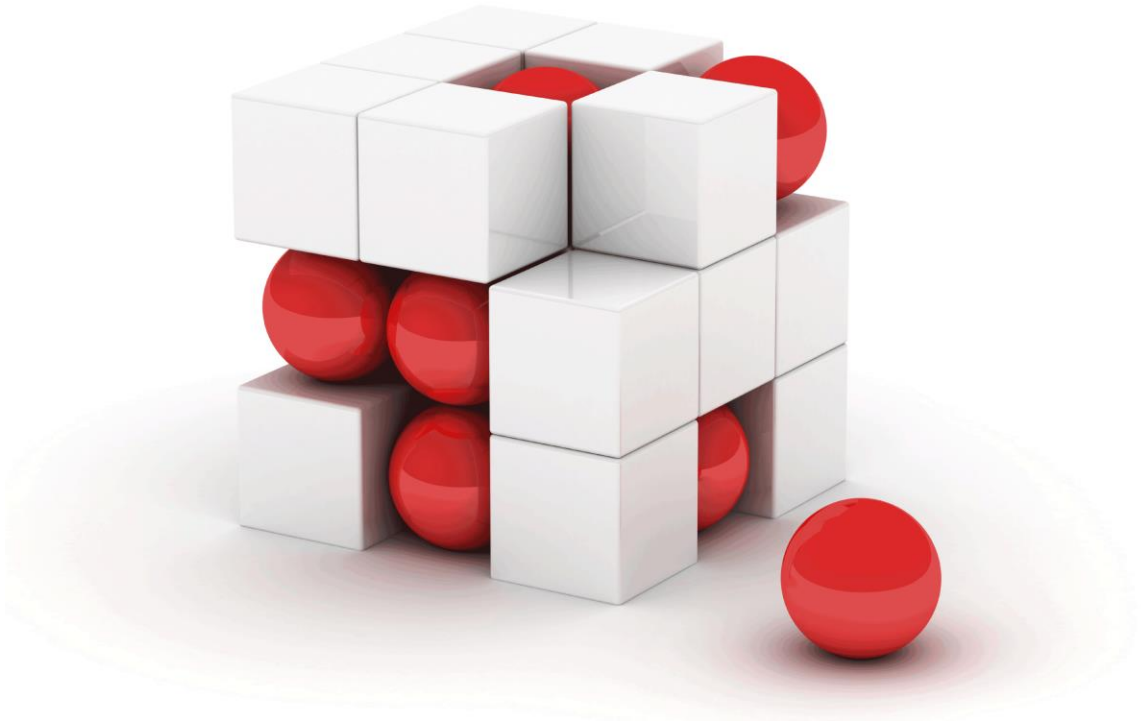


Modelagem de Banco de Dados Relacional

Professor Ari Oliveira

EDITORIAL

Este material foi organizado e desenvolvido pelo professor Ari Barreto de Oliveira, comouma extensão do material didático original do professor Robson Potier para a disciplina de Modelagem de Banco de Dados.

SUMÁRIO

1. A IMPORTÂNCIA DO ESTUDO DOS DADOS NO DESENVOLVIMENTO DE UM SISTEMA COMPUTACIONAL

Estudar e projetar Bancos de Dados (BD´s) de um sistema é hoje tão importante, ou talvez mais, quanto estudar a funcionalidade dele. Para entendermos melhor esta afirmação, vejamos o seguinte exemplo:

> *Um banco descobre que seu programa de controle de aplicações não é bom. Por causa de rotinas mal escritas ele demora o dobro do tempo necessário para fazer a maioria de suas operações. A equipe de desenvolvimento do Banco resolve então construir um novo programa. Enquanto o novo programa não está pronto o velho vai sendo usado. Quando o novo programa ficar pronto e testado, ele é implantado e pronto.*
>
> *Agora imagine que, no mesmo banco, descobriram que havia redundância de informação. Toda vez que se registrava uma aplicação de um cliente o nome dele era gravado junto com as informações da aplicação. Bem, o nome do cliente é um só. E, assim sendo, deve estar registrado em um só local, e não em vários. Para ter noção do problema imagine que descubram que o nome de um cliente está errado. Vamos ter que consertar em todos os locais nos quais ele aparece. Então a equipe de desenvolvimento do banco resolve remodelar o banco de dados. Até aí tudo bem. Mas como implantar este novo modelo se todas as informações estão armazenadas no modelo velho? E vale lembrar que, enquanto a equipe pensa o que fazer, o problema está aumentando. Pois novas informações não param de ser registradas. É resolvido então que o banco tem que parar de trabalhar por uns dias até que todas as informações antigas sejam convertidas para o novo formato. Vocês conseguem imaginar isto?*

2. O QUE É UM BANCO DE DADOS

É um agrupamento de informações interligadas que podem conter diversos conjuntos de dados ou somente um. Por exemplo: sua agenda pessoal é um banco de dados. Nela você tem um grupo de informações interligadas: Nome, Endereço, Telefone e Data de Aniversário, que tem uma relação forte pois identificam uma pessoa conhecida. Nesse caso, o conjunto de dados é um só. Já no caso de uma ficha de empréstimos de livros de uma biblioteca, que também é um banco de dados, geralmente temos as seguintes informações: código, título, usuários que emprestaram, as datas de empréstimo e as de devolução. Abaixo um exemplo da ficha.

BIBLIOTECA DA ESCOLA FICHA DE LIVRO		
Código: 67.986 CRÔNICA DE UMA MORTE ANUNCIADA		
Usuário	**Empréstimo**	**Devolução**
JULIANA	08/09/2012	18/09/2012
ROBSON	20/10/2013	25/10/2013
CLÁUDIO	30/05/2014	

Percebam que código e título pertencem ao conjunto de informações de livros (estão fortemente ligadas) mas usuários, datas de empréstimo e devolução pertencem a outro conjunto: empréstimos ou movimentos de livros. Existe uma ligação entre elas, porém elas pertencem a conjuntos distintos.

Um dos problemas cruciais em modelagem de banco de dados é justamente identificar quais são os conjuntos de informação existentes em um sistema.

3. OBJETIVOS E REQUISITOS DE UM BANCO DE DADOS AUTOMATIZADO

3.1 EVITAR REDUNDÂNCIAS

Nos sistemas manuais a redundância de dados é normal e necessária. Em algumas bibliotecas, por exemplo, cada vez que você aluga um livro o título dele é anotado em sua ficha de usuário. Isto facilita uma posterior pesquisa. Em um sistema automatizado, isso seria um desastre, pois além de ocupar mais espaço em disco do que o necessário, uma posterior mudança no título do livro levaria o sistema a percorrer milhares de registros. Nos sistemas automatizados, nós sempre perseguiremos o mínimo de redundância possível.

Imagine outro caso, de alguém que vai ao posto de saúde todos os dias, e todas as vezes tem que fazer uma nova ficha de cadastro contendo: seu nome, endereço, telefone, documentos, etc, antes de ser atendido. Isso facilita para o médico, para que ele rapidamente veja seus dados. Porém, isto vai fazer com que o fichário fique incontrolavelmente grande!

3.2 FACILITAR ATUALIZAÇÕES

Como vocês viram no tópico anterior, um banco de dados mal projetado pode dificultar a atualização das informações. Uma das principais vantagens dos computadores é justamente permitir que se atualize (edite) dados com rapidez e eficiência. Se você cria um BD que dificulta a atualização, então este BD é um problema!

Imagine só que um paciente do hospital mudou o seu número de telefone, por exemplo. Neste caso seria necessário encontrar todas a fichas de atendimento e alterá-las uma a uma! Seria necessário um desprendimento muito grande!

3.3 EVITAR INCONSISTÊNCIA DE DADOS

Imagine que você crie uma estrutura de BD que permita que seja alocado um professor para uma turma cujo curso não faz parte do elenco de cursos que o professor pode lecionar. Isso é uma inconsistência de dados. E você tem como evitar isto em seu projeto.

4. NÍVEIS DE ABSTRAÇÃO

No desenvolvimento de um sistema de banco de dados temos que levar em conta que o nosso trabalho vai basear-se em um modelo real. Ou seja, que existe no mundo real. E na maioria dos casos não está nem parcialmente automatizado. Então, temos que ter condições de visualizar, compreender e interpretar o modelo desde o real (o que existe) até o modelo operacional que será implementado.

4.1 MUNDO "REAL"

No nível do mundo "Real" nós não temos um modelo formal de informações. Elas estão dispostas sem limitações. O projetista de banco de dados tem que definir o que interessa do mundo real para o seu projeto. Os objetos do mundo real são os seres, os fatos, as coisas e os organismos sociais.

4.2 MODELO DESCRITIVO

Neste modelo você já filtrou o que interessa no mundo real e já estabelece alguns limites para a organização da informação utilizando linguagens não formais. Perceba que, neste nível, você começa a ter algo palpável, pois, neste modelo, já serão colocadas impressões a respeito de como os dados irão organizar-se. No nível de mundo real, você irá observar, entrevistar, pesquisar para poder ter subsídios para o modelo descritivo. Resumindo: o mundo real não é modelado, ele existe e pronto. Já o modelo descritivo é produto do seu trabalho. Por ser um nível onde não usamos ainda linguagens formais, a escolha da linguagem a ser utilizada é bem subjetiva. Alguns podem querer usar o próprio português, outros, podem querer usar alguma linguagem gráfica. Este é um nível de idéias e pensamentos de como as informações irão organizar-se.

4.3 MODELO CONCEITUAL

Nosso primeiro nível formal: Aqui definiremos estruturas de informação que servirão de base para o nosso modelo operacional. Nesse nível, são identificados os conjuntos de informação e as ligações existentes entre eles. Utilizaremos, nesta fase, o Modelo de Entidade e Relacionamento (MER) e sua linguagem gráfica (Diagrama de Entidade e Relacionamento).

4.4 MODELO LÓGICO

É uma das última etapa de modelagem, quando fazemos uma decomposição ou desmembramento do modelo Conceitual. Este modelo, apesar de completo, é independente de SGDB, ou seja, poderá ser criado em qualquer sistema de gerenciamento de banco de dados realacionais disponível no mercado.

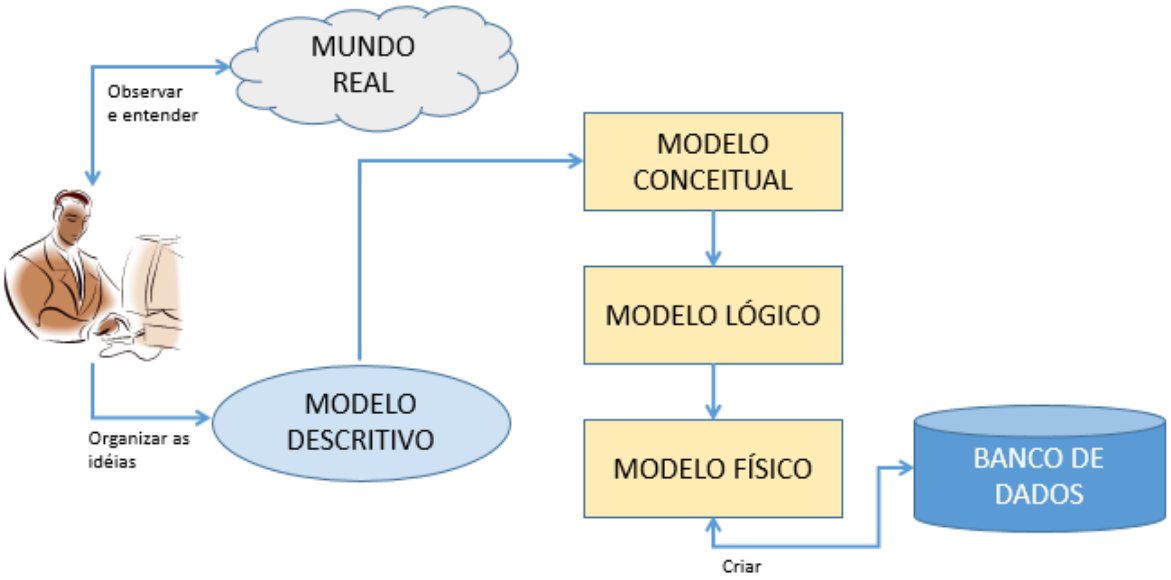
4.5 MODELO FÍSICO

É o modelo de banco de dados já focado para a utilização em um Sistema Gerenciador de Banco de Dados (SGBD), tais como: Access, Oracle, SQL Server, MySQL, Firebird, PostreSQL, DB2, Dbase, Paradox, Lotus Approach. Cada um destes SGBD´s tem sua maneira de implementar o seu modelo operacional. Muitos deles utilizam o SQL como linguagem para manipulação do banco de dados, o que faz com que a forma de operá-los seja muito parecida, porém com pequenas diferenças entre eles.

5. PROJETO DESCENDENTE (TOP-DOWN)

No desenvolvimento do nosso projeto de banco de dados, utilizaremos uma abordagem descendente. Começaremos no mundo real e terminaremos no modelo operacional. No caso de projetos de banco de dados, não temos muito o que decidir a respeito de abordagens, pois é quase uma imposição o projeto descendente.

Na passagem do mundo real para o modelo descritivo e deste para o conceitual, não temos ferramentas que nos ajudem na transição. Porém, na transição do modelo conceitual para o operacional, utilizaremos um processo chamado decomposição. A inexistência de ferramentas de transição, nos primeiros níveis, é conseqüência da informalidade dos dois primeiros níveis. Sem modelos formais é difícil estabelecer formalidades de passagem.



6. MODELO DESCRITIVO

Como já comentamos, a construção do modelo descritivo a partir do mundo real dá-se através de uma filtragem. Por meio de observações, entrevistas e pesquisas o programador irá definir o que lhe interessa no mundo real. Para exemplificar, vamos ver um modelo descritivo do sistema da coordenação pedagógica de uma escola.

