Matemática relacional, teoria dos conjuntos e a resolução de problemas

Visão Geral:

Um banco de dados é um conjunto de dados relacionados de maneira lógica de forma a tornar seu acesso facilitado ao usuário, mantendo um alto nível de integridade e segurança dos mesmos. Quando nos referimos a dados, estamos nos referindo a fragmentos de informações, podendo estes, se analisados isolados, não fornecerem informações relevantes. Podemos tomar como exemplo um nome qualquer, “Alan”, fora de um contexto ele se torna um fragmento de informação irrelevante, pois não tem uma utilidade pratica, não sabemos onde, porque, nem como utiliza-lo ou sua procedência, entretanto, se juntarmos ele a outros fragmentos, como, “casa”, “foi”, “para”, e ordena-los de maneira concisa, poderemos obter uma informação completa, neste caso, “Alan foi para casa”.

Um sistema de gerenciamento de banco de dados, ou SGBD, é um software, ou programa de computador, que possui recursos para a manipulação dos fragmentos de informações e interagir com o usuário. Esses programas são ferramentas que utilizamos, os administradores de banco de dados, para organizar de maneira concisa, por meios de logica matemática, os diversos fragmentos de informações de uma forma que eles possam ser manipulados pelo usuário, da forma mais pratica possível.

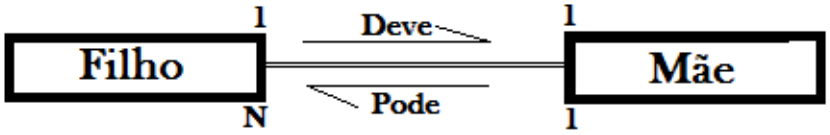
Modelagem de dados

O modelo de dados que demonstraremos nesse artigo se baseia no modelo relacional algébrico, partindo da premissa, de que os dados serão organizados em tabelas. Este conceito foi criado por Edgar Frank Codd, engenheiro da IBM, sendo publicado no artigo *“Relational Model of Data for Large Shared Data Banks”*. Nele temos conceitos relevantes a sua estrutura, dos quais, o tipo de relação, onde definimos qual entre os diversos tipos possíveis de relação, a entidade, que caracteriza o objeto em analise, e os atributos, que representam os dados ligados a entidade. **Exemplo:**

Louis é um garoto de sete anos, estudante, que é filho de Ana, uma analista de trinta e quatro anos. Ao analisarmos as informações, poderemos definir duas entidades, Louis e Ana, e seus atributos, como representados na tabela:

Louis	Ana
7 anos	34 anos
Estudante	Analista

Após definirmos seus atributos, poderemos analisar seu relacionamento, neste caso um familiar, uma relação de filho para mãe, se conceituarmos um pouco mais, e devemos, poderemos representar essa relação como, 1 para 1, no ponto de vista de Louis, e 1 para N, no de Ana, pois, um indivíduo, deve ter apenas uma única mãe biológica, mas uma mãe pode ter vários filhos.



O sistema de gerenciamento de banco de dados deve garantir uma forte hierarquia de acesso e manipulação dos dados, pois são estas que garantem a segurança dos dados. Podemos destacar três níveis de acesso principais, o do usuário, onde as informações podem ser inseridas e consultadas, o de controle, onde o administrador junto com o usuário, definirão o que e como será armazenado e acessado na camada superior, e o nível de armazenamento, onde os dados ordenados, ou informações, serão mantidos, este deverá ser consultado pela estrutura criada no nível de controle.

Modelando o nível de controle:

O administrador usará de uma série de conjuntos de estruturas logicas, que conceituaremos como modelagens, para organizar as ideias e dados do usuário, em uma estrutura real que possa trazer soluções ao problema que motivou a criação do banco de dados.
A primeira é a modelagem conceitual, ela surge das primeiras análises das ideias do usuário, ela será de extrema importância, pois ditará a viabilidade do projeto, a melhor forma de torna-lo real, o tamanho do projeto e principalmente, o que será armazenado. Quando concluímos o modelo conceitual, iremos revisar todo o projeto, mas agora, com uma visão mais matemática, buscando evitar a repetição de informações, informações imprecisas e qualquer estrutura que possa gerar dados que comprometam a integridade dos dados. Após esta modelagem revisória, iremos para a modelagem física, que basicamente, é torna real, por meio de um SGBD, o nosso banco de dados, transformando essas ideias em um sistema de gerenciamento de informações.

