Artigo

MER e DER: Modelagem de Bancos de Dados

Veja neste artigo as definições de Modelo Entidade Relacionamento (MER) e Diagrama Entidade Relacionamento (DER), utilizados na modelagem de bancos de dados.

Quando se inicia o desenvolvimento de um novo sistema, ou mesmo de uma nova funcionalidade para um sistema existente, um dos primeiros passos a ser executado é o estudo e levantamento dos requisitos necessários para a construção do produto final. Durante essa análise, identifica-se as principais partes e objetos envolvidos, suas possíveis ações e responsabilidades, suas características e como elas interagem entre si.

A partir das informações obtidas, pode-se desenvolver um modelo conceitual que será utilizado para orientar o desenvolvimento propriamente dito, fornecendo informações sobre os aspectos relacionados ao domínio do projeto em questão.

Saiba mais sobre levantamento de requisitos na Engenharia de Software.

Modelo Entidade Relacionamento

O Modelo Entidade Relacionamento (também chamado Modelo ER, ou simplesmente MER), como o nome sugere, é um modelo conceitual utilizado na Engenharia de Software para descrever os objetos (entidades) envolvidos em um domínio de negócios, com suas características (atributos) e como elas se relacionam entre si (relacionamentos).

Em geral, este modelo representa de forma abstrata a estrutura que possuirá o banco de dados da aplicação. Obviamente, o banco de dados poderá conter várias outras entidades, tais como chaves e tabelas intermediárias, que podem só fazer sentido no contexto de bases de dados relacionais.

> **Observação**: Nem sempre criaremos modelos para um sistema completo, pois isso poderia resultar em

> finanças, recursos humanos, etc. Várias entidades estão presentes em mais de uma parte do sistema, mas não seria muito interessante, e provavelmente nem mesmo necessário, criar um único modelo para todo o sistema, por isso pode-se dividir a modelagem em várias partes menores.

Entidades

Os objetos ou partes envolvidas um domínio, também chamados de entidades, podem ser classificados como físicos ou lógicos, de acordo sua existência no mundo real. Entidades físicas: são aquelas realmente tangíveis, existentes e visíveis no mundo real, como um cliente (uma pessoa, uma empresa) ou um produto (um carro, um computador, uma roupa). Já as entidades lógicas são aquelas que existem geralmente em decorrência da interação entre ou com entidades físicas, que fazem sentido dentro de um certo domínio de negócios, mas que no mundo externo/real não são objetos físicos (que ocupam lugar no espaço). São exemplos disso uma venda ou uma classificação de um objeto (modelo, espécie, função de um usuário do sistema).

Série: Levantamento de Requisitos

As entidades são nomeadas com substantivos concretos ou abstratos que representem de forma clara sua função dentro do domínio. Exemplos práticos de entidades comuns em vários sistemas são Cliente, Produto, Venda, Turma, Função, entre outros.

Podemos classificar as entidades segundo o motivo de sua existência:

- **Entidades fortes**: são aquelas cuja existência independe de outras entidades, ou seja, por si só elas já possuem total sentido de existir. Em um sistema de vendas, a entidade produto, por exemplo, independe de quaisquer outras para existir.
- **Entidades fracas**: ao contrário das entidades fortes, as fracas são aquelas que dependem de outras entidades para existirem, pois individualmente elas não fazem

- **Entidades associativas**: esse tipo de entidade surge quando há a necessidade de associar uma entidade a um relacionamento existente. Na modelagem Entidade-Relacionamento não é possível que um relacionamento seja associado a uma entidade, então tornamos esse relacionamento uma entidade associativa, que a partir daí poderá se relacionar com outras entidades. Para melhor compreender esse conceito, tomemos como exemplo uma aplicação de vendas em que existem as entidades Produto e Venda, que se relacionam na forma muitos-para-muitos, uma vez que em uma venda pode haver vários produtos e um produto pode ser vendido várias vezes (no caso, unidades diferentes do mesmo produto). Em determinado momento, a empresa passou a entregar brindes para os clientes que comprassem um determinado produto. A entidade Brinde, então, está relacionada não apenas com a Venda, nem com o Produto, mas sim com o item da venda, ou seja, com o relacionamento entre as duas entidades citadas anteriormente. Como não podemos associar a entidade Brinde com um relacionamento, criamos então a entidade associativa "Item da Venda", que contém os atributos identificadores das entidades Venda e Produto, além de informações como quantidade e número de série, para casos específicos. A partir daí, podemos relacionar o Brinde com o Item da Venda, indicando que aquele prêmio foi dado ao cliente por comprar aquele produto especificamente.