Através de entrevistas e pesquisas nos documentos e relatórios utilizados, você chegou as seguintes conclusões:

É registrado, para cada professor, um número de matrícula, o nome dele, sua data de admissão e os cursos que ele pode ministrar;
Cada curso pode ter vários professores habilitados a lecioná-lo e é registrado para cada um deles um código, o nome dele e sua carga horária;
As turmas tem um código e é registrada a data de início e término, horário, professor, sala e o curso da turma;
Cada turma tem apenas um professor, porém este pode ser substituído por outro;
Cada turma está associada a uma sala, mas ela pode mudar de sala;
Cada turma faz um só curso;

Estas afirmações são o seu modelo descritivo.

7. MODELO CONCEITUAL

7.1 DEFININDO OS CONJUNTOS DE INFORMAÇÃO

O primeiro passo na construção de um modelo conceitual é o reconhecimento dos conjuntos de informação. Uma boa dica para encontrar os conjuntos está no próprio modelo descritivo. No exemplo acima, perceba que estão descritas ligações entre seres, coisas, fatos e organismos sociais. Identificando estes elementos você estará isolando os conjuntos de informação que neste caso são: professores, cursos e turmas.

Outra maneira de identificar os conjuntos é listar as informações do sistema e descobrir quem elas identificam.

Informação	**É sobre**
Matrícula	Professores
Carga Horária	Cursos
Data de Término	Turmas

7.2 O MODELO DE ENTIDADE-RELACIONAMENTO (MER)

O MER é o modelo escolhido para representar o modelo conceitual do nosso banco de dados. Ele utiliza-se de uma linguagem gráfica e é o mais aceito pelos projetistas de bancos de dados. Esta linguagem é chamada de Diagrama de Entidades e Relacionamentos (DER).

7.2.1 Entidades

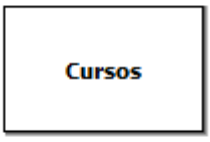
Os conjuntos de informação darão origem as entidades do DER. Cada entidade irá representar todos os elementos de um conjunto de informação. Elas serão modeladas com retângulos e com o nome do conjunto escrito dentro dele no plural porque a entidade representa todos os elementos daquele conjunto. Por exemplo: a entidade Professores não representa somente o Professor Ari e sim todos os professores da escola.

Existem algumas regras para determinar o que é ou não uma entidade. Veja a seguinte lista:

1. *Uma entidade é algo do mundo real que possui uma **existência independente**.*
2. *Pode ser um objeto com uma **existência física** - uma pessoa, carro ou empregado - ou pode ser um objeto com **existência conceitual** - uma companhia, um trabalho ou um curso universitário*
3. *A entidade tem que ter **mais de uma característica (atributo)***
4. *Algo que represente uma ação ou verbo **não** pode ser uma entidade!*

Ao estudar os conjuntos de informação presentes neste sistema poderemos encontrar:

- **Professores**: 👍 é uma entidade, pois possui existência independente e várias características.
- **Cursos**: 👍 é uma entidade, pois possui existência independente e várias características.
- **Turmas**: 👎 não é uma entidade! Apesar de ter várias características, mas turma é algo que existe apenas quando juntamos professores e cursos. Não possui existência independente!
- **Salas**: 👎 não é uma entidade! Apesar de ter existência independente, mas na descrição do sistema só foi dada uma característica sobre a sala: o nome da sala. Neste caso, ela é apenas uma informação a mais sobre a turma.



Vamos tentar explicar com outro exemplo. Um hospital precisa de um sistema para automatizar o atendimento aos seus pacientes. É necessário um cadastro de paciente, com nome, endereço, telefone, e tipo sanguíneo. Também é necessário um cadastro de médico, com nome, especialidade e telefone. Por último, também precisamos de um cadastro de atendimentos, registrando qual foi o paciente, quem foi o médico que atendeu, a data e o diagnóstico. Ao analisar as entidades, descobriremos que:

- **Pacientes**: 👍 é uma entidade, pois possui existência independente e várias características.
- **Médicos**: 👍 é uma entidade, pois possui existência independente e várias características.
- **Atendimentos**: 👎 não é uma entidade! Apesar de ter várias características, atendimento é uma ação, um acontecimento, ou seja, algo que existe apenas quando juntamos médicos e pacientes, e neste caso, não possui existência independente!

7.2.2 Atributos e Conjuntos de Valores

Os atributos são as informações de uma determinada entidade. A entidade professores, ou seja todos os professores, podem ter matrícula, nome e data de admissão. Cursos podem ter código, nome e carga horária.

Sobre os atributos, iremos colocar apenas uma regra: não podem existir atributos que façam referência a outra entidade. Se existirem, não criaremos os mesmos neste momento. Veja o exemplo de professores, onde deveríamos ter um atributo para armazenar a informação de que **cursos** ele ministra. Mas "**cursos**" é o nome de outra entidade existente. Neste caso não criaremos este atributo, e resolveremos isto mais tarde. Veja abaixo como ficarão as entidades e seus atributos.

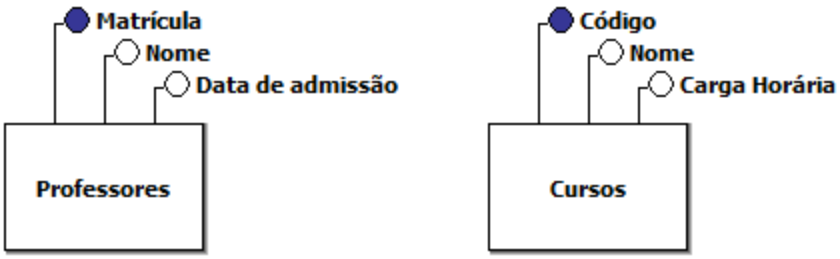
Percebam que o atributo representa todos os possíveis valores que aquela informação pode assumir. O atributo nome, por exemplo, representa todos os nomes de professores da escola.

Como a entidade representa todos os elementos de um conjunto de informações, futuramente poderemos armazenar as informações neste banco de dados dando valores para cada um dos atributos. Cada conjunto de valores é uma instância desta entidade. Veja abaixo o exemplo de 3 instâncias, ou seja, 3 registros da entidade **professor**:

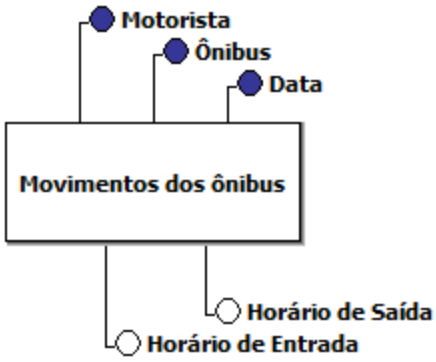
Matrícula	Nome	Data de Admissão
1230	NICHOLAS SERGEY	01/01/1993
1245	ARI OLIVEIRA	02/09/2001
1256	ANTÔNIO MARCOS	29/03/2000

7.2.3 Tipos de Atributos

Em cada entidade, além dos atributos simples, que representam informações básicas, devemos ter um atributo (ou uma junção de atributos) que tenha um valor único para cada instância. Ou seja, este valor não poderá se repetir de uma instância (registro) para outra, e tampouco poderá ficar em branco. Damos a este atributo o nome de **atributo determinante**. Ele recebe este nome porque através dele nós determinamos os outros atributos. No caso da entidade professores, o atributo determinante é Matrícula, pois cada professor tem seu número de matrícula que é exclusivo dele. No caso do curso é o código. Cada curso da escola tem seu código e ele não será usado para nenhum outro. Para identificar quais atributos são determinantes, usaremos uma bolinha pintada.



Em alguns casos, para termos um valor único que identifique somente uma instância da entidade, teremos que concatenar dois ou mais atributos. No caso de uma empresa de ônibus que queira saber qual motorista estava responsável pelo ônibus em determinado dia. Sabendo que cada ônibus somente pode ser usado por um motorista a cada dia, abaixo teríamos um modelo para armazenar essas informações:

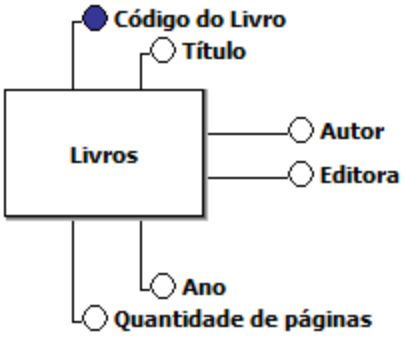


Neste caso, para obtermos um valor que seja exclusivo para cada instância, nós juntamos três atributos. Pela tabela abaixo, perceba que essa foi uma boa saída:

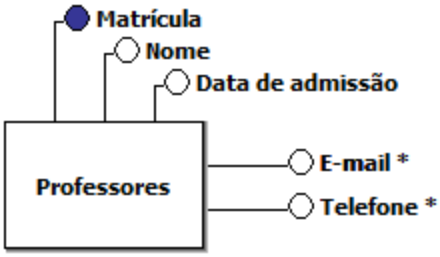
Motorista	Ônibus	Data	Saída	Entrada
0989	0010/92	05/06/97	08:00	16:00
0989	0010/92	06/06/97	07:30	15:00
0712	0032/93	05/06/97	07:00	16:00
0809	0032/93	06/06/97	07:00	16:00

Com este modelo apenas **não será possível:** registrar duas vezes o mesmo motorista conduzindo o mesmo ônibus na mesma data. Para outras possibilidades, será possível o cadastro, como por exemplo, o motorista que conduz dois diferentes ônibus, ou um ônibus que é conduzido por diferentes motoristas durante o dia, etc.

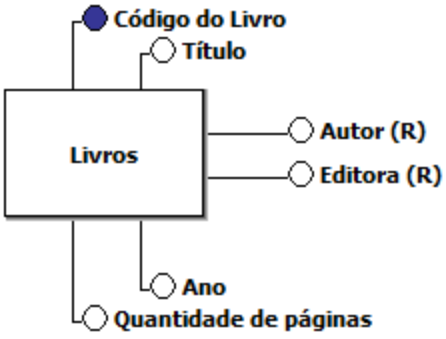
Existe a possibilidade de que com os atributos naturais de uma entidade você não encontre um que possa ser determinante. Neste caso o projetista tem que criar um **atributo artificial** que será determinante. Geralmente será um código, como por exemplo **Código do Cliente**, ou **Código do Curso**. Sempre que tiver dificuldades de encontrar um determinante, crie um determinante artificial como este.



Um atributo é dito **multivalorado** quando ele pode conter diversos valores. Imagine que nós resolvamos armazenar, para cada professor, todos os números de telefone que possua. Muitos têm o telefone residencial, telefone celular e telefone do escritório (ou de outro trabalho), mas cada atributo simples somente pode armazenar um valor. Para poder armazenar vários valores em um atributo este tem que ser do tipo **multivalorado**. Para indicar que um atributo é multivalorado usamos um asterisco (*).

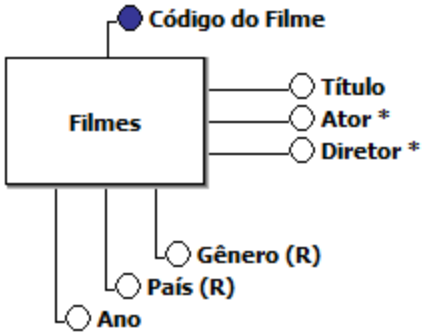


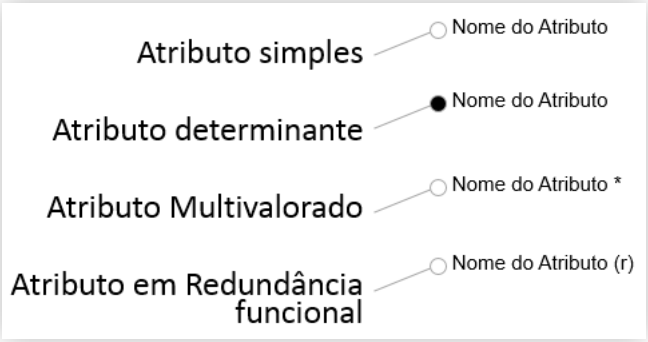
O último tipo de atributo é o atributo em **redundância funcional**. Isto acontece quando o valor do atributo pode se repetir excessivamente em um banco de dados. Por exemplo, quando cadastramos um livro temos que informar qual a categoria deste livro. Percebemos então que MUITOS livros terão como categoria “romance” ou “infantil” ou “drama”. Na verdade, serão apenas poucos tipos de categoria que se repetirão inúmeras vezes. Esta repetição, como já estudamos, é prejudicial quando precisamos atualizar o nome de alguma categoria, além de gastar mais espaço em disco. Por agora, podemos identificar as categorias colocando a expressão (R) ao fim do nome do atributo.



Observe que alguns dos atributos podem até se repetir (título do livro), mas caso não seja uma repetição excessiva, eles não cairão em redundância funcional. Atributos de valores numéricos e de datas nunca entram em **redundância funcional** (Ano, quantidade de páginas)

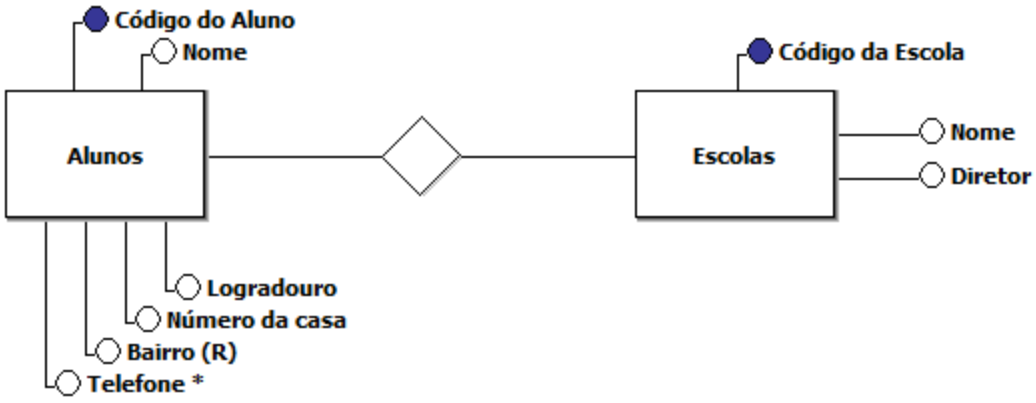
Se em algum local houver **redundância funcional e multivaloração** ao mesmo tempo, marcamos apenas **multivaloração.** Veja o exemplo avaixo de Ator. Ator cai em redundância funcional e em multivaloração. Neste caso marcamos apenas multivalocação. É o mesmo caso do diretor.



