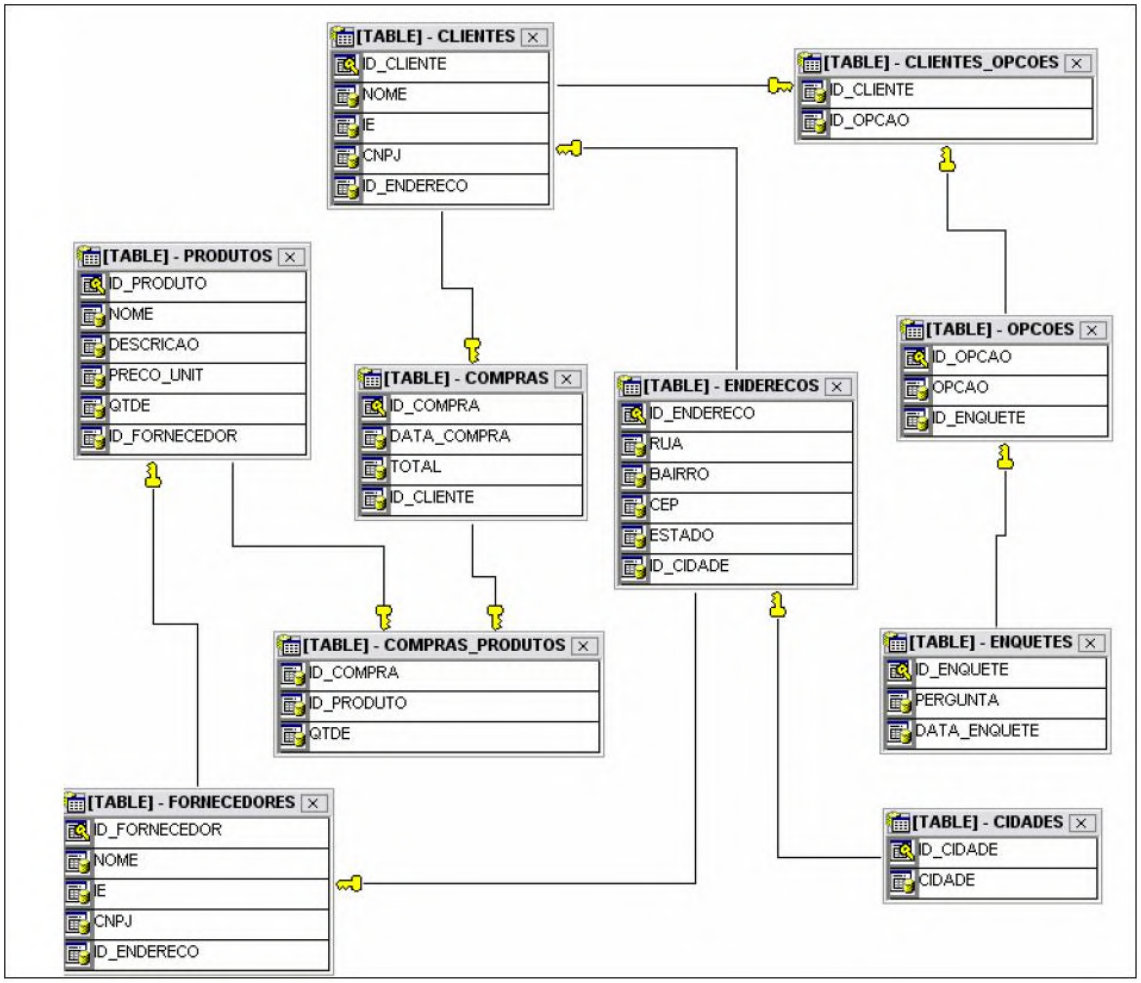
SIGNORE, O et al. **Using procedural patterns in abstracting relational schemata**. In: PROGRAM COMPREHENSION, 3., 1994, Washington. Using procedural patterns in abstracting relational schemata. Washington: Ieee Computer Society, 1994. p. 128 - 135.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S. **Sistema de banco de dados**. 5. ed. Rio de Janeiro: Elsevier, 2006.

