

# O Modelo Universal e a abstração radical de dados

Fábio Medeiros Nasário (UNIVALI)

fabio@dpfsc.gov.br

Vinícius Medina Kern (UNIVALI e PPGE/UFSC)

kern@eps.ufsc.br

Eugênio Rubens Cardoso Braz (UNIVALI e PPGE/UFSC)

ercb@eps.ufsc.br

**Resumo.** A literatura apresenta, sem documentação suficiente, um Modelo Universal de Dados que permite a implementação de qualquer banco de dados através de um conjunto de sete relações. Este modelo é, na verdade, uma abstração de dados e metadados. Procuramos desvendar o Modelo Universal fazendo uma engenharia reversa de um esquema de banco de dados, produzindo uma abstração de uma abstração de dados – uma estrutura relacional normalizada de altíssimo nível de abstração. Aperfeiçoamos esta abstração para que contemple também dados operacionais e ilustramos seu funcionamento comparando acessos a banco de dados utilizando expressões SQL tradicionais e acessos segundo este novo Modelo Universal. Concluímos com recomendações para extensão desta pesquisa.

**Palavras-chave:** Modelo Universal de Dados, projeto de banco de dados, modelo físico de dados

## 1 Introdução

Hay (1999) apresenta um Modelo Universal de Dados (MUD), como pode ser visto na figura 1, mas não documenta suficientemente seu significado. Segundo o autor, este modelo abrange qualquer tipo de dado existente no Universo. Hay também não explica que, na verdade, o MUD representa um modelo interno de banco de dados que permite manipular dados e metadados com o mesmo tipo de ferramenta.



Figura 1 - Modelo Universal de Dados ou MUD (Hay 1999)

Nenhum dos textos mais conhecidos de teoria de bancos de dados menciona idéia semelhante à do MUD. Os bons textos disponíveis em português, como Date (2000) e Silberschatz, Korth e Sudarsan (1999), ou em inglês, como Ramakrishnan (2000) e Elmasri e Navathe (1994), não discutem a possibilidade de implementar um banco de dados qualquer com poucas tabelas. A idéia, no entanto, é pertinente, pois o MUD nada mais é do que um esquema interno de banco de dados, segundo a arquitetura ANSI/SPARC (Tsichritzis e Klug 1978). O MUD representa, simultaneamente, uma abstração de dados e metadados.

Para compreender o MUD é necessário entender o significado da expressão “abstração de dados e metadados”. A manipulação de um banco de dados é feita através de um modelo que abstrai os dados, usando um sistema de gerência de banco de dados (SGBD). Este modelo é a abstração de dados, que compreende entidades, relacionamentos, atributos, chaves e outras restrições.

SGBDs são softwares genéricos construídos para manter dados e abstração de dados de qualquer banco de dados específico. Sendo assim, é necessário que o SGBD seja capaz de manipular qualquer abstração de dados (desconhecida previamente). Portanto, o modelo interno do banco de dados deve conter uma abstração (genérica) das abstrações de dados (específicas) dos bancos de dados que vier a manipular. Esta abstração das abstrações de dados, ou abstração de metadados, corresponde a um modelo interno de banco de dados.

Baseado na proposta de Hay (1999), o MUD é um modelo de abstração que pode ser entendido como um esquema que possui um conjunto de relações (ou tabelas) no qual se consegue abstrair tanto os dados quanto os metadados. Por exemplo, em uma relação para representar pessoas com número de registro e nome, são guardados no MUD a descrição da relação “Pessoa”, com os seus atributos “Registro” e “Nome” (metadados) e também os dados de registro e nome das pessoas que, na concepção tradicional, estariam cadastrados na tabela “Pessoa”. Tal organização relaciona-se com a proposta de Codd (1985, 1985b) segundo a qual um sistema, para ser considerado relacional, precisa permitir a manipulação de dados e metadados com a mesma ferramenta.