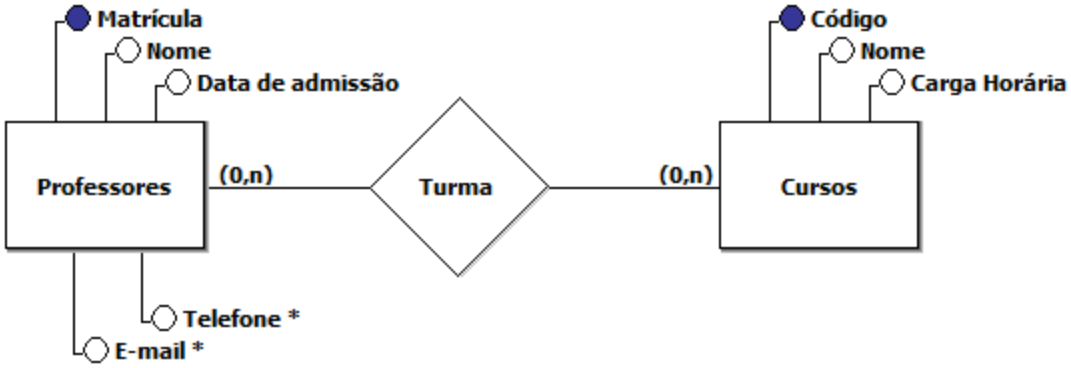


7.2.4 Relacionamentos

Em um sistema, as entidades não ficam isoladas sem nenhuma ligação com outros conjuntos de informação. A não ser que o sistema só tenha uma entidade. As entidades em um sistema estão ligadas através de relacionamentos. Um professor está habilitado a ministrar cursos. No DER nós vamos mostrar estes relacionamentos. Um relacionamento é representado por um losango que é colocado entre as duas entidades ou mais que ele liga. Dentro do losango pode ser escrito um nome para o relacionamento, mas isso não é tão importante. Muitos projetistas colocam as iniciais das entidades ligadas. Veja o exemplo abaixo.



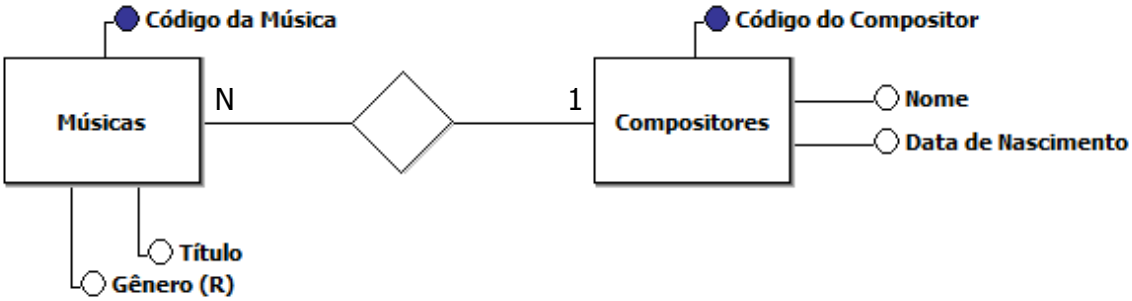
As vezes, a ligação entre entidades revela ações ou acontecimentos, e neste caso deveremos obrigatoriamente escrever o nome do relacionamento.



7.2.4.1 Cardinalidade de Relacionamentos

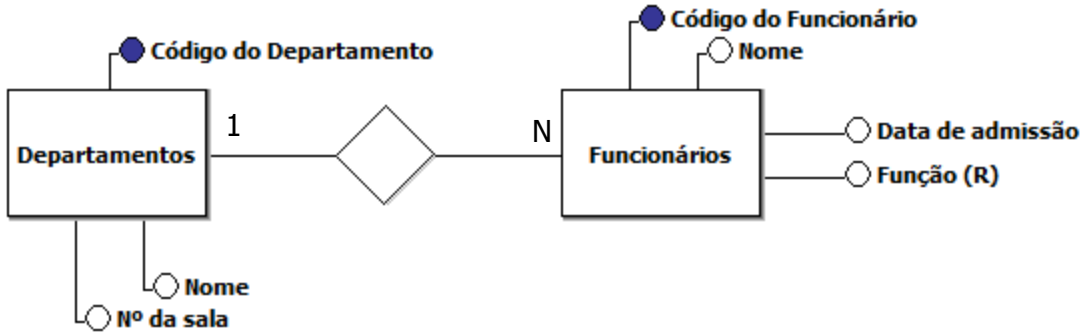
Não basta dizer que existe um relacionamento entre determinadas entidades. É preciso dizer também como os elementos de uma entidade se ligam aos elementos de outra. Por exemplo, um elemento do conjunto professores pode ligar-se a quantos de cursos? E um de cursos pode relacionar-se a quantos de professores? A resposta destas perguntas irá resultar na classe do relacionamento.

As classes de relacionamento principais são a 1:N e a N:N. Traduzindo: um para vários (ou vários para um) e vários para vários. Todas as outras classes são simplificações destas duas. Na 1:N temos que cada elemento de uma entidade X liga-se a vários de outra entidade Y relacionada, e que cada elemento da entidade Y relacionada, liga-se a somente um da entidade X. Tomo como exemplo um sistema de um arquivo de músicas e compositores. Cada música só tem **1** compositor, e cada compositor pode ser compositor de **N** (muitas) músicas.

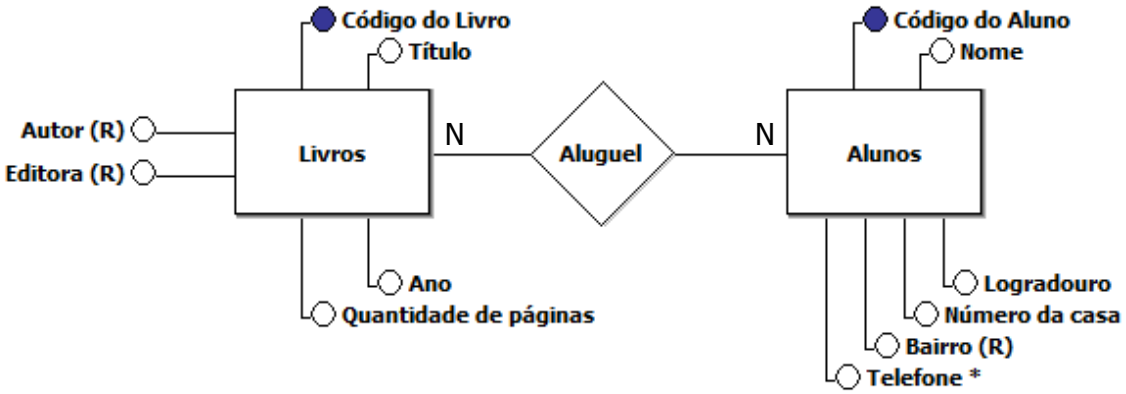


Observe atentamente como está disposta a cardinalidade no modelo acima. Para se descobrir qual a cardinalidade de um relacionamento, basta fazer a seguinte pergunta do "cada":

"CADA música se relaciona com quantos COMPOSITORES?" E a resposta, anotar "do lado de lá", em Compositores. Veja o outro exemplo abaixo. **Cada** departamento pode ter muitos (**N**) funcionários, mas cada funcionário só trabalha em **1** departamento.



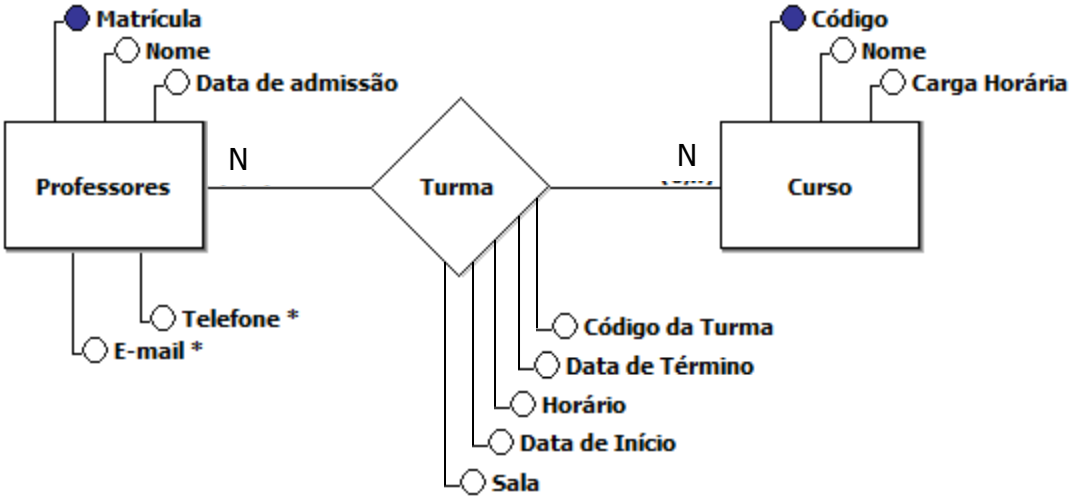
Já no relacionamento N:N, cada elemento de uma entidade X liga-se a vários elementos da entidade Y, e cada elemento de Y liga-se a vários de X. No exemplo dado abaixo, temos um relacionamento N:N. Cada livro pode ser alugado por muitos (**N**) alunos, e cada aluno pode alugar muitos (**N)** livros.



Alguns projetistas gostam de criar simplificações e/ou limitações das classes, tais como: relacionamentos 1:3, 2:5, mas no nosso entender isso gera problemas. Ao definir, por exemplo, que em uma locadora cada usuário tem no máximo 3 dependentes, mesmo que esta seja a política atual, você estará limitando o seu sistema.

7.2.4.2 Atributos de Relacionamentos

Capítulos atrás vimos o modelo descritivo da escola, e vimos que uma turma possui suas próprias características, lembra? Uma turma possui: código, data de início e término, horário e sala! Neste caso, estas informações nem entraram nos professores, nem no curso. Assim, podemos inserir todos estes atributos diretamente no relacionamento. Veja o exemplo a seguir.

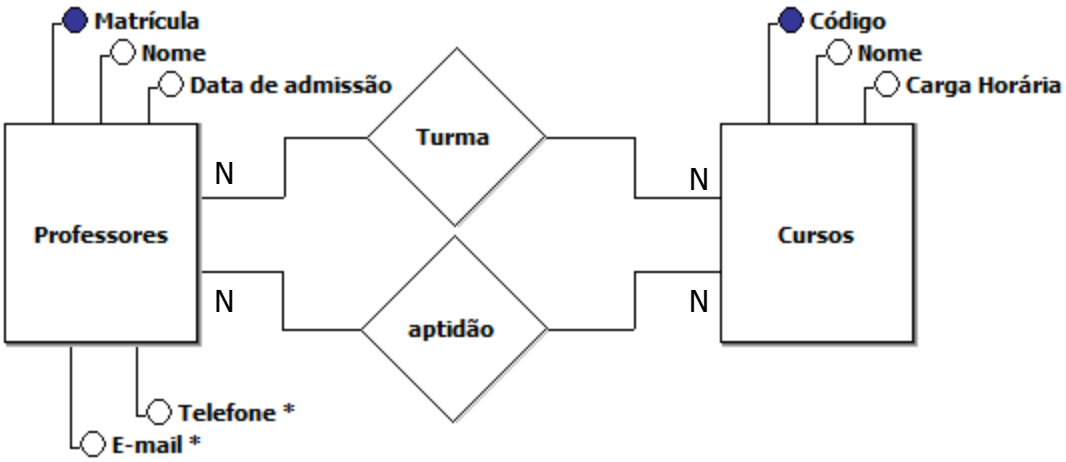


7.2.4.3 Quantidade de relacionamentos

Considere a possibilidade de que eventualmente podem surgir mais de um relacionamento entre duas entidades. Sim, isto é possível! Para identificar isto, basta verificar o que este relacionamento representa entre as entidades. Se forem coisas distintas, poderão ser criados mais de um relacionamento.

No caso da escola, vimos que um professor se relaciona com um curso formando uma turma. Mas também é necessário informar que um professor se relaciona com o curso para

formar os cursos que o professor ministra. Neste caso, teremos os seguintes relacionamentos:

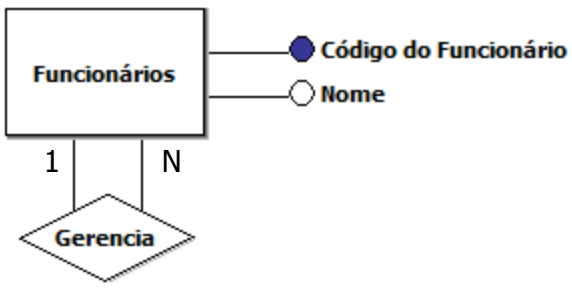


O primeiro relacionamento informa que cada professor pode entrar em turmas de **muitos** cursos, e cada curso pode ser dado por **muitos** professores (em diferentes turmas). Se é complicado pensar assim agora, basta saber que todo relacionamento que possui um nome que representa uma ação ou acontecimento (turma, aluguel, movimentação, etc), geralmente receberá N todas as pontas, ou seja, N-N!

O segundo relacionamento, que chamamos de aptidão, ou professores do cursos, ou mesmo poderíamos deixar sem nome, pois representa uma hierarquia. É apenas uma listagem de professores e os cursos que eles ministram. Neste caso, cada professor está apto para ensinar **muitos** cursos, e cada curso pode ser ministrado por **muitos** professores.