Mais adiante veremos um exemplo prático onde poderemos observar a existência dessas entidades de forma mais clara.

Relacionamentos

Uma vez que as entidades são identificadas, deve-se então definir como se dá o relacionamento entre elas. De acordo com a quantidade de objetos envolvidos em cada lado do relacionamento, podemos classifica-los de três formas:

- **Relacionamento 1..1 (um para um)**: cada uma das duas entidades envolvidas

currículo na base, ao mesmo tempo em que cada currículo só pertence a um único usuário cadastrado.

- **Relacionamento 1..n ou 1..* (um para muitos)**: uma das entidades envolvidas pode referenciar várias unidades da outra, porém, do outro lado cada uma das várias unidades referenciadas só pode estar ligada uma unidade da outra entidade. Por exemplo, em um sistema de plano de saúde, um usuário pode ter vários dependentes, mas cada dependente só pode estar ligado a um usuário principal. Note que temos apenas duas entidades envolvidas: usuário e dependente. O que muda é a quantidade de unidades/exemplares envolvidas de cada lado.
- **Relacionamento n..n ou *..* (muitos para muitos)**: neste tipo de relacionamento cada entidade, de ambos os lados, podem referenciar múltiplas unidades da outra. Por exemplo, em um sistema de biblioteca, um título pode ser escrito por vários autores, ao mesmo tempo em que um autor pode escrever vários títulos. Assim, um objeto do tipo autor pode referenciar múltiplos objetos do tipo título, e vice versa.

Os relacionamentos em geral são nomeados com verbos ou expressões que representam a forma como as entidades interagem, ou a ação que uma exerce sobre a outra. Essa nomenclatura pode variar de acordo com a direção em que se lê o relacionamento. Por exemplo: um autor escreve vários livros, enquanto um livro é escrito por vários autores.

Atributos

Atributos são as características que descrevem cada entidade dentro do domínio. Por exemplo, um cliente possui nome, endereço e telefone. Durante a análise de requisitos, são identificados os atributos relevantes de cada entidade naquele contexto, de forma a manter o modelo o mais simples possível e consequentemente armazenar apenas as informações que serão úteis futuramente. Uma pessoa possui atributos pessoais como cor dos olhos, altura e peso, mas para um sistema que funcionará em um supermercado, por exemplo, estas informações dificilmente serão relevantes.

- **Descritivos**: representam característica intrínsecas de uma entidade, tais como nome ou cor.
- **Nominativos**: além de serem também descritivos, estes têm a função de definir e identificar um objeto. Nome, código, número são exemplos de atributos nominativos.
- Referenciais: representam a ligação de uma entidade com outra em um relacionamento. Por exemplo, uma venda possui o CPF do cliente, que a relaciona com a entidade cliente.

Quanto à sua estrutura, podemos ainda classificá-los como:

- **Simples**: um único atributo define uma característica da entidade. Exemplos: nome, peso.
- **Compostos**: para definir uma informação da entidade, são usados vários atributos. Por exemplo, o endereço pode ser composto por rua, número, bairro, etc.

Alguns atributos representam valores únicos que identificam a entidade dentro do domínio e não podem se repetir. Em um cadastro de clientes, por exemplo, esse atributo poderia ser o CPF. A estes chamamos de Chave Primária.

Série: MVC e Regras de negócio.

Já os atributos referenciais são chamados de Chave Estrangeira e geralmente estão ligados à chave primária da outra entidade. Estes termos são bastante comuns no contexto de bancos de dados. Mantendo o exemplo anterior, a entidade cliente tem como chave primária seu CPF, assim, a venda possui também um campo "CPF do cliente" que se relaciona com o campo CPF da entidade cliente.