Os sistemas de gerência de banco de dados (SGBD) atuais utilizam-se de uma espécie de “MUD” próprio para definir (descrever) suas tabelas, ligações, chaves, etc. Cada modelo interno de SGBD possui características diferenciadas. No SGBD Oracle®, por exemplo, a consulta “SELECT table\_name FROM user\_tables;” retorna os nomes das tabelas existentes – ou seja, é uma consulta à tabela de tabelas.

Apesar das características próprias de cada SGBD, todos os modelos internos ou “MUDs” se assemelham em sua função, que é a da representação interna (física) dos dados e modelos de dados construídos no SGBD. Será que é possível conceber a existência de um único MUD, para ser utilizado por todos os SGBDs? Levando em conta que o MUD da figura 1 não foi suficientemente detalhado, este artigo propõe reconstruí-lo a partir da abstração de uma abstração de dados, sem a obrigatoriedade de se reconstruir um modelo idêntico ao de Hay.

A idéia é, a partir de um exemplo de modelo de banco de dados, fazer uma engenharia reversa, abstraindo um meta-esquema a partir de um esquema de definição de dados. Em seguida, o meta-esquema será normalizado, resultando em um novo esquema, em nível de abstração superior ao do esquema original. Algumas modificações no esquema são feitas para contemplar restrições adicionais, evoluindo para o resultado final, o MUD. Finalmente, ilustramos a manipulação de dados e metadados em SQL no MUD que derivamos.

## 2 Obtenção do meta-esquema

A idéia de modelar em níveis de abstração variados já podia ser observada no trabalho de Hay e Healy (1997). As figuras 2 e 3 ilustram modelos representados em IDEF1X (NIST 1993) para o mesmo ambiente de negócios: uma organização composta por vários tipos de estrutura organizacional, com níveis hierárquicos diferenciados (no caso, filiais e departamentos, sendo as filiais superiores aos departamentos).

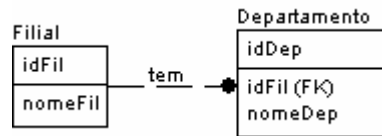


Figura 2 – Modelo IDEF1X de filiais e departamentos de uma empresa

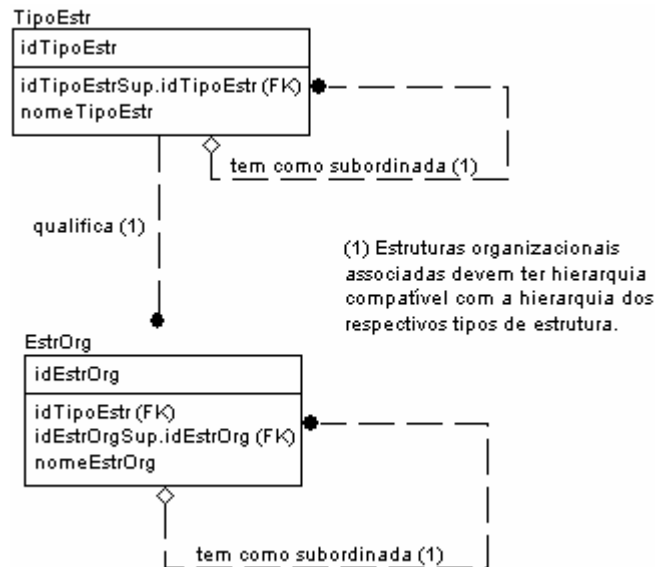


Figura 3 - Abstração da abstração de dados da fig. 2 (adaptado de Hay e Healy (1997))

Cada filial pode conter vários departamentos. Na figura 2, os atributos “idFil” e “idDep” equivalem às chaves primárias, que identificam cada ocorrência em cada relação. Já “nomeFil” e “nomeDep” são atributos para designar os nomes de filiais e departamentos, respectivamente. Na figura 3, todos os códigos e nomes de estruturas organizacionais, sejam filiais ou departamentos, estão representados por “idEstrOrg” e “nomeEstrOrg”, respectivamente. Cada estrutura é de um certo tipo de estrutura (TipoEstr), e pode ter uma estrutura hierarquicamente superior.