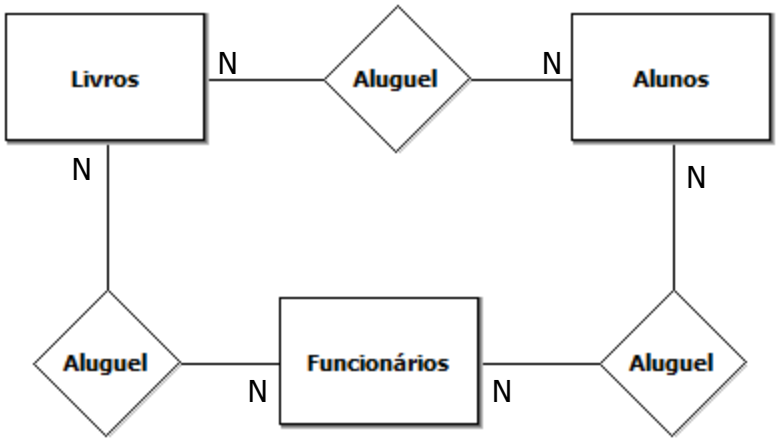
7.2.4.4 Auto-relacionamento

Uma entidade pode relacionar-se com ela mesma. O caso mais trivial é o de funcionários que gerenciam outros: cada um pode gerenciar vários e pode ser gerenciado diretamente por um. Nesse caso, temos um auto-relacionamento 1:N.

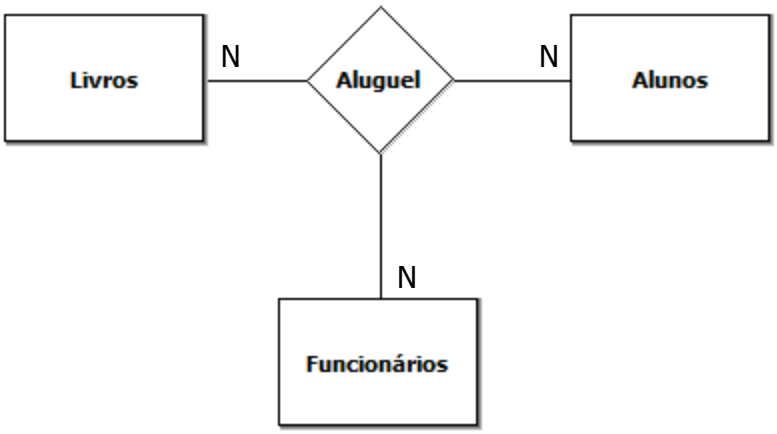


7.2.4.5 Relacionamentos Múltiplos

Nos relacionamentos vistos até agora, a ligação é sempre dupla, uma entidade relaciona-se com outra. Este é o tipo mais comum de ligação. Porém com alguma freqüência você encontra nos sistemas ligações triplas e até quádruplas. Nestes casos, nós dizemos que existe um relacionamento múltiplo. No exemplo abaixo, temos um caso de ligação múltipla. Alunos e livros estão relacionados formando o aluguel. Livros e Funcionários também só se encontram formando um aluguel. Da mesma forma, Alunos e Funcionários também se encontram no ato do aluguel. Veja o exemplo inicial:



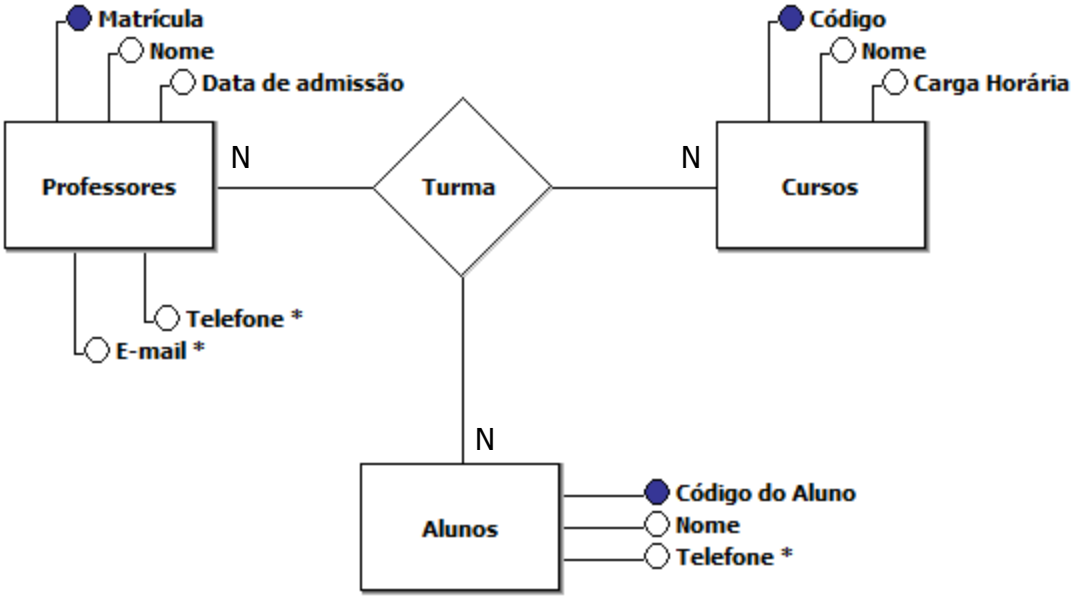
Neste caso, porém, como todos os relacionamentos possuem o mesmo nome, devemos juntá-los em um só, gerando uma figura como esta:



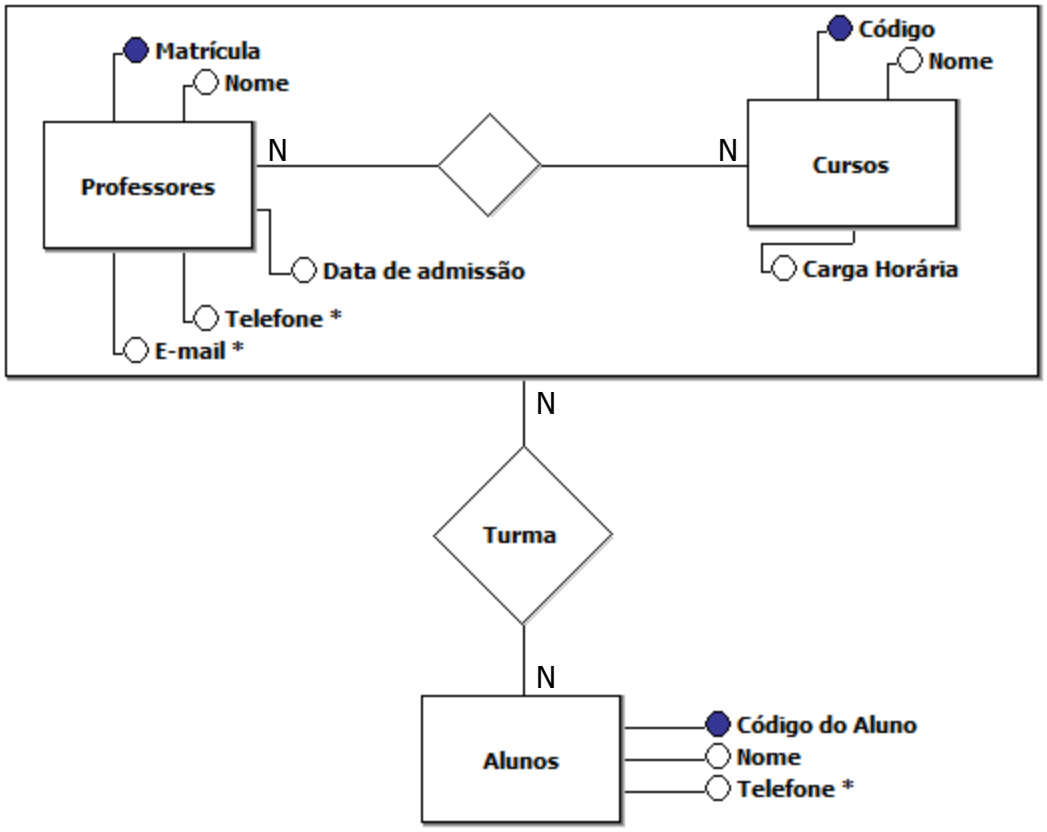
Em geral, os relacionamentos múltiplos terão cardinalidade N em todas as pontas.

7.2.4.6 Agregação

Em alguns casos de relacionamentos, temos uma ligação de uma entidade com outro relacionamento. A entidade não relaciona-se com outra mas com uma ligação entre duas entidades. Voltemos ao exemplo da escola. Sabemos que uma turma tem um professor e uma curso. Se adicionarmos uma entidade para representar os alunos, teríamos a seguinte disposição:



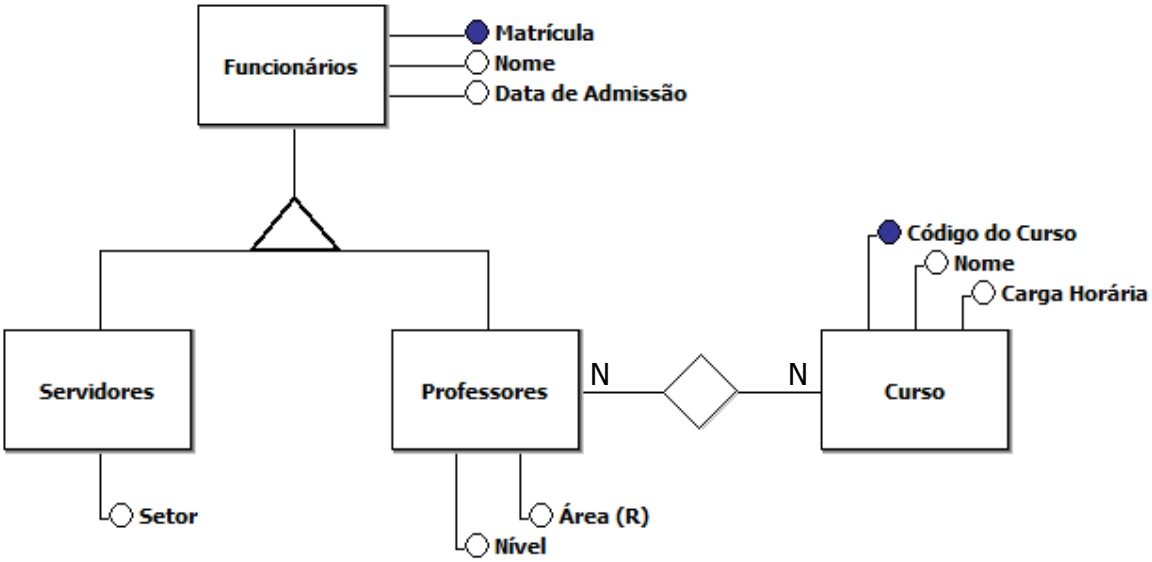
Só que o modelo, desta forma, permite que professores que não podem lecionar em um curso sejam registrados. Pois as ligações entre turmas e professores e entre turmas e cursos são independentes. Para evitar esta inconsistência de dados, usamos a agregação:



7.2.4.7 Particionamento

Às vezes, um conjunto de informações pode conter subconjuntos. Por exemplo, uma entidade funcionários de uma empresa pode conter os subconjuntos motoristas, vendedores, secretárias, etc. Funcionários não deixa de ser uma entidade, porém perceba que, para cada subconjunto, nós teremos atributos e relacionamentos específicos. Para as secretárias pode interessar o registro de quais idiomas ela fala, mas para motoristas, não. Para o vendedor registraremos qual seu setor de vendas, mas para secretárias, não. Motoristas estão relacionados a veículos, mas secretária e vendedores, não. Para evitar este problema, a entidade é particionada e os atributos e relacionamentos específicos de cada subconjunto são alocados a ele.

Para ilustrar este tipo de ligação, vamos usar um exemplo baseado numa escola. Neste existem dois tipos de funcionários: servidores e professores. Os servidores são o pessoal de secretária, serviços gerais, coordenação e administração. Para a empresa interessa registrar qual o setor de cada servidor, mas para professores, não. Cada professor está relacionado aos cursos que ele pode ministrar e tem registrada qual sua área de atuação e nível salarial, porém os servidores não.


Funcionários
Matrícula
Nome
Data de Admissão
Servidores
Setor
Professores
Área (R)
Nível
N
N
Curso
Código do Curso
Nome
Carga Horária

8. MODELO LÓGICO

O modelo lógico é o modelo que mostra toda a estrutura do banco de dados, mas é ainda independente de SGBD, ou seja, pode ser usado em qualquer banco de dados. Quando estiver pronto, podemos ter noção da estrutura e de todas as tabelas (entidades) que o sistema terá, com consistência, segurança e sem redundâncias. Após este modelo, já direcionaremos o nosso banco para o SGBD a ser utilizado, ou seja, Oracle, MySQL, SQL Server, PostgreSQL, etc.

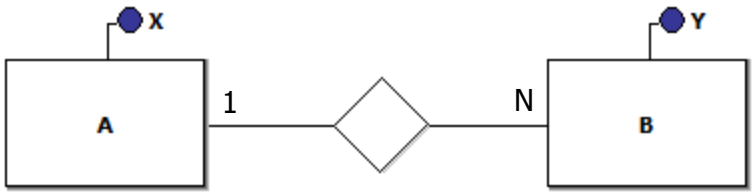
O método para se chegar ao Modelo Lógico consiste em realizar a "decomposição" do nosso modelo anterior, o Modelo Conceitual, fazendo o detalhamento total dos itens do DER. Existem vários métodos conhecidos para realizar esta tarefa. Alguns dos métodos mais conhecidos são a **Normalização** e o **Top-down**. Nos nossos estudos aqui, continuaremos usando as regras da modelagem Top-down, o que simplificará bastante nosso trabalho e chegará ao mesmo resultado!

Bem, este processo basicamente consiste tem aplicar ao nosso modelo 8 regras, pois quando terminarmos já teremos o modelo pronto. Aqui abaixo estão listada elas:

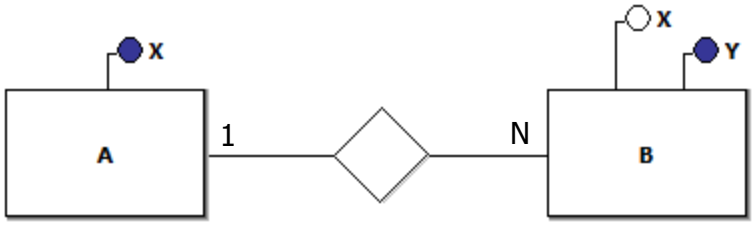
1. *1:N*
2. *Redundância Funcional*
3. *Multivaloração*
4. *N:N*
5. *Relacionamento Múltiplo*
6. *Agregação*
7. *Autorrelacionamento*
8. *Particionamento*

Veremos a seguir a aplicação de cada uma destas regras, passo a passo.