Diagrama Entidade Relacionamento

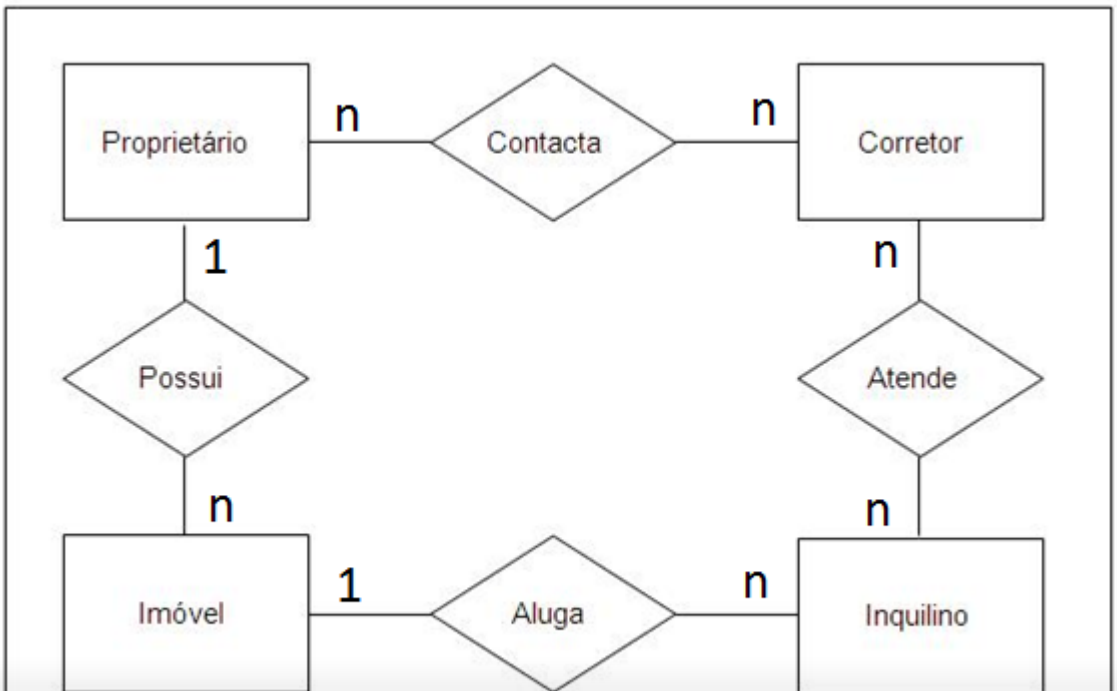
Enquanto o MER é um modelo conceitual, o Diagrama Entidade Relacionamento

sem uma forma de visualizar as informações, o modelo pode ficar abstrato demais para auxiliar no desenvolvimento do sistema. Dessa forma, quando se está modelando um domínio, o mais comum é já criar sua representação gráfica, seguindo algumas regras.

O diagrama facilita ainda a comunicação entre os integrantes da equipe, pois oferece uma linguagem comum utilizada tanto pelo analista, responsável por levantar os requisitos, e os desenvolvedores, responsáveis por implementar aquilo que foi modelado.

Em sua notação original, proposta por Peter Chen (idealizador do modelo e do diagrama), as entidades deveriam ser representadas por retângulos, seus atributos por elipses e os relacionamentos por losangos, ligados às entidades por linhas, contendo também sua cardinalidade (1..1, 1..n ou n..n). Porém, notações mais modernas abandonaram o uso de elipses para atributos e passaram a utilizar o formato mais utilizado na UML, em que os atributos já aparecem listados na própria entidade. Essa forma torna o diagrama mais limpo e fácil de ser lido.

Observe na **Figura 1** um exemplo simples de um diagrama para um sistema de imobiliárias.



No domínio representado pelo diagrama acima temos as seguintes entidades e relacionamentos:

- Proprietário contata Corretor (um proprietário pode contatar vários corretores e um corretor pode ser contatado por vários proprietários).
- Corretor atende Inquilino (um corretor pode atender vários inquilinos e um inquilino pode ser atendido por vários corretores).
- Inquilino aluga Imóvel (um inquilino aluga um imóvel e um imóvel pode ser alugado por vários inquilinos).
- Proprietário possui Imóvel (um proprietário possui vários imóveis e um imóvel pertence a apenas um proprietário).

Uma variante da **Figura 1** pode ser vista na **Figura 2**, onde a cardinalidade do relacionamento é exibida junto do losango.