Hay e Healy (1997) propunham flexibilizar um modelo de dados como o da figura 2 aumentando o nível de abstração e permitindo que mudanças organizacionais (por exemplo, a inclusão de um tipo de estrutura intermediária entre filial e departamento) não requeressem alteração de modelo, mas apenas de instâncias do banco de dados.

Por exemplo, se a estrutura organizacional “Divisão” for incluída entre “Filial” e “Departamento”, o modelo da figura 2 deverá ser alterado para contemplar esta alteração. No entanto, no modelo da figura 3, as divisões podem ser cadastradas como instâncias da entidade EstrOrg (estrutura organizacional), desde que as instâncias de EstrOrg e TipoEstr (tipo de estrutura) sejam alteradas para refletir a nova organização Filial-Divisão-Departamento. Neste caso, nenhuma alteração de modelo é necessária.

Este modelo é limitado à abstração de hierarquias organizacionais. No entanto, é possível abstrair os dados de um ambiente de negócios de forma tão genérica que qualquer ambiente resulta na mesma abstração. A demonstração desta abstração genérica inicia pelo código SQL de definição de dados mostrado na figura 4, criado a partir do modelo da figura 2.

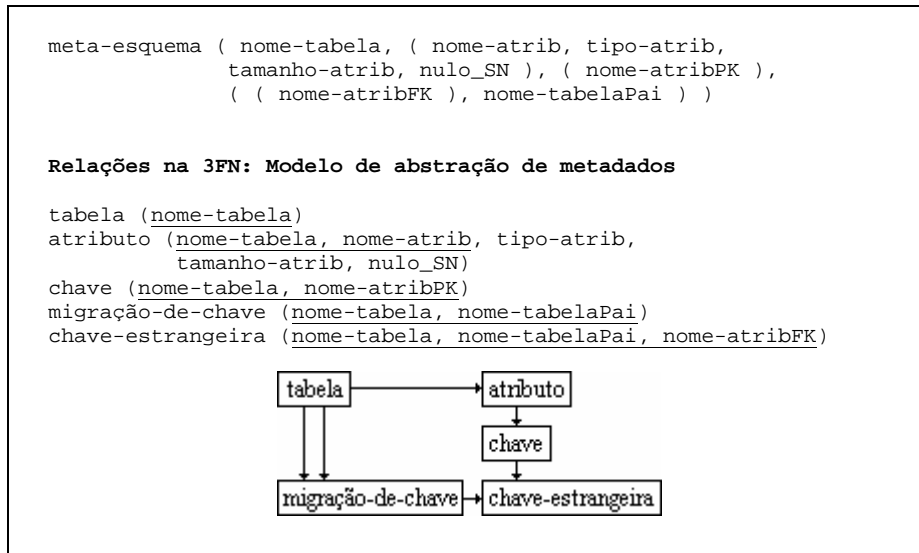
```
/* Esquema "Estrutura organizacional" */  
  
CREATE TABLE Filial (  
    idFil                INTEGER NOT NULL,  
    nomeFil              VARCHAR(20),  
    PRIMARY KEY (idFil) );  
  
CREATE TABLE Departamento (  
    idDep                INTEGER NOT NULL,  
    idFil                INTEGER NOT NULL,  
    nomeDep              VARCHAR(20),  
    PRIMARY KEY (idDep),  
    FOREIGN KEY (idFil) REFERENCES Filial );
```

**Figura 4 – Esquema em SQL para o modelo da figura 2**

Neste esquema existem relações ou tabelas (“Filial” e “Departamento”), atributos (“idFil”, “nomeFil”, “idDep”, etc.), um relacionamento (no qual cada “Departamento” faz referência a uma “Filial”), chaves, etc. Estas relações, atributos, chaves, etc. são metadados que abstraem os dados que ficarão contidos no banco de dados. Como a proposta do MUD é abstrair também os metadados, então é necessário criar outros metadados para abstrair estas relações, atributos,

chaves, etc. Isto é o que denominamos abstração da abstração, ou abstração de metadados.