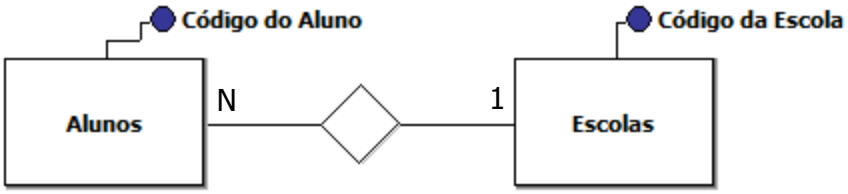
8.1 Regra para 1:N

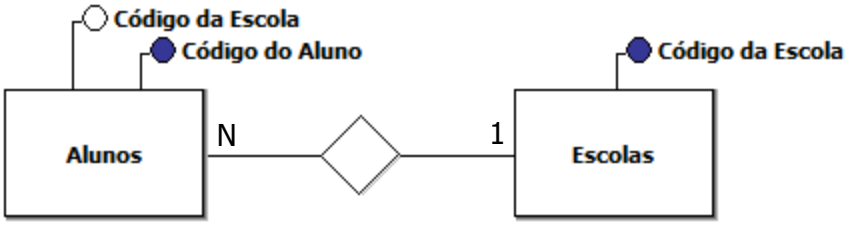


Regra: Copiamos o atributo determinante do lado 1 para o lado N, mas sem ser determinante.



Exemplo:





Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

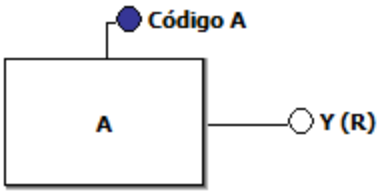
Alunos

Cód. do Aluno	Nome	Cód. da Escola
1	Alberto Machado da Silva	1
2	Antônio Machado da Silva	1
3	Clécio Grilo Meireles	2
4	Márcio de Souza Gomes	1

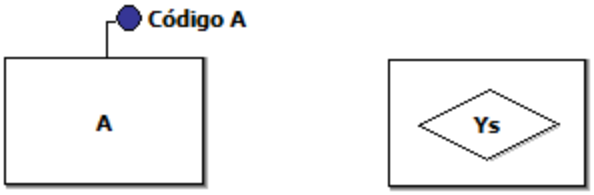
Escolas

Cód. da Escola	Gênero
1	Salesiano
2	Marista

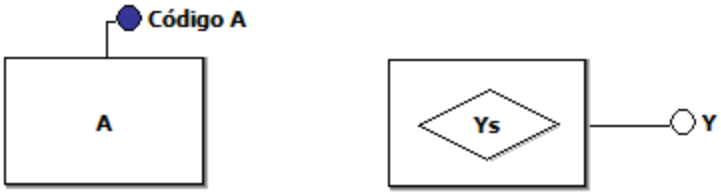
8.2 Regra para Redundância Funcional



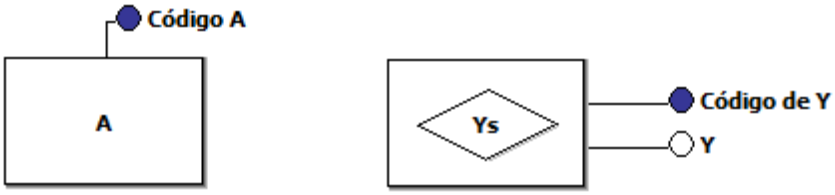
Passo 1: Remove-se o atributo em redundância e surge uma nova entidade cujo nome será o do atributo, no plural.



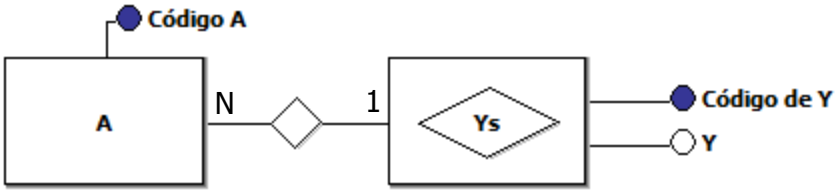
Passo 2: O atributo em redundância passa para a nova entidade sem ser redundante.



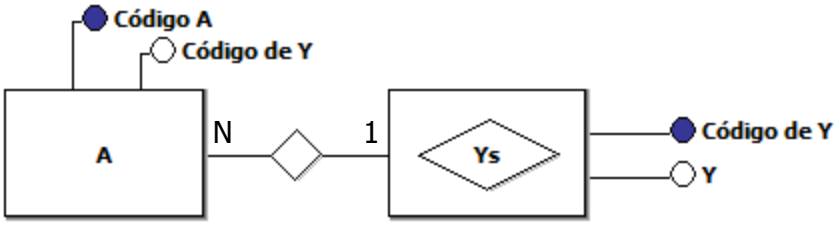
Passo 3: Cria-se um determinante artificial para a entidade nova.



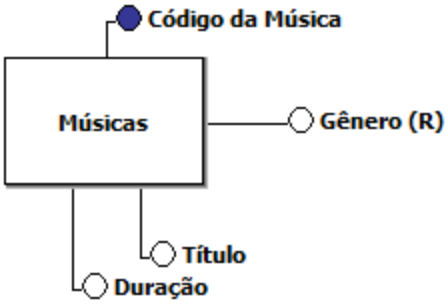
Passo 4: Surge um relacionamento de cardinalidade 1:N da entidade nova para a antiga.

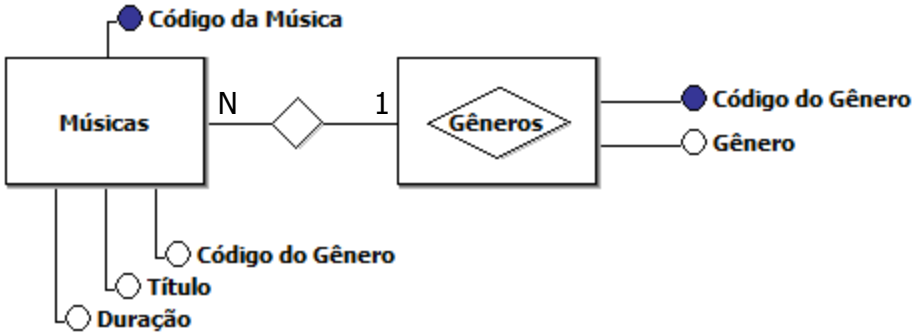


Passo 5: Decompõe-se o relacionamento 1:N.



Exemplo:





Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

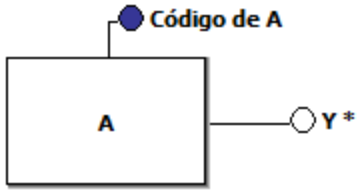
Músicas

Cód. da Música	Título	Duração	Cód. do Gênero
1	5ª Sinfonia de Beethoven	5:20	1
2	Não deixe o samba morrer	3:43	3
3	Morango do Nordeste	3:15	2
4	Mais que nada	2:55	3
5	9ª Sinfonia de Beethoven	4:18	1

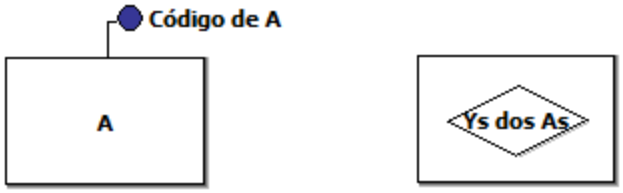
Gêneros

Cód. do Gênero	Gênero
1	Clássica
2	Forró
3	Samba

8.3 Regra para Multivaloração

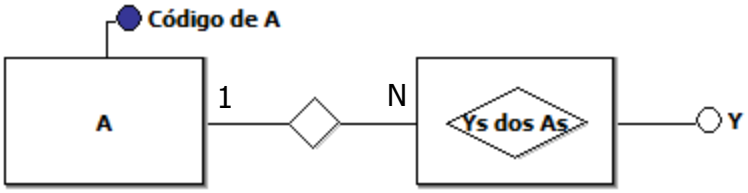


Passo 1: Apagamos o atributo multivalorado e surge uma nova entidade cujo nome será uma combinação do nome do atributo, no plural, com o nome da entidade antiga.

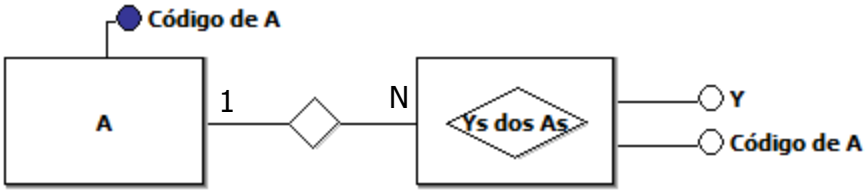


Passo 2: O atributo multivalorado passa para a nova entidade sem ser multivalorado.

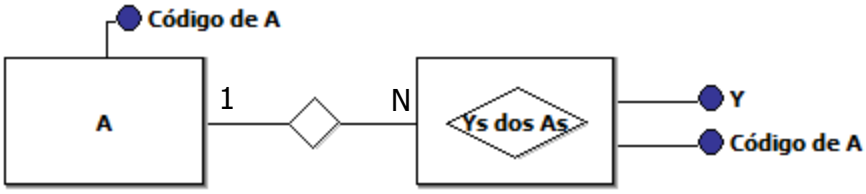
Passo 3: Surge um relacionamento de cardinalidade 1:N da entidade antiga para a nova.



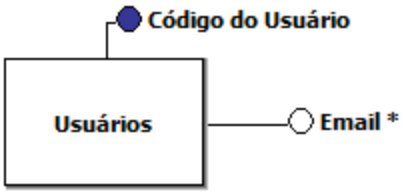
Passo 4: Decompõe-se o relacionamento 1:N



Passo 5: Os atributos da nova entidade irão formar um determinante composto.



Exemplo:



Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

Usuários

Cód. do Usuário	Nome
1	Cephas Barreto da Silva
2	Marcos Vitorino Medeiros
3	César Leonardo Pereira

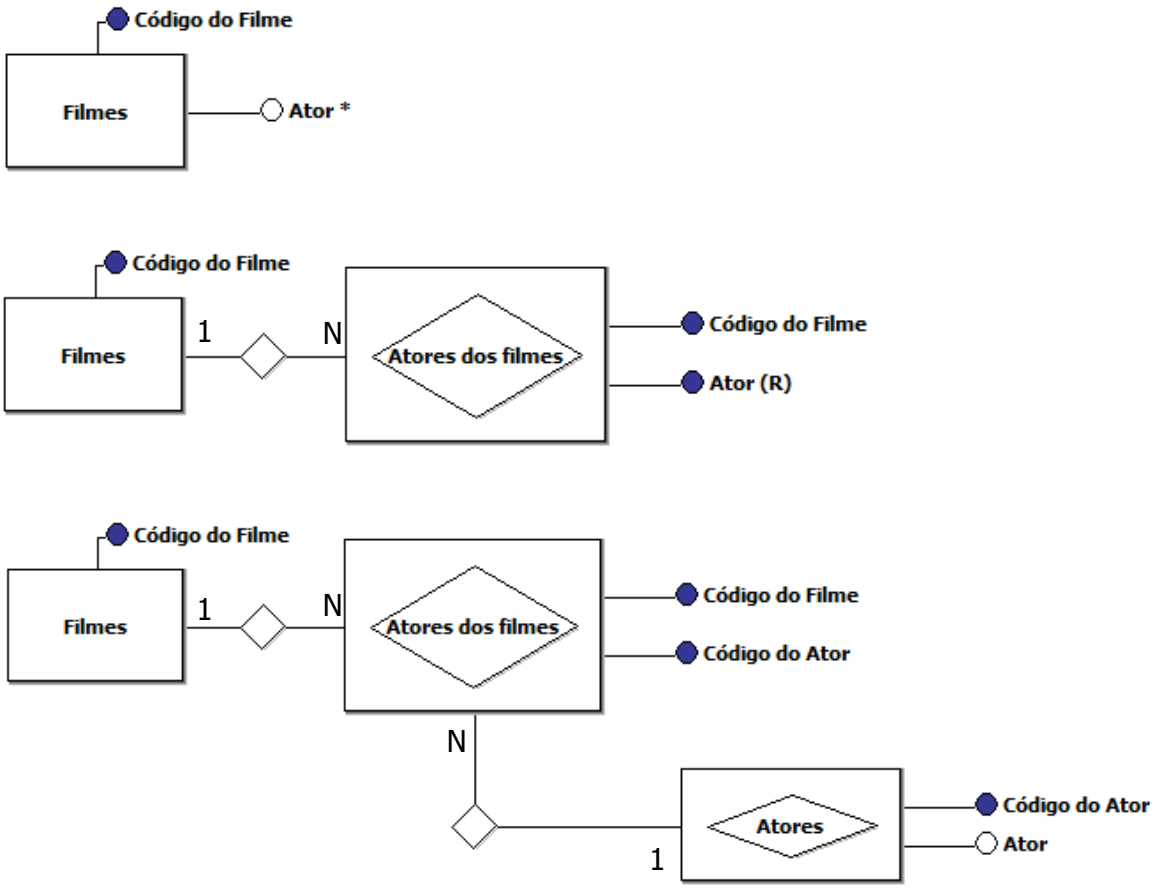
Emails dos usuários

Cód. do Usuário	Email
1	cephas@hotmail.com
1	cephas@gmail.com
2	marcosv@gmail.com
3	cesarleo2@hotmail.com
3	cesinha@gmail.com

Atenção!

Em alguns casos, após decomposta a multivaloração, o antigo atributo multivalorado cai em redundância funcional. Nesses casos a decomposição do atributo em redundância deverá ser efetuada (usando a regra da redundância funcional).