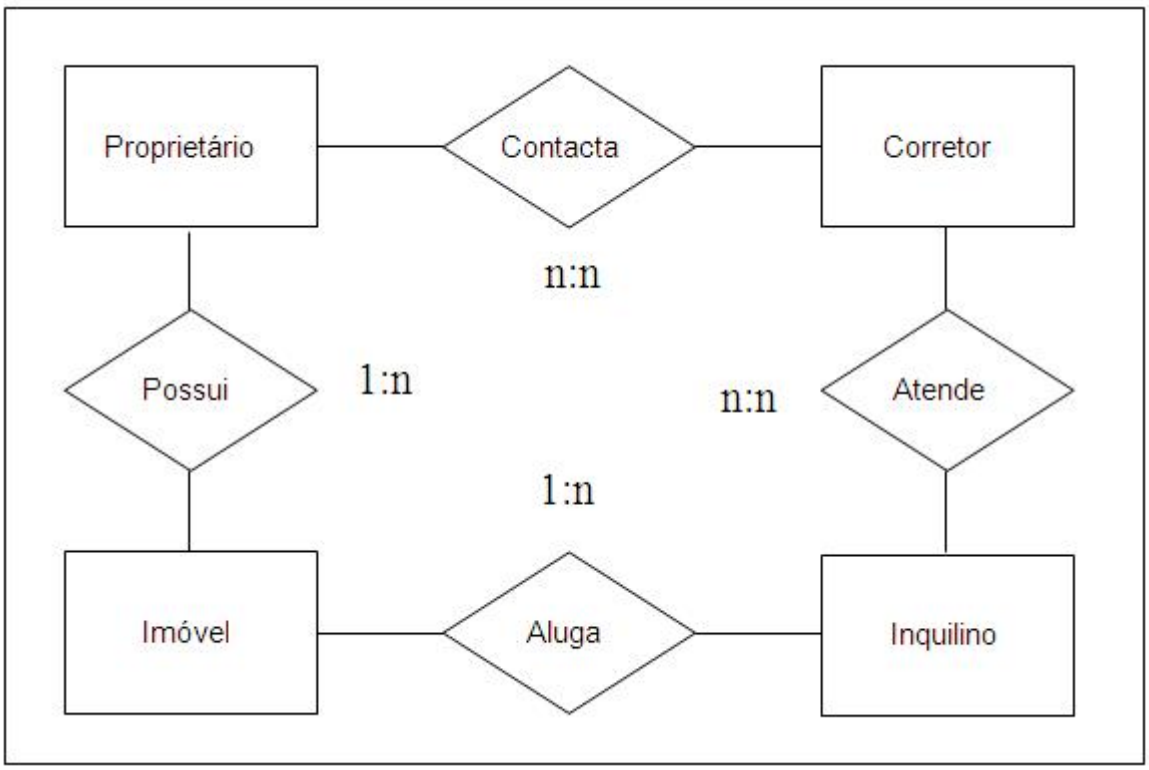


Figura 2. Diagrama de Entidade Relacionamento (variação)

modelo, em cada lado do relacionamento os números aparecem no formato (X,Y) ao invés de um único número como vemos nas figuras anteriores. A **Figura 3** ilustra um exemplo desse tipo.

Figura 3. Diagrama Entidade Relacionamento (variação 2)

Saiba mais sobre **Cardinalidade**

Neste diagrama, lemos os relacionamentos da seguinte forma:

- 1 ou 1 grupo possui 0 ou muitos produtos. Como de um lado temos "1 ou 1", isso equivale a apenas "1", pois não temos várias possibilidades. Já do lado do produto, indicamos que um grupo pode possuir nenhum produto, mas também pode possuir vários.
- 0 ou várias vendas contém 1 ou muitos produtos. Ou seja, um produto pode nunca ser vendido (0 vendas) como também pode ser vendido várias vezes (n vendas). Já uma venda deve conter 1 ou vários produtos, pois uma venda não pode estar vazia (0 produtos).

Os atributos, como já foi dito, podem aparecer no diagrama na forma de elipses ligadas às entidades. Essa foi a notação original proposta, mas como podemos ver na **Figura 4**, ela deixa o diagrama com muitos itens e pode atrapalhar um pouco a organização destes.