A figura 5 apresenta um esquema baseado nesta idéia de abstração. Apresenta-se o meta-esquema (abstração da abstração da figura 4) e sua normalização conforme a terceira forma normal (3FN), bem como uma representação gráfica na notação de Bachman (1969). As relações ou tabelas resultantes são: “tabela”, “atributo”, “chave”, “migração-de-chave” e “chave-estrangeira”. Estas estruturas representam, numa visão de altíssimo nível, a abstração de um esquema de banco de dados relacional qualquer, ou uma abstração de uma abstração de dados qualquer.



**Figura 5 - Meta-esquema abstraído a partir do modelo das figuras 2 e 4, relações derivadas segundo a terceira forma normal e respectiva representação gráfica**

Este meta-esquema abstrai somente os metadados; não consegue abstrair os próprios dados. Por exemplo, ele consegue guardar a definição de uma relação de pessoas com seus atributos e restrições, mas não consegue guardar os dados das próprias pessoas. Para que esta e outras necessidades possam ser atendidas, promovemos algumas alterações no meta-esquema, na próxima seção.

### 3 O MUD reconstruído – aperfeiçoando o meta-esquema

A figura 6 apresenta o MUD reconstruído com a inclusão da tabela “Valor”, permitindo o armazenamento dos valores das instâncias de atributos. Qualquer dado em um banco de dados deve ser acessável através da identificação da tabela,

da linha ou tupla (identificada pelo valor da chave registrado em “descValChave”) e do atributo ou coluna da tabela que se pretende acessar.

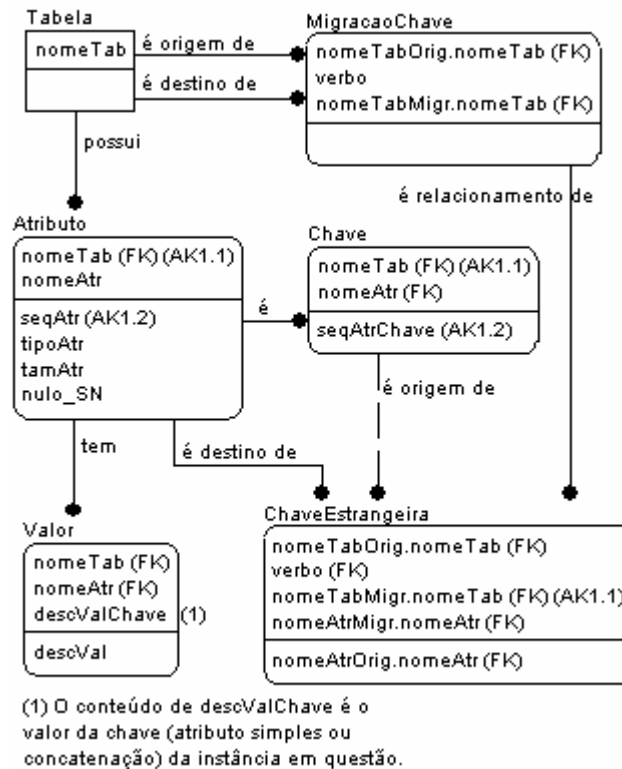


Figura 6 – MUD baseado num aperfeiçoamento do meta-esquema da figura 5

Também são feitas algumas modificações sobre a parte estrutural já contida no metamodelo da figura 5, em função de algumas restrições importantes para a manipulação e a gerência do acesso a bancos de dados:

- Em “Atributo”: inclusão de “seqAtr” (número de ordem seqüencial do atributo na tabela), pois os atributos em uma tabela têm ordem.
- Em “Chave”: inclusão de “seqAtrChave” (número de ordem seqüencial do atributo na chave), pois a ordem dos atributos na chave não segue necessariamente a mesma ordem dos atributos na tabela.
- Em “MigracaoChave”: inclusão de “verbo” (verbo transitivo que permite ler o relacionamento como uma sentença), pois é possível que duas tabelas estejam associadas por mais de um relacionamento, mas os verbos que permitem ler os relacionamentos são necessariamente distintos.

- Em “ChaveEstrangeira”: inclusão do relacionamento “Atributo é destino de ChaveEstrangeira”, pois um atributo chave estrangeira faz referência a uma chave, mas seu nome não é necessariamente uma cópia do nome da chave que referencia, portanto é necessário manter informação sobre o nome do atributo na tabela onde é chave estrangeira. É uma forma de explicitar os nomes original e migrado do atributo que participa de chave estrangeira.