Exemplo:



Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

Filmes

Cód. do Filme	Título
1	Rota de Fuga
2	Esqueceram de mim
3	Os Mercenários

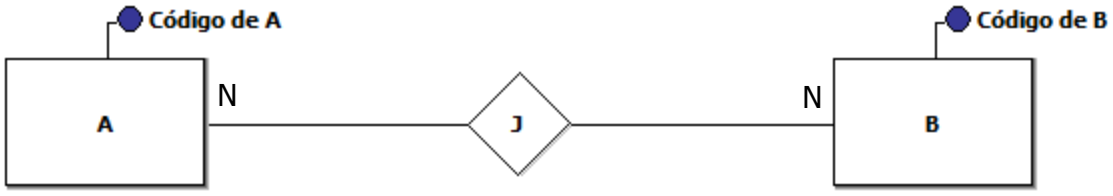
Atores

Cód. do Ator	Ator
1	Arnold Schwarzenegger
2	Macaulay Culkin
3	Sylvester Stallone
4	Jet Li

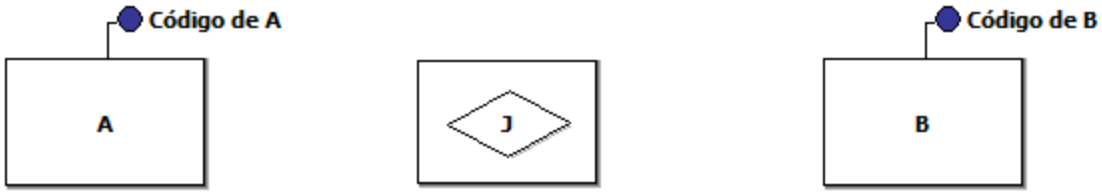
Atores dos Filmes

Código do Filme	Código do Ator
3	1
3	3
3	4
2	2
1	1
1	3

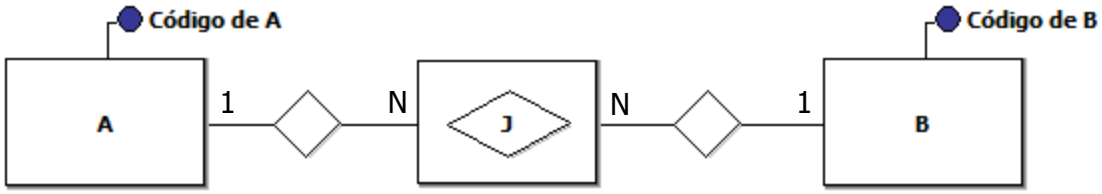
8.4 Regra para N:N



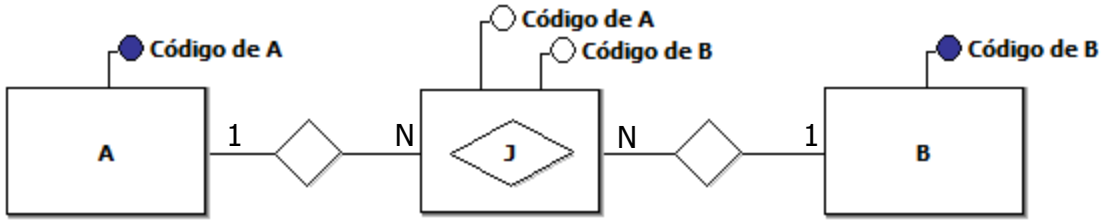
Passo 1: Quebra-se as pernas do relacionamento e surge uma nova entidade no lugar da relação.



Passo 2: Surgem duas novas relações de cardinalidade 1 (do lado da entidade antiga), para N (do lado da nova entidade).

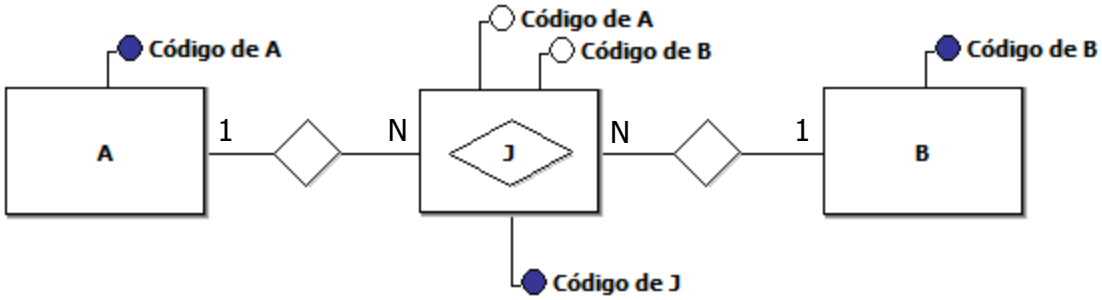


Passo 3: Decompõe-se os dois relacionamentos 1:N.

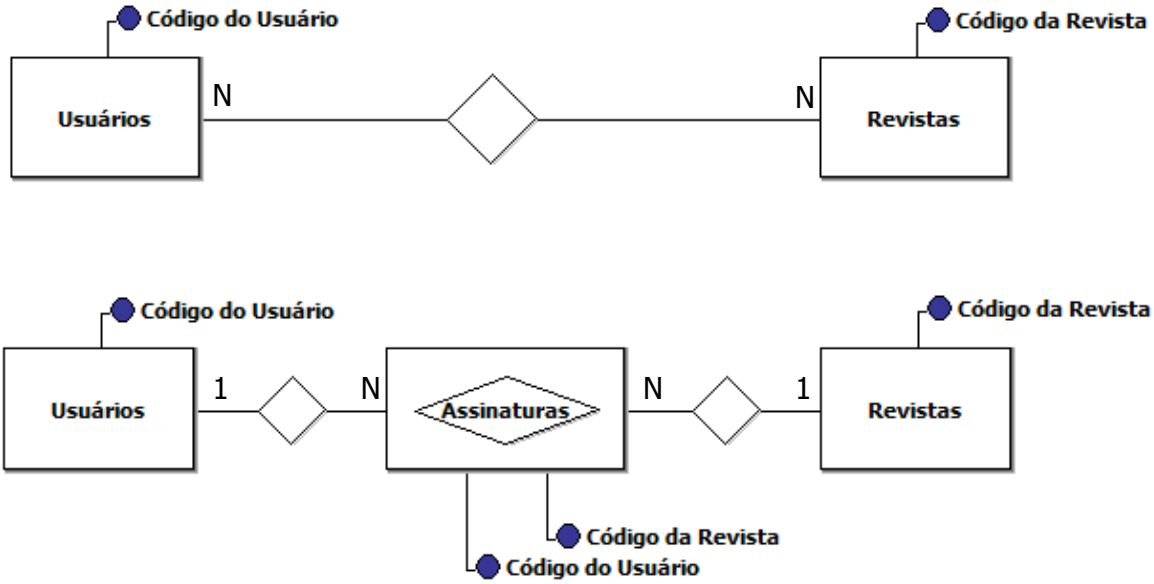


Passo 4: Deve-se definir qual ou quais atributos formarão a determinação.

Em alguns casos, pode-se criar um determinante composto (com os atributos que vieram da decomposição 1:N). Em outros casos se poderá criar um novo determinante artificial.



Exemplo:



Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

Usuários

Cód. do Usuário	Nome
1	Mikaely Dias da Silva
2	Eliézio Soares Gomes
3	Silas Alves Júnior

Revista

Cód. da Revista	Nome
1	Veja
2	Casa e Jardim
3	4rodas

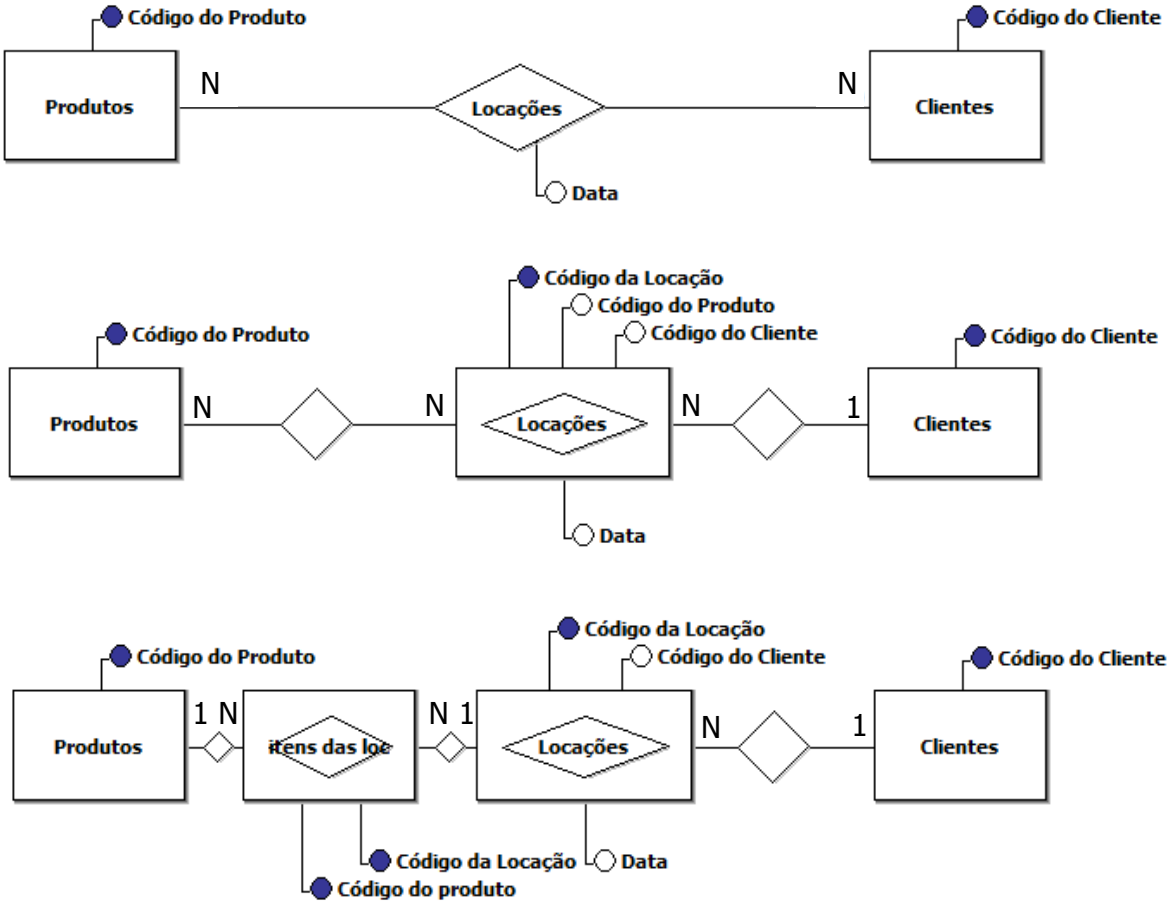
Assinaturas

Cód. do Usuário	Cód. da Revista
1	2
2	1
2	3
3	3

Atenção!

Nem sempre a regra básica de decomposição N:N funcionará, pois, cada uma das novas relações ainda podem resultar em cardinalidade N:N mais uma única vez.

Para verificar se isto ocorreu, teste a cardinalidade entre as relações. No exemplo abaixo, temos um caso de alugueis, onde numa só locação devem ser alocados muitos produtos. Veja a resolução:



Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

Clientes

Cód. do Cliente	Nome
1	Érico Barreto Marinho
2	André Varella Alves
3	Jardson Amaral Câmara
4	Cíntia de Oliveira Neta

Produtos

Cód. do Produto	Nome
1	Cadeira
2	Mesa
3	Carro utilitário
4	Toalhas

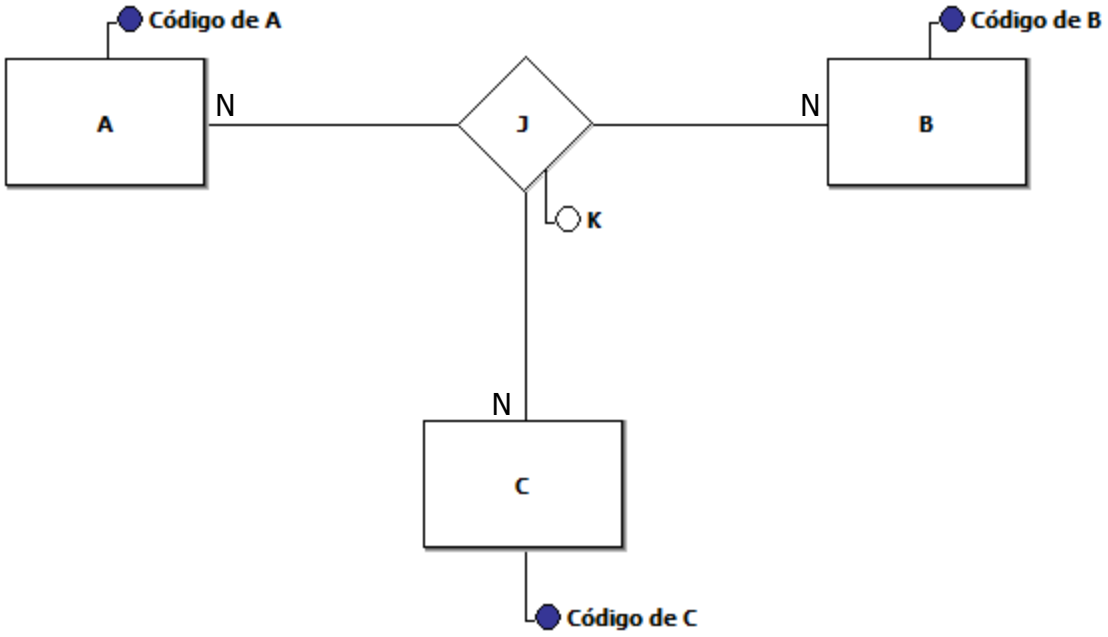
Locações

Cód. da Locação	Cód. do Usuário	Cód. do Cliente	Data
1	1	1	01/02/2014
2	2	1	05/02/2014
3	2	4	08/03/2014
4	3	2	08/03/2014