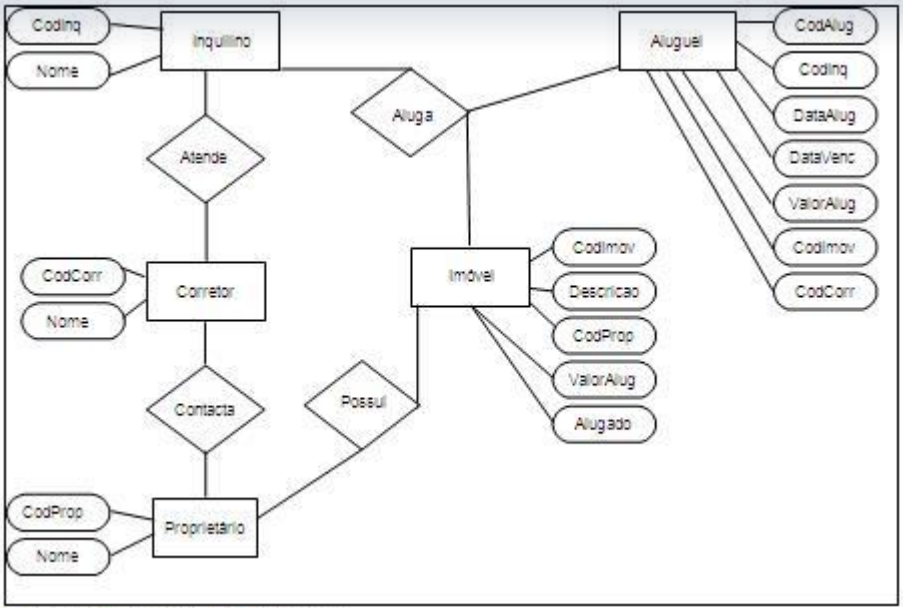


Figura 4. Atributos apresentados como elipses

Em uma notação mais atual, comumente utilizada na UML, os atributos aparecem listados dentro do próprio retângulo da entidade, enquanto o nome da entidade aparece no topo na forma de título. Na **Figura 5** temos um exemplo.

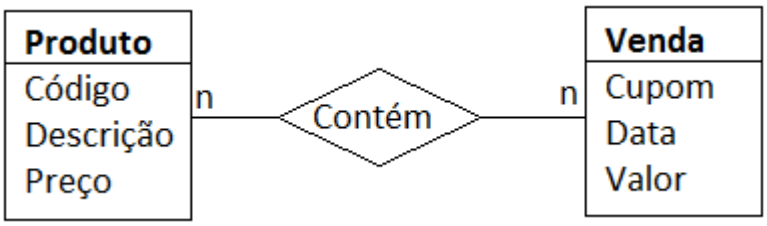


Figura 5. Diagrama com atributos nas entidades

Ferramentas CASE

Do inglês Computer-Aided Software Engineering, as chamadas ferramentas CASE são aquelas baseadas em computadores (softwares) utilizadas na Engenharia de Software para auxílio nas atividades desde análise de requisitos até, **modelagem de dados**.

No contexto desse artigo, as **ferramentas CASE** permitem a criação de diagramas de forma simples em um ambiente de fácil utilização e com recursos para incluir as principais regras de composição dos diagramas. Exemplos comuns desse tipo de ferramenta são: Star UML, Astah e ERwin Data Modeler. Na **Figura 6** vemos um exemplo de diagrama sendo construído no Astah.