Estas duas últimas alterações permitem que se implemente, em uma mesma tabela, mais de uma chave estrangeira da mesma tabela original (neste caso, usando diferentes nomes de papel – “nomeAtrOrig” e “nomeAtrMigr” – para o nome do atributo, “nomeAtr”, que migra a partir de “Atributo”). Desta maneira, “nomeAtrOrig” é o nome do atributo chave primária na tabela original, e “nomeAtrMigr” é o nome do atributo migrado que funciona como chave estrangeira. Os nomes de atributo original e migrado podem coincidir, ou o segundo pode ser um nome de papel diferente do original.

#### 4 Ilustração de funcionamento do MUD

A figura 7 mostra, de forma esquemática, uma estrutura organizacional composta de filiais e departamentos, compatível com o modelo da figura 2. A figura 8 exibe esta estrutura organizacional armazenada segundo o MUD. Os dados em si são armazenados na tabela “Valor”, enquanto a abstração de dados está contida nas demais estruturas.

1 Filial São José	2 Filial Blumenau
101 Dep Compras	103 Dep Produção
102 Dep Vendas	104 Dep Compras

**Figura 7 – Representação esquemática de uma estrutura organizacional**

Para ilustrar o funcionamento do MUD, apresenta-se a seguir um comparativo utilizando consultas em linguagem SQL sobre um banco de dados com projeto tradicional e sobre um banco de dados segundo o MUD. O primeiro exemplo trata da definição de dados tradicional, conforme a figura 9, mostrando o uso do comando CREATE TABLE para criar uma relação de filiais no modelo da figura 2.

Atributo					
nomeTab	nomeAtr	seqAtr	tipoAtr	tamAtr	nulo_SN
Filial	idFil	1	INTEGER	null	N
Filial	nomeFil	2	VARCHAR	20	N
Departamento	1	idDep	INTEGER	null	N
Departamento	2	idFil	INTEGER	null	N
Departamento	3	nomeDep	VARCHAR	20	N

Tabela		Chave		
nomeTab		nomeTab	nomeAtr	seqAtrChave
Filial		Filial	idFil	1
Departamento		Departamento	idDep	1

MigracaoChave		
nomeTabOrig	verbo	nomeTabMigr
Filial	tem	Departamento

ChaveEstrangeira				
nomeTabOrig	verbo	nomeTabMigr	nomeAtrMigr	nomeAtrOrig
Departamento	tem	Filial	idFil	idFil

Valor			
nomeTab	nomeAtr	descValChave	descVal
Filial	idFil	1	1
Filial	nomeFil	1	Filial São José
Filial	idFil	2	2
Filial	nomeFil	2	Filial Blumenau
Departamento	idDep	101	101
Departamento	idFil	101	1
Departamento	nomeDep	101	Dep. Compras
Departamento	idDep	102	102
Departamento	idFil	102	2
Departamento	nomeDep	102	Dep. Vendas
Departamento	idDep	103	103
Departamento	idFil	103	3
Departamento	nomeDep	103	Dep. Produção
Departamento	idDep	104	104
Departamento	idFil	104	4
Departamento	nomeDep	104	Dep. Compras

Figura 8 - Tabelas de um banco de dados baseado no MUD e no modelo da figura 2

```

CREATE TABLE Filial (
    idFil          INTEGER NOT NULL,
    nomeFil       VARCHAR(20),
    PRIMARY KEY (idFil)
);

```

Figura 9 – Comando em SQL para criar tabela e atributos em sistemas tradicionais

A figura 10 mapeia o comando CREATE TABLE para o MUD, resultando em inserções nas tabelas “Tabela”, “Atributo”, e “Chave” conforme o modelo da

figura 6 e o banco de dados da figura 8. Outras instruções CREATE TABLE podem requerer, também, inserções nas tabelas “MigracaoChave” e “ChaveEstrangeira”.

```
INSERT INTO Tabela VALUES ('Filial');
INSERT INTO Atributo VALUES ('Filial', 'idFil', 1, 'INTEGER', null,
'N');
INSERT INTO Atributo VALUES ('Filial', 'nomeFil', 2, 'VARCHAR', 20,
'N');
INSERT INTO Chave VALUES ('Filial', 'idFil', 1);
```