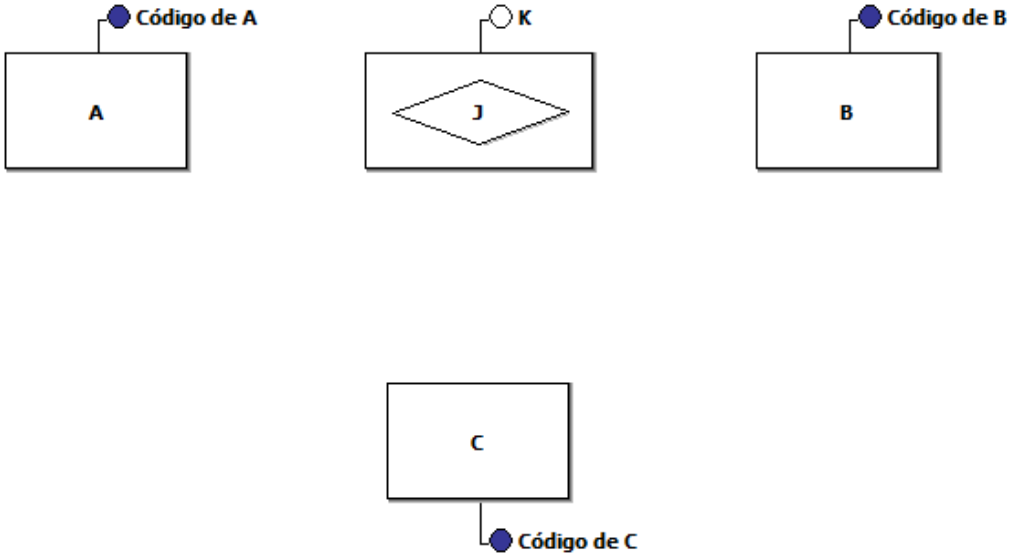
Itens das Locações

Cód. da Locação	Cód. do Produto
1	1
1	2
2	4
3	2
4	1
4	2
4	3
4	4

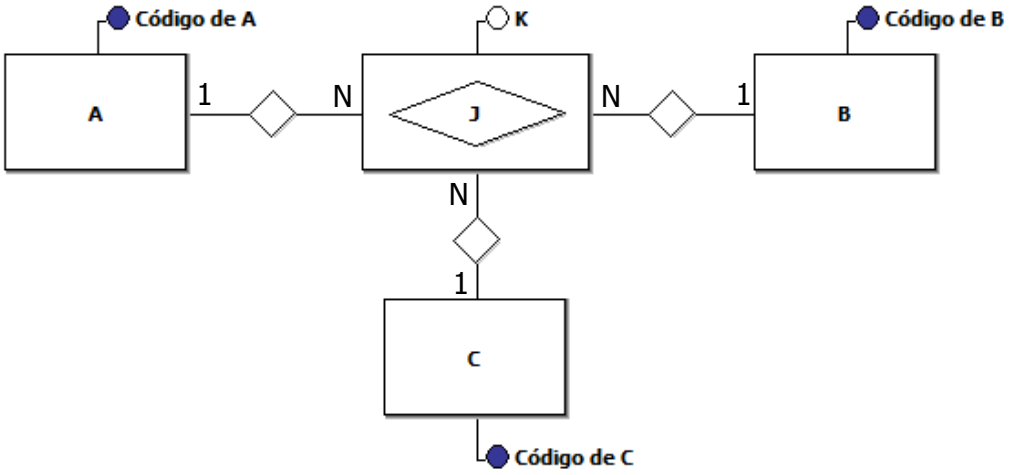
8.5 Regra para Relacionamentos Múltiplos



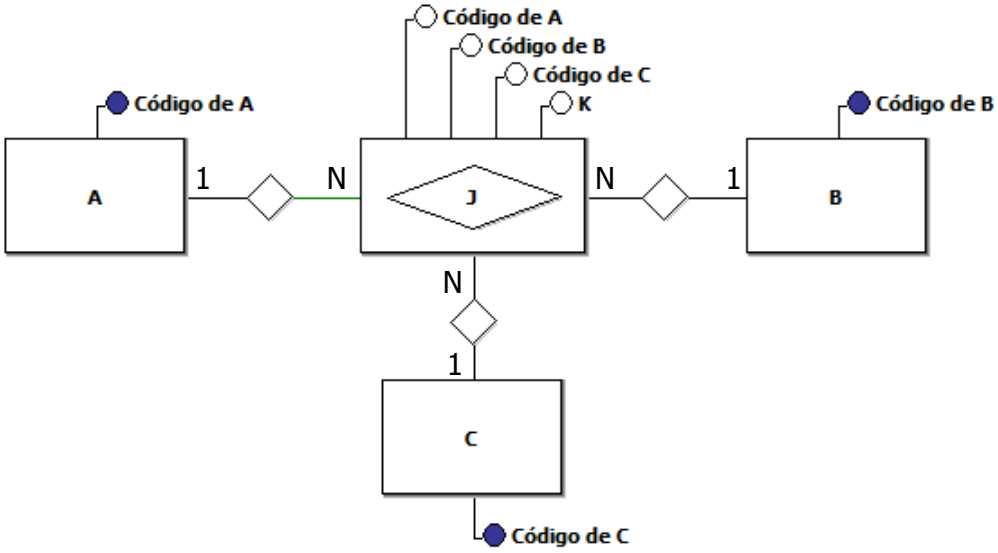
Passo 1: Quebra as pernas do relacionamento e surge uma nova entidade no lugar da relação.



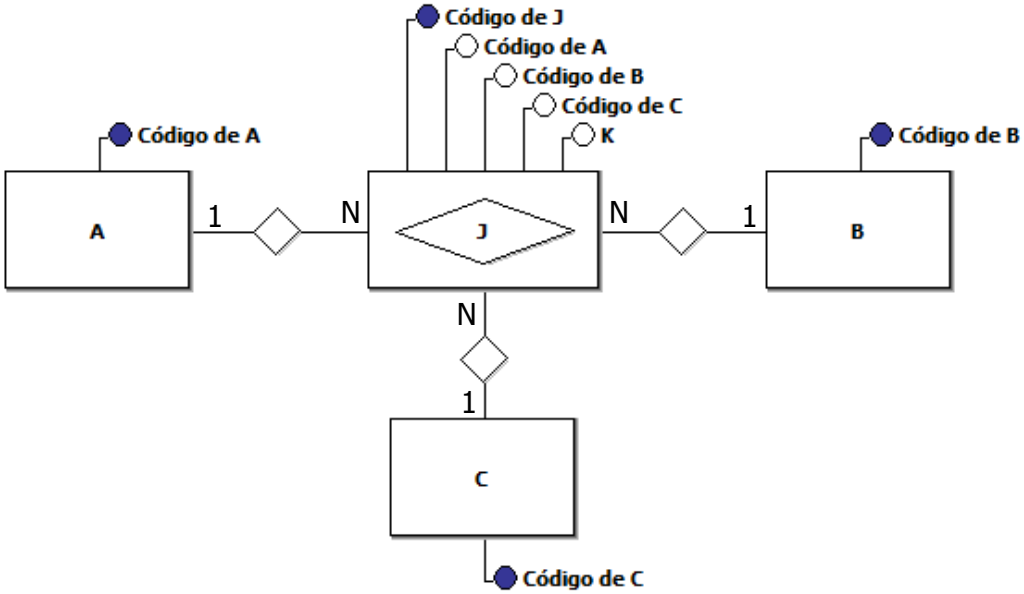
Passo 2: Surgem três novas relações de cardinalidade 1 (do lado da entidade antiga) para N (do lado da nova entidade).



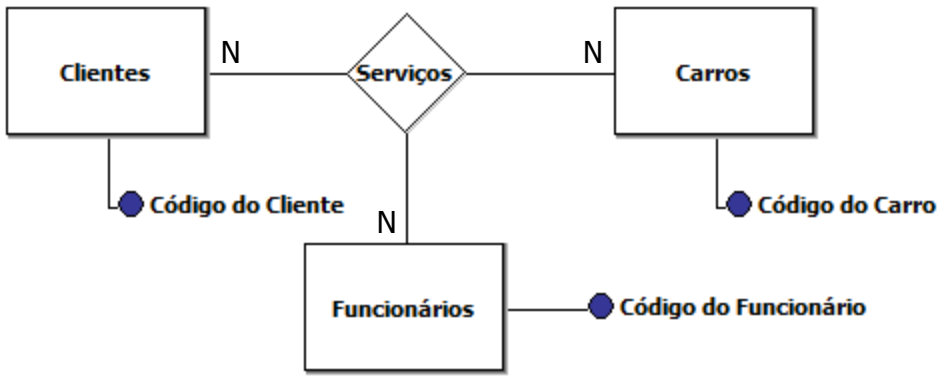
Passo 3: Decompõe-se os três relacionamentos 1:N.

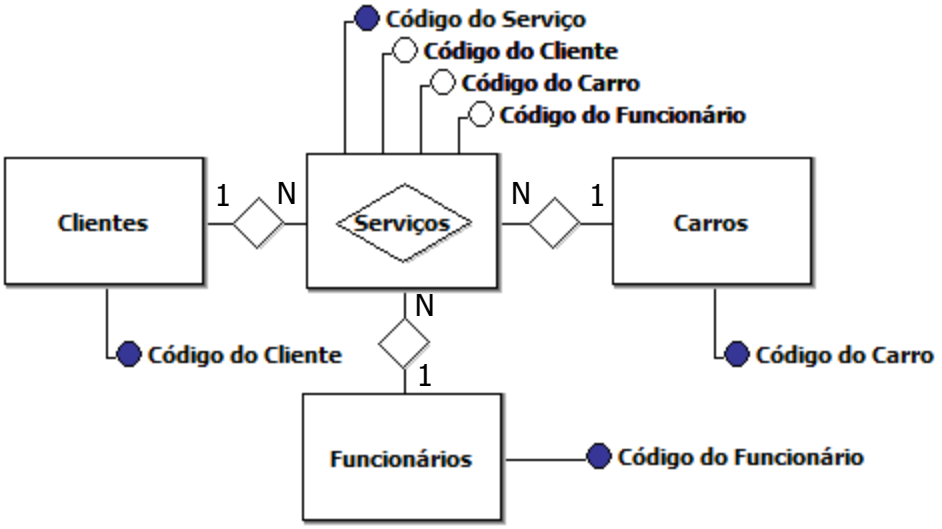


Passo 4: Deve-se definir qual (quais) atributo(s) formará a determinação (ou criar um artificial)



Exemplo:





Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

Clientes

Cód. do Cliente	Nome
1	Daely Manhães
2	Patrícia Furtado
3	Baruck Pegado

Funcionário

Cód. do Funcionário	Nome
1	Claúdio
2	Sebastião
3	Wellen

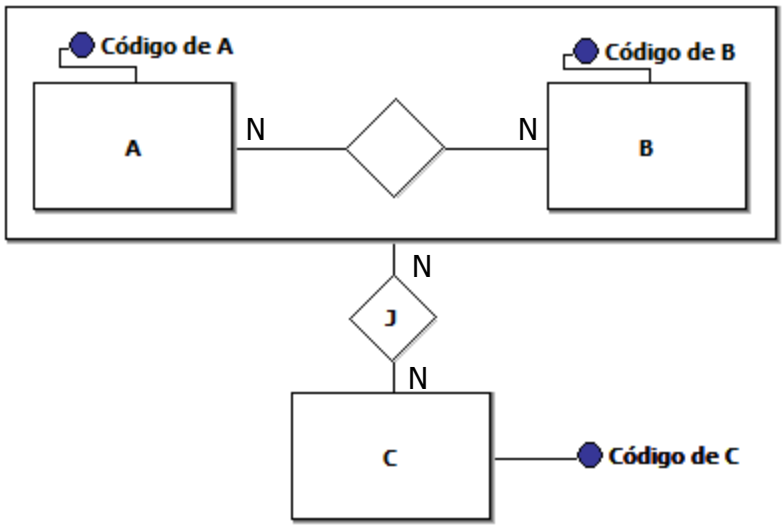
Carros

Cód. do Carro	Nome
1	Uno Mile branco 92
2	Fusca azul 76

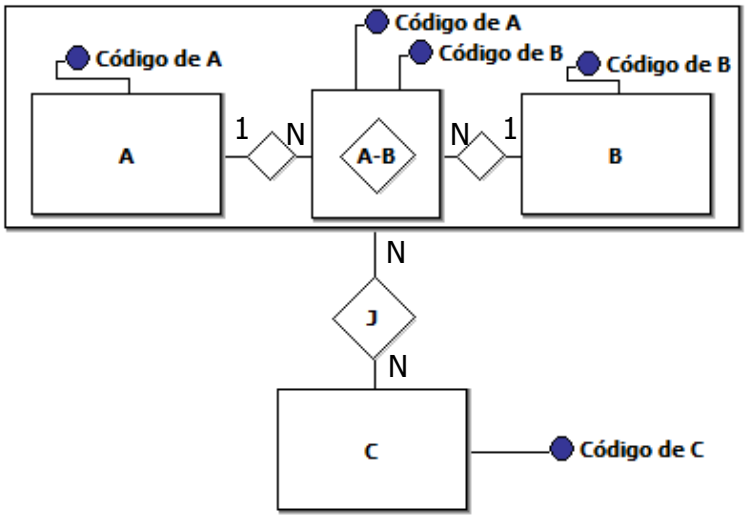
Serviço

Cód. do Serviço	Cod. do Cliente	Cod. do Carro	Cod. do Funcion.
1	1	1	2
2	2	1	3
3	2	2	3
4	3	1	1