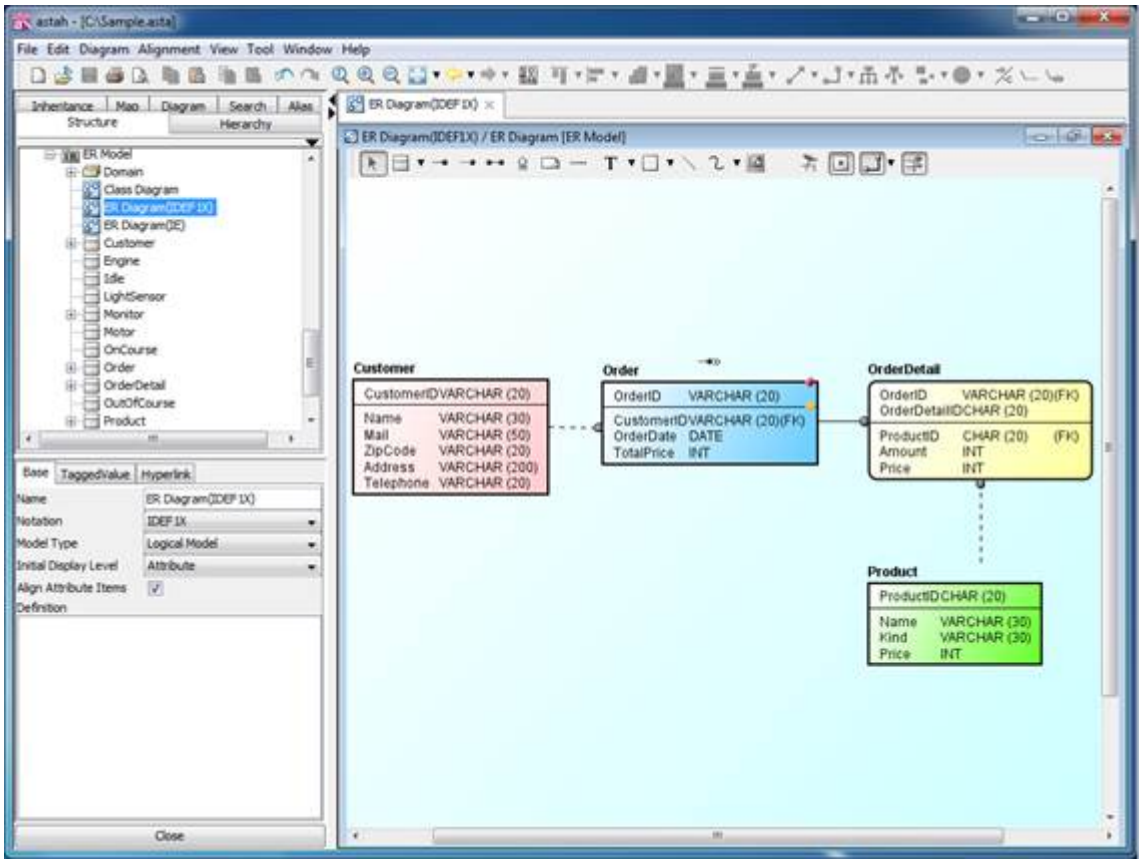


Figura 6. Diagrama no Astah Community

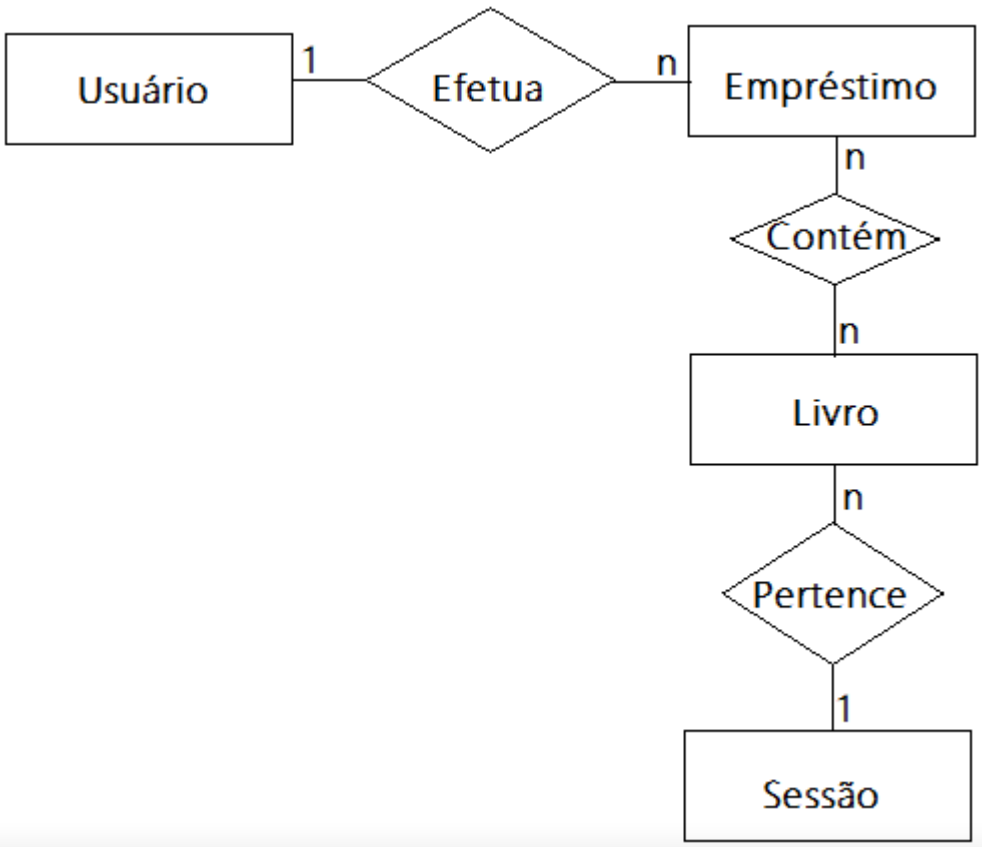
Além dessas ferramentas específicas, alguns IDEs (Integrated Development Environment ou Ambiente de Desenvolvimento Integrado) como o Visual Studio e ferramentas de gerenciamento de bancos de dados como SQL Server Management Studio possuem funcionalidades para criar diagramas facilmente e já gerar o código equivalente (SQL para criação das tabelas, chaves e relacionamentos, por exemplo).