**Figura 10 – Comandos em SQL para criar relação e atributos no MUD**

A figura 11 mostra uma consulta à tabela “Filial” conforme o modelo da figura 2. Trata-se de uma consulta simples às informações da tabela.

```
SELECT idFil, nomeFil
FROM Filial ;
```

**Figura 11 – Consulta à tabela de filiais conforme o modelo da figura 2**

Esta mesma consulta sobre um banco de dados cujo modelo é o MUD tem sua tradução na figura 12. A consulta é apresentada na forma de visão, o que permite apresentar a consulta de forma idêntica à consulta tradicional, cujo resultado é exibido na figura 13.

```
CREATE VIEW V_select_Filial AS
SELECT idF.descVal as idFil, nomeF.descVal as nomeFil
FROM Valor idF, Valor nomeF
WHERE
    idF.nomeTab = nomeF.nomeTab and
    idF.descValChave = nomeF.descValChave and
    idF.descVal = nomeF.descValChave and
    idF.descVal <> nomeF.descVal ;
```

**Figura 12 – Comandos SQL para listar “Filial”, no MUD**

idFil	nomeFil
1	Filial São José
2	Filial Blumenau

Figura 13 – Ilustração do resultado da consulta da figura 12

Como apreciação geral dos mapeamentos, pode-se observar que o nível de complexidade das expressões é maior do que no caso tradicional. No entanto, com o MUD é possível manipular dados e metadados usando a mesma ferramenta, no caso a linguagem SQL.

## 5 Conclusões

Neste artigo relatamos a apresentação, por Hay (1999), de um MUD cuja documentação é deficiente. A seguir, reconstruímos um MUD a partir de uma abstração de uma abstração de dados. Finalmente, ilustramos o funcionamento de um banco de dados manipulado através do MUD, que pode ser encarado como uma representação interna de modelo de banco de dados segundo a arquitetura ANSI/SPARC (Tsichritzis & Klug 1978).

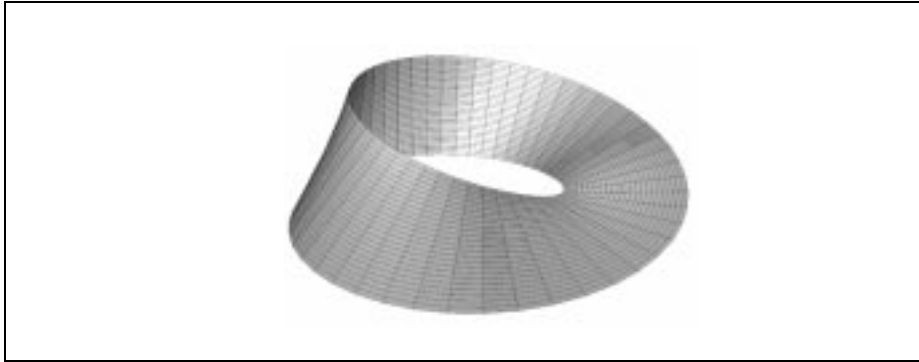
Os mapeamentos e ilustrações de funcionamento da seção 4 são construções ad hoc. A pesquisa está em desenvolvimento; o desenho do MUD pode ainda ser alterado em função dos resultados futuros. Porém, a disponibilidade de um MUD já demonstra potenciais vantagens – por exemplo, a de permitir a implementação de sistemas com grande flexibilidade, incluindo a capacidade de permitir alterações na estrutura da informação em tempo de execução, já que qualquer alteração estrutural requer simplesmente inserções, alterações e exclusões de registros. Também, o MUD pode ser utilizado como instrumento pedagógico para auxiliar a compreensão de modelos físicos de dados.