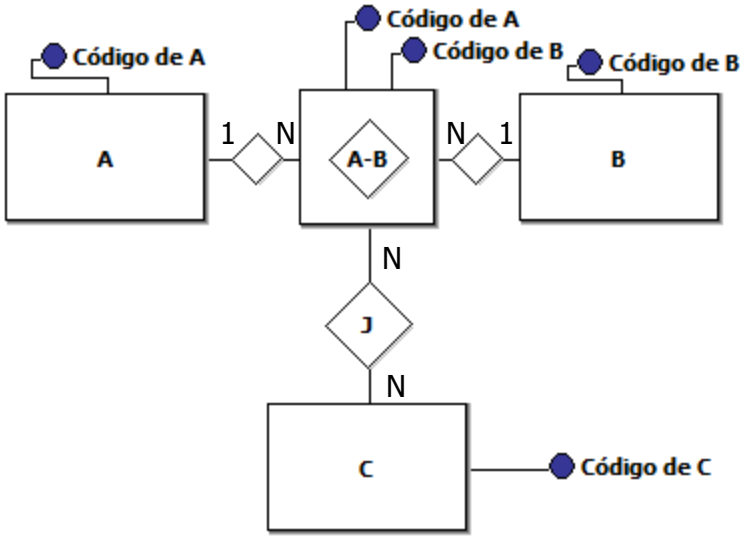
8.6 Regra para Agregação



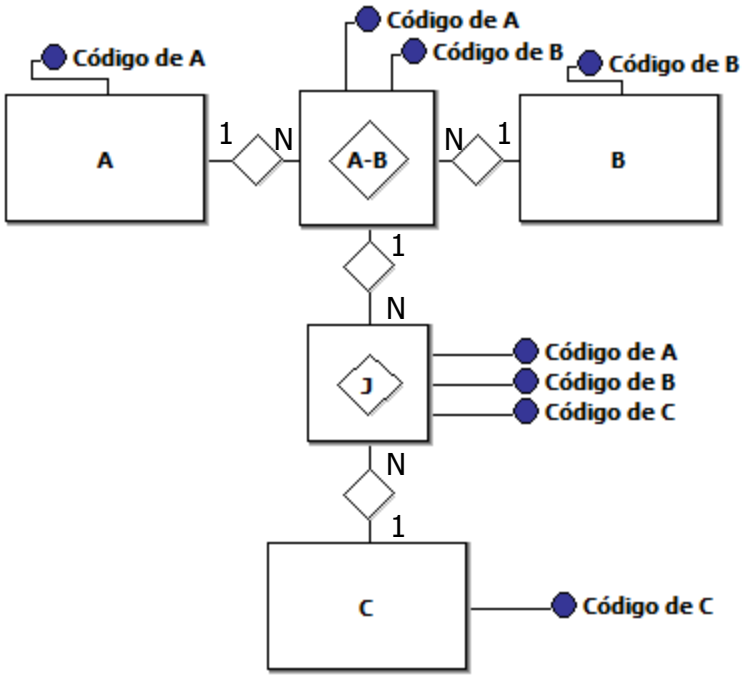
Passo 1: Decomponha a relação dentro da Agregação;



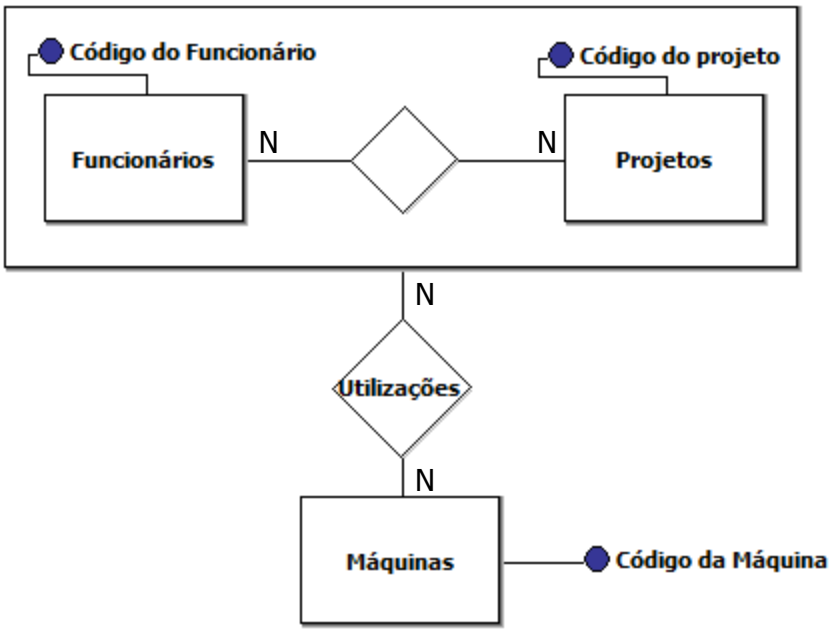
Passo 2: Ligue a entidade externa à agregação com a nova entidade;

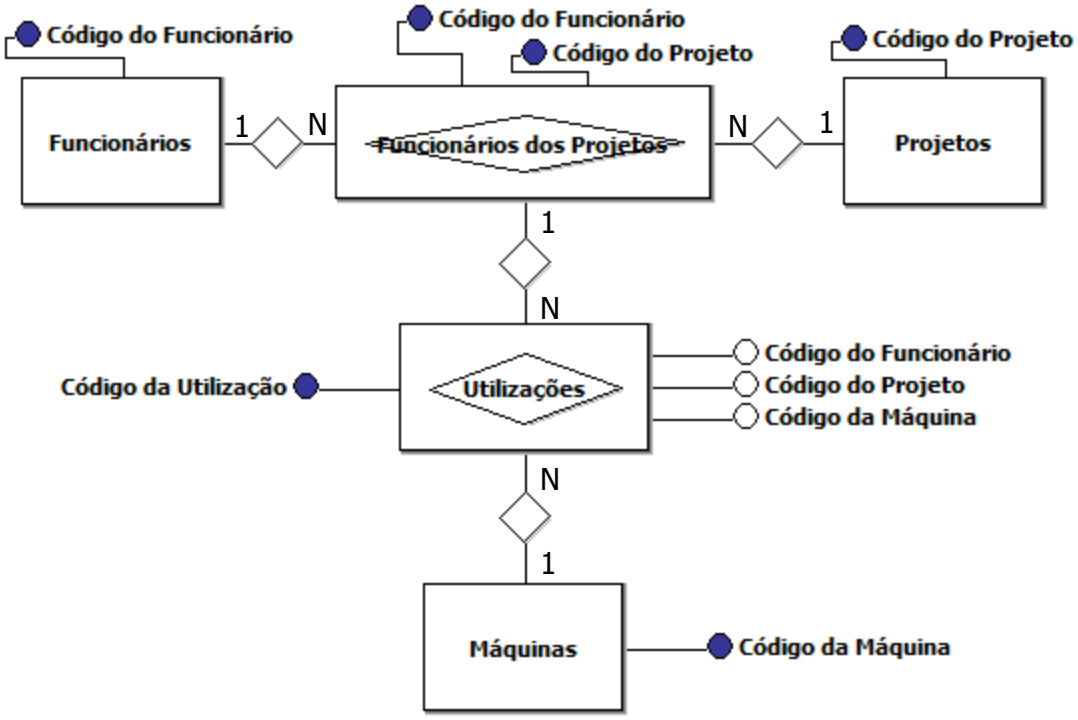


Passo 3: Decomponha o relacionamento que existia fora da agregação;



Exemplo:





Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias das entidades.

Funcionários

Cód. do Funcionário	Nome
1	Helder Pacheco Jr
2	Diego Pegado Gomes
3	Kairon Ramon Perez

Projetos

Cód. do Projeto	Nome
1	Projeto 2014
2	Projeto 2015

Máquinas

Cód. da Máquina	Nome
1	Computador
2	Impressora
3	Maq. Fotográfica
4	Cronômetro

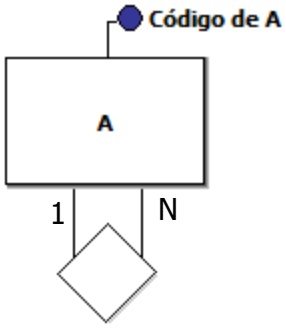
Funcionários dos projetos

Cód. do Funcionário	Cód. do Projeto
1	1
2	1
2	2
3	2

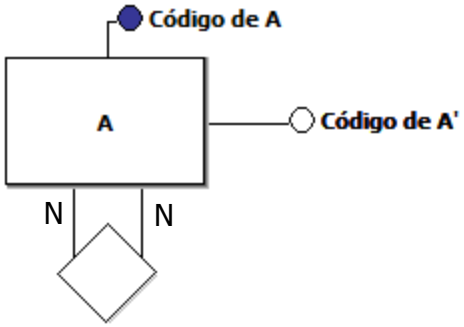
Utilizações

Cód. da Utilização	Cód. do Projeto	Cód. do Funcion.	Cód da Máquina
1	1	1	1
2	1	1	3
3	2	3	4
4	2	2	2
5	1	2	4

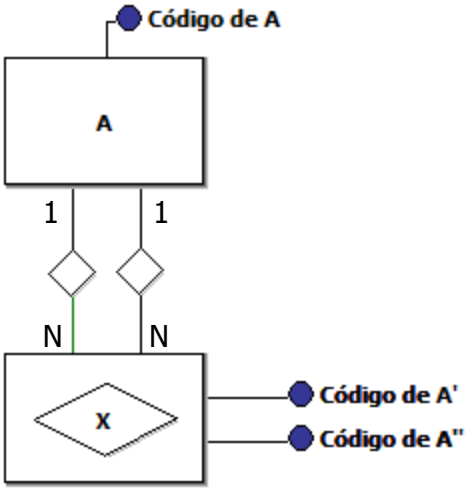
8.7 Regra para Agregação



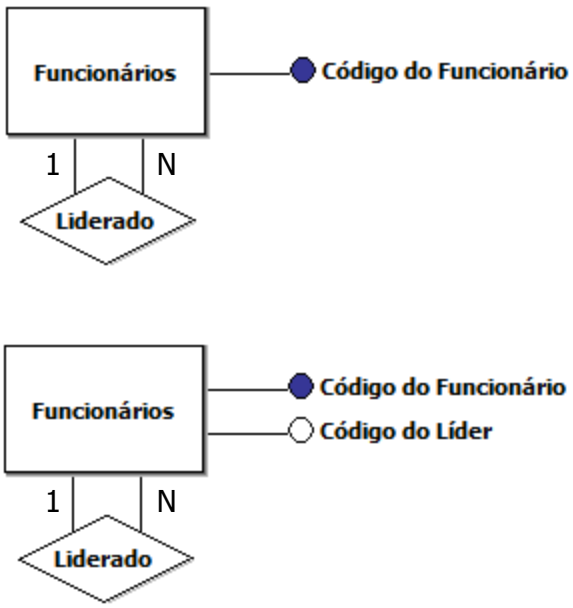
Caso 1:N - O atributo determinante se repete com outro nome, sem ser determinante.



Caso N:N
Passo1: Surge uma nova entidade e dois novos relacionamentos 1:N, da entidade antiga para a nova.
Passo 2: Os dois relacionamentos devem ser decompostos gerando no lado N dois novos atributos de nomes diferentes.



Exemplo:

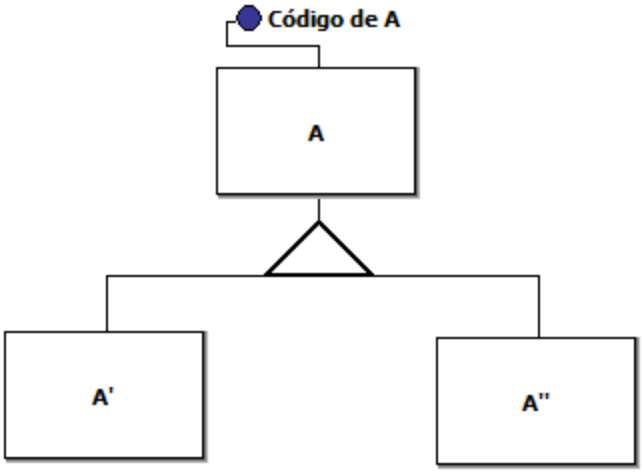


Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias da entidade.

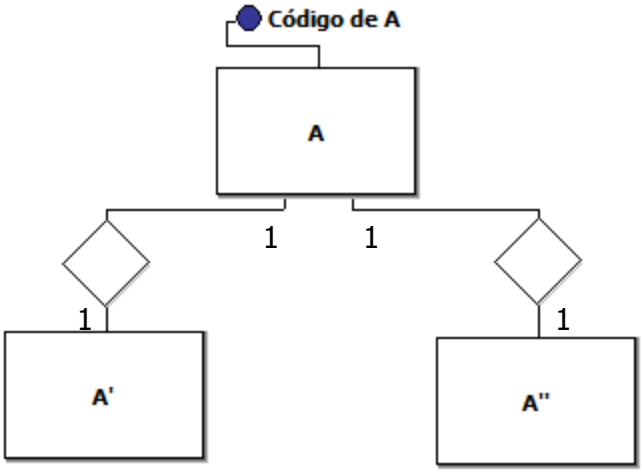
Funcionários

Código do Funcionário	Nome	Código do Líder
1	Presidente Lula Molusco	-
2	Gov. Maluf Fofo	1
3	Gov. Rosalba Boa	1
4	Profeita Micarla Anjo	3
5	Vereador Dagô do Samba	4
6	Ministro Garibaldi Bonito	1
7	Deputado Super Moura	3

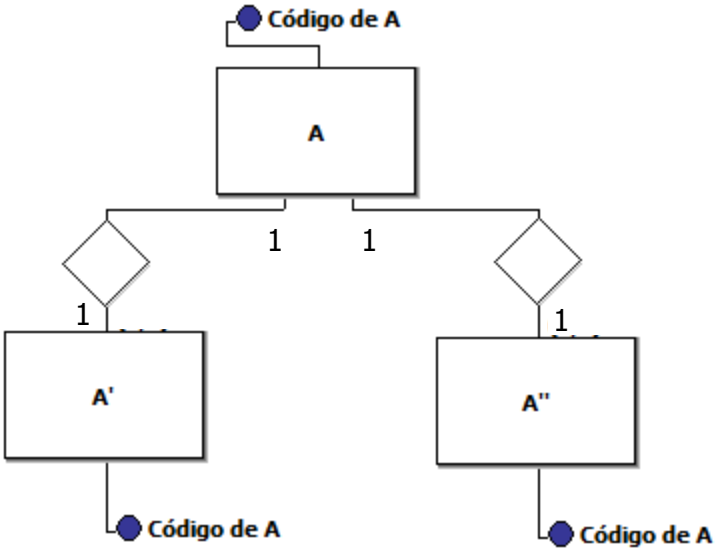
8.8 Regra para Particionamento