Exemplo prático

Para fixar tudo que foi visto ao longo deste artigo, vamos agora desenvolver um pequeno exemplo prático em que modelaremos um sistema de bibliotecas, focando especificamente no empréstimo de livros.

Primeiramente precisamos identificar as entidades envolvidas nesse contexto. Sabemos que as entidades físicas existentes são o Usuário da biblioteca e o Livro que será emprestado. Além disso, consideraremos aqui que o livro pertence a uma Sessão, que ajuda na organização das obras do acervo. Em um sistema real pode haver outras informações sobre o livro, mas para esse exemplo a sessão é o bastante. Por fim, temos a entidade lógica Empréstimo, que tanto está relacionada com o usuário, quanto com o livro.

Assim já podemos esboçar nosso primeiro diagrama, simples, contendo as principais entidades e o relacionamento entre elas (**Figura 7**).



Neste primeiro diagrama podemos identificar alguns dos conceitos vistos:

- Entidades fortes: Usuário, Livro e Sessão;
- Entidades fracas: Empréstimo;
- Relacionamentos: um Usuário efetua vários Empréstimos, vários Empréstimos contêm vários Livros, vários Livros pertencem a uma Sessão.

Agora que visualizamos o domínio no diagrama, podemos adicionar os atributos e outras entidades que se façam necessárias. Assim, passamos à **Figura 8**,