As comparações ou mapeamentos apresentados na seção 4 permitem vislumbrar este caráter pedagógico. O MUD é uma representação do modelo interno de dados e pode ser mapeado a partir do modelo de nível conceitual, como previa a arquitetura ANSI/SPARC (Tsichritzis & Klug 1978).

Hay (1999) menciona a tira de Möbius<sup>1</sup> para sugerir uma ilustração do que acontece com um banco de dados implementado segundo o MUD. O mapeamento de instruções de definição e manipulação de dados entre o modelo conceitual tradicional e o modelo interno (o MUD) mostrados na seção 4 parece exigir uma flexão da linguagem SQL que se compara à tira de papel que se dobra sobre si própria, na representação da tira de Möbius da figura 14. O mapeamento requer escrever, em SQL, as instruções que permitem consultar “tabelas” cuja estrutura está armazenada no próprio conjunto de tabelas que compõe o MUD.

---

<sup>1</sup> August Ferdinand Möbius, ou Moebius, matemático saxão (1790-1878), autor da equação que descreve a tira que só tem um lado e uma borda. Outras informações podem ser encontradas em <http://www-gap.dcs.st-and.ac.uk/~history/Mathematicians/Mobius.html>.



**Figura 14** - Tira de Möbius

A ilustração de funcionamento que apresentamos é apenas introdutória. O MUD presta-se a demonstrar várias características de sistemas de banco de dados, como a manutenção da integridade da entidade e da integridade referencial. Estas demonstrações devem ser objeto de futuros desenvolvimentos, a partir dos resultados relatados neste artigo.

Outro desdobramento previsto para esta pesquisa é a aplicação do MUD como base para sistemas de informações com estrutura configurável em tempo de execução. Aplicações de portais web podem beneficiar-se desta abordagem, se o desenvolvedor de aplicativo dispuser de um módulo que automatize o mapeamento entre o MUD e a maneira usual-tradicional de modelar e manipular bancos de dados.

Outra possibilidade interessante do MUD é sua realização em XML. Uma vez que um só esquema representa a abstração de dados e metadados e um mesmo conjunto de tabelas pode armazenar um banco de dados e o catálogo deste banco de dados, pode-se vislumbrar a realização do MUD através de uma especificação XML Schema (W3C 2002) e um arquivo XML. Este desenvolvimento é uma etapa futura desta pesquisa, com grande potencial de aplicação na integração de informações de fontes variadas.

## **6 Referências**

- BACHMAN, C. W. "Data Structure Diagrams". **Data Base** 1 (2), pp. 4-10, March 1969.
- CODD, E.F. "Does Your DBMS Run By The Rules?" **Computerworld**, Oct. 21, 1985.
- CODD, E.F. "Is Your Database Really Relational?" **Computerworld**, Oct. 14, 1985b.
- DATE, C. J. **Introdução a sistemas de bancos de dados**. Tradução da

- 7ª edição norte-americana. Rio de Janeiro: Campus, 2000.
- ELMASRI, R.; NAVATHE, S. **Fundamentals of database systems**. Addison-Wesley, 1994.
- HAY, D. **Princípios de modelagem de dados**. São Paulo: Makron Books, 1999.
- HAY, D. & HEALY, K. A. **GUIDE Business Rules Project**. Final Report, revision 1.2, October 1997.
- NIST (National Institute of Standards and Technology), Federal Information Processing Standards Publication 184: **Integration Definition for Information Modeling (IDEF1X)**, Gaithersburg, MD, December 1993.
- RAMAKRISHNAN, R. **Database management systems**. Second edition. WCB/McGraw-Hill, 2000.
- SILBERSCHATZ, A.; KORTH, H.; SUDARSHAN, S. **Sistema de Banco de Dados**. 3ª edição. São Paulo: Makron Books, 1999.
- TSICHRITZIS, D. & KLUG, A. (eds.). "The ANSI/X3/SPARC DBMS framework report of the study group on database management systems". **Information Systems** 3, pp. 173-191, 1978.
- W3C (World Wide Web Consortium). **W3C XML Schema**. Disponível em <http://www.w3.org/XML/Schema>, acessado em 2002.09.02.