As operações:

Após criarmos conceitualmente o banco de dados, precisaremos torna-lo real, usando um SGBD, e para isso, precisaremos entender as operações que ele pode realizar, e como utilizaremos essas operações, ou funções. Para demonstrar o funcionamento de um SGBD, optamos pela escolha da ferramenta desenvolvida pela Microsoft, o Sql-Server 2012, e demonstraremos suas principais funções, com o desenvolvimento de uma aplicação simples, mas antes de começarmos, analisaremos as funções:

Create database nome_do_banco: Para criarmos um novo banco, precisaremos informar ao SGBD que criaremos um novo objeto, por meio da função “create”, em seguida, informar o tipo do objeto, no caso um novo banco de dados, a “linguagem”, como chamamos na computação uma serie já estabelecida de palavras que chamam

funções, SQL foi escrita usando termos em inglês, por isso utilizaremos a expressão “database”, por fim, definiremos um nome para o nosso banco.

Use nome_do_banco : Para utilizarmos nosso banco, precisamos informar o SGBD, então usaremos o comando “use”.

Create table nome_da_tabela : Nossas entidades, terão de ser organizadas em tabelas, sendo que nelas vamos expressar seus atributos, para criarmos uma nova entidade, passamos o parâmetro “create” e definimos o seu tipo, no caso “table”, para a nova entidade.

Atributos e tipos de dados:

Precisaremos definir que tipo de dados vamos guardar, para que o SGBD saiba como lidar com eles, existem diversos tipos de dados, para o nosso exemplo, utilizaremos três bem comuns, o para números inteiros, expressado como *“int”*, o para caracteres, quando sabemos exatamente quantos vamos utilizar, *“char(numero de caracteres)”*, e quando temos uma estimativa, *“varchar(numero máximo de caracteres)”*. Temos algumas funções que poderemos utilizar com os atributos, exemplo, se quisermos que um código se incremente sozinho, utilizaremos a função *“identity”*, ou a função *“not null”* que define os atributos que não podem ficar sem informações.

A estrutura de um atributo é: *Nome_atributo tipo_atributo incremento*.

Relacionamentos:

Para expressar um relacionamento entre duas entidades, precisaremos definir, qual atributo fará o elo de ligação entre elas, este atributo, nunca poderá ter valor nulo, nem valor repetido, este recebera o incremento *“primary key”*. Para efetuarmos a ligação, teremos que ligar nossa *“primary key”* a um atributo de outra tabela, que possua o mesmo tipo de dados de nossa chave, este atributo chamaremos de *“foreign key”*. Utilizaremos, para definir a ligação, a expressão:

constraint Nome_atributo foreign key (Nome_atributo) references Nome_tabela (Nome_Atributo)

Nome_atributo: Representa o atributo que receberá o valor da chave primaria.
Nome_tabela: Representa o nome da entidade onde se encontra nossa “primary key”.
Nome_Atributo: Representa a *“primary key”*.

Estrutura da entidade ou tabela:

Para que o SGBD possa compreender nossos comandos, precisaremos expressa-los em um padrão, ou melhor, em uma estrutura. Para qualquer tabela, seguira essas regras e estrutura:

Create Nome_tabela(
Nome_atributo tipo_atributo incremento,
Nome_atributo tipo_atributo incremento,
…
)

Todos os atributos serão definidos no espaço entre os parênteses, cada linha de definição de atributos deverá terminar com uma virgula, salvo a última linha.

Orientação a objetos:

Programação orientada a objetos, ou POO, é um paradigma de programação, que busca simular o mundo real no virtual, definindo tudo como objetos, para compreendermos POO, teremos que definir o que são classes, métodos e propriedades de objetos:

Classes: Representa um conjunto de objetos com característica parecidas, exemplo, a classe calculadora, nela teremos diversos tipos de calculadoras, que serão os objetos, mesmo que cada uma tenha características que as tornem únicas, ainda sim compartilharão diversas características, como sua função principal, a de calcular.

Métodos: Representa as operações que uma determinada classe, ou objeto, possa fazer, seguindo o exemplo anterior, os métodos da classe calculadora, poderiam ser subtrair, somar, dividir, ou qualquer outra função matemática.

Propriedades: As propriedades tem a mesma função dos atributos de dados, servem para identificar as características, seguindo o exemplo, nossa classe poderia receber como propriedade, a cor azul, logo todos os seus objetos, teriam de compartilhar desta característica.

A realidade nebulosa:

Usaremos uma situação hipotética para exemplificar os conceitos demonstrados no artigo. Analisaremos um problema envolvendo uma pequena escola, com base no texto que especificará o problema, vamos criar a realidade nebulosa do problema, ou seja, nossa primeira visão geral do problema, para que com ela possamos passar por todas as etapas de modelagem, até chegarmos a um simples aplicativo de cadastro e consulta.
Uma escola qualquer está enfrentando alguns problemas por não ter uma maneira pratica e informatizada de guardar os registros de seus alunos, tendo de recorrer a formulários de papel, para registros simples, de alunos e cursos, como nome, RA, ano cursado, RG, seu endereço e qual curso frequenta, para os alunos e, nome, duração e período, para os cursos. Usando os conhecimentos expressados no artigo, como poderemos resolver o problema da escola?