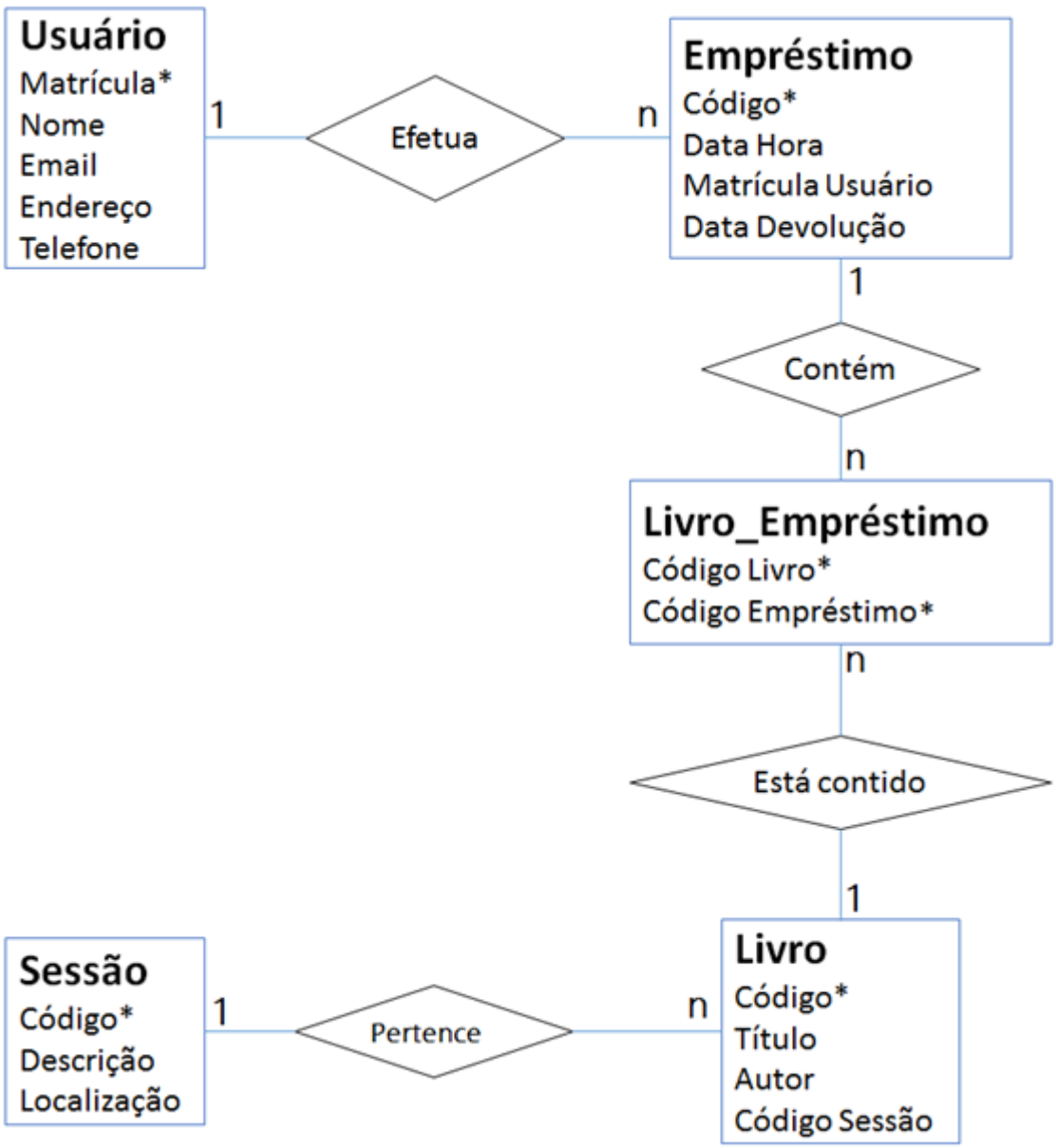


Figura 8. DER mais completo do sistema para bibliotecas

Neste ponto cabe fazer algumas observações importantes:

Especificamos os atributos de cada entidade e marcamos algumas elas com um asterisco, indicando que aquela é a chave primária da tabela, ou seja, um atributo único, que nunca poderá se repetir entre as entidades do mesmo tipo. Note que neste momento ainda não é necessário especificar o tipo de cada atributo (texto, número, data, etc.), isso só será necessário mais adiante, quando já estivermos planejando o banco de dados da aplicação.

Surgiu a entidade associativa Livro_Empréstimo, que representa os livros contidos em um empréstimo (considerando um empréstimo contém vários livros e um livro pode estar contido em vários empréstimos). Esta entidade é composta pelas chaves das duas entidades principais. Se fosse necessário, nesta entidade também poderíamos adicionar informações complementares como quantidade (não se aplica neste caso, mas caberia em um sistema de vendas, por exemplo) e observações sobre o item.

Na entidade associativa, o relacionamento n..n foi dividido em dois relacionamentos do tipo 1..n, agora lidos da seguinte forma: um empréstimo contém vários itens, mas um item só pode estar contido em um único empréstimo (restrito pelas chaves primárias); um livro pode estar contido em vários itens de empréstimo (ser emprestado várias vezes), mas cada item refere-se a um único livro.

O **Modelo Entidade Relacionamento** (e principalmente o diagrama) é uma importante ferramenta durante o desenvolvimento de sistemas, principalmente aqueles mais complexos e difíceis de visualizar sem uma análise mais aprofundada.

A correta modelagem auxilia no correto **desenvolvimento da base de dados** e evita que várias alterações sejam necessárias para corrigir erros de concepção provenientes de falhas durante a análise, ou ainda por problemas de comunicação entre os membros da equipe.

Banco de Dados Engenharia de Software Levantamento de Requisitos Modelagem UML

Voltar Anotar Marcar como concluído

Confira outros conteúdos:

Teste de Acessibilidade de Software

Boas Práticas em TDD

MATRICULE-SE

Por **Joel**
Em 2014

Tecnologias

Exercicios

Artigos

Quem Somos

Fale conosco

Plano para Instituição de ensino

Assinatura para empresas

Assine agora

Hospedagem web por Porta 80 Web Hosting.