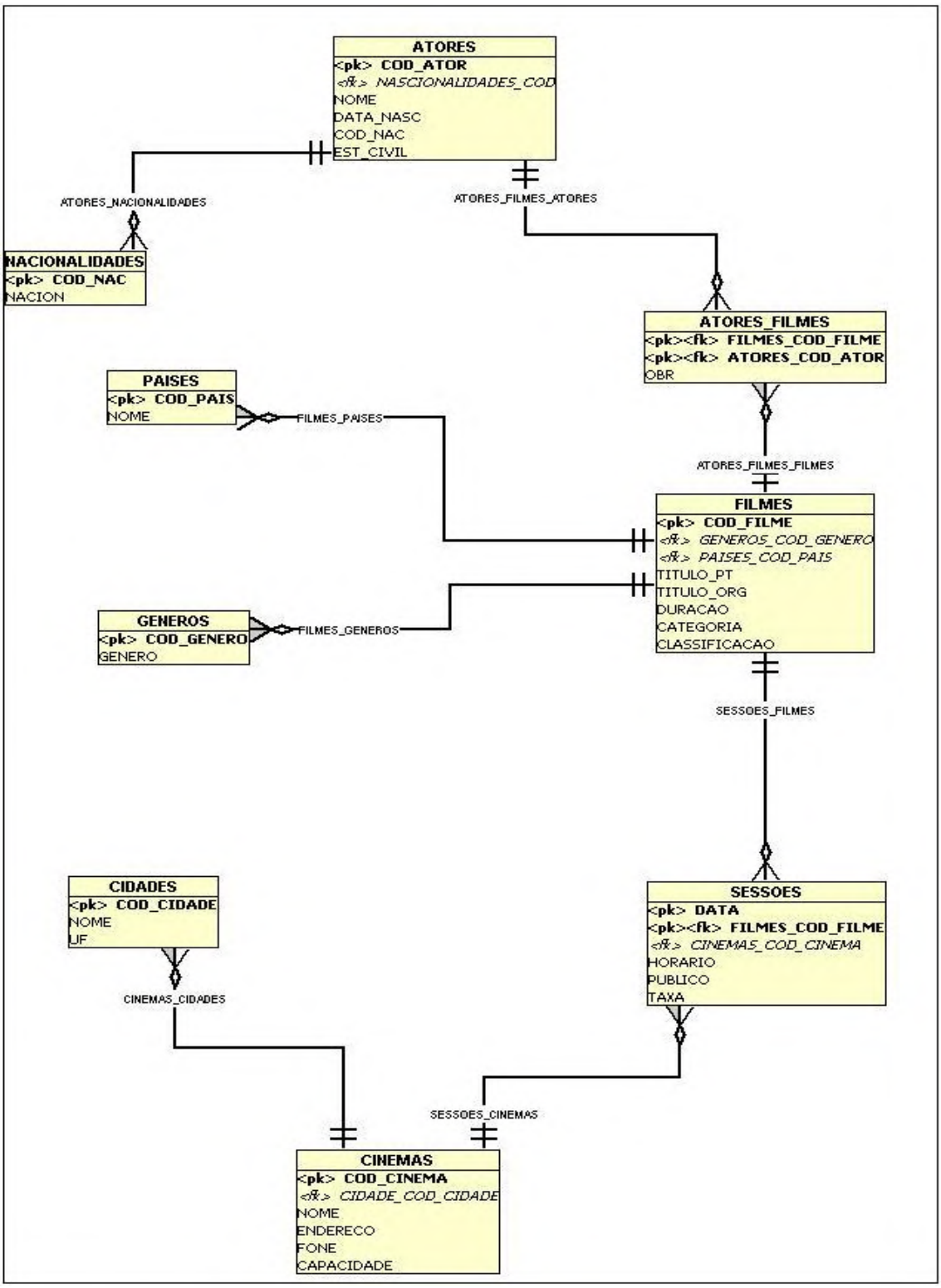
APÊNDICE A – DIAGRAMA ER DO BANCO DE DADOS CINEMAS



APÊNDICE B – DIAGRAMA ER DO BANCO DE DADOS COMPRAS WEB



APÊNDICE C – DIAGRAMA ER GERADO PELA FERRAMENTA PROPOSTA DO BANCO DE DADOS CINEMAS



APÊNDICE D – DIAGRAMA ER GERADO PELA FERRAMENTA PROPOSTA DO BANCO DE DADOS COMPRAS WEB