Vamos começar identificando as entidades do problema, no caso, alunos e cursos, e definirmos seus atributos, utilizaremos tabelas para expressar os dados:

Alunos	Curso
Nome	Ano
RA	Endereço

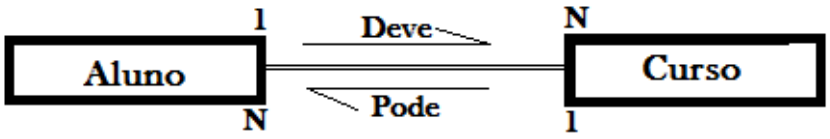
Cursos
Nome
Periodo
Duração

O segundo passo conceitual é avaliar os dados que possam fornecer informações imprecisas, podemos notar nesse caso, o atributo endereço, pois ele é muito generalista, dá margem a interpretação, tomaremos como medida fragmentar esse atributo, em outros mais precisos, rua, bairro, estado e cidade.
Para evitar erros, colocaremos um código para o curso, pois um nome pode ser escrito de diversas formas, mas uma sequência numérica não:

Aluno	Rua
RA	Bairro
Nome	Cidade
Ano	Estado
RG	CodCurso

Curso
Nome
Duração
Periodo
CodCurso

Como podemos notar na tabela, já foi definido o elo de ligação, *"foreign key"*, que pode ser representado pelo diagrama:



Um aluno deve estar cadastrado em um ou mais cursos, para ter a condição de aluno, um curso pode ter um ou mais alunos, sem perder a condição de curso.
Com nosso modelo conceitual pronto, poderemos passar a próxima etapa, a da criação em si de nosso conjunto de dados, conforme previamente demonstrado, usaremos do SQL Server 2012 para desenvolver nosso código:

```
create database DB_Escola
use DB_Escola

create table TB_Curso(
    Codcurso int identity primary key,
    Nome varchar(50),
    Duracao int,
    Periodo varchar(15)
)

create table TB_Aluno(
    RegistroAluno int primary key,
    Nome varchar(50) not null,
    Ano int,
    Rua varchar(50),
    Bairro varchar(50),
    Cidade varchar(50),
    Estado varchar(50),
    RG char(9),
    CodCurso int,
    constraint CodCurso foreign key (CodCurso) references TB_Curso (CodCurso)
)
```

Após o desenvolvimento do mesmo, nos resta apenas criar uma interface para o usuário gerenciar os dados.

Conclusão

Neste artigo demonstramos como realizar todas as etapas relacionadas a uma matriz de dados, partindo desde o momento em que surge o problema, como uma escola que precisa de uma maneira eficiente de gerenciar seus recursos, até a aplicação da solução, a implantação do sistema de gerenciamento. Desde o artigo publicado por um dos gênios da historia da IBM, até a publicação deste, muitas tecnologias e soluções surgiram, e indiretamente, as aplicações da área de informática tendem a dar uma maior prioridade ao gerenciamento de dados, por este motivo, não é só necessário conhecer esse processo, como tentar aprimorá-lo, nossa sociedade depende dessas tecnologias, mesmo não parecendo, se um modelador de dados entende errado a ideia de um cliente, ou não aplica as modelagens precisas, podemos ter muitos problemas, imagine se não conseguir realizar uma compra, pelo simples motivo de um erro no momento da modelagem do sistema de acesso do servidor do cartão? Embora muitos, não só da área de TI, não gostem de etapas mais ortodoxas, como as diversas modelagens representadas, elas representam um avanço desenvolvido ao longo dos anos, por diversas pessoas, representando um desenvolvimento conjunto de nossa sociedade.

Resumo
Com este artigos pretendemos demonstrar o funcionamento da teoria de modelagens de banco de dados, que descende de um ramo da álgebra relacional, explicando cada uma das diversas etapas, que recebem o nome de modelagens, e quando ou como devemos utiliza-las, para solucionar um problema qualquer. Essas teorias surgiram com o proposito de resolver diversos problemas ao longo do tempo, mas no geral, atendem ao mesmo proposito, como podemos guardar nossas informações? Como podemos garantir a disponibilidade? Será que apenas as pessoas com permissão vão acessa-las? Elas não vão ser corrompidas? Para responder essas duvidas, a tecnologia teve que avançar e encontrar padrões para garantir uma resposta positiva a esses questionamentos. Se perguntarmos a um profissional da área, que comumente recebe o nome de DBA, que podemos traduzir como administrador de banco de dados, ele nos explicará que um dos maiores desafios é garantir a disponibilidade, confidencialidade e integridade dos dados, sendo esses, os maiores objetivos de qualquer solução em relação as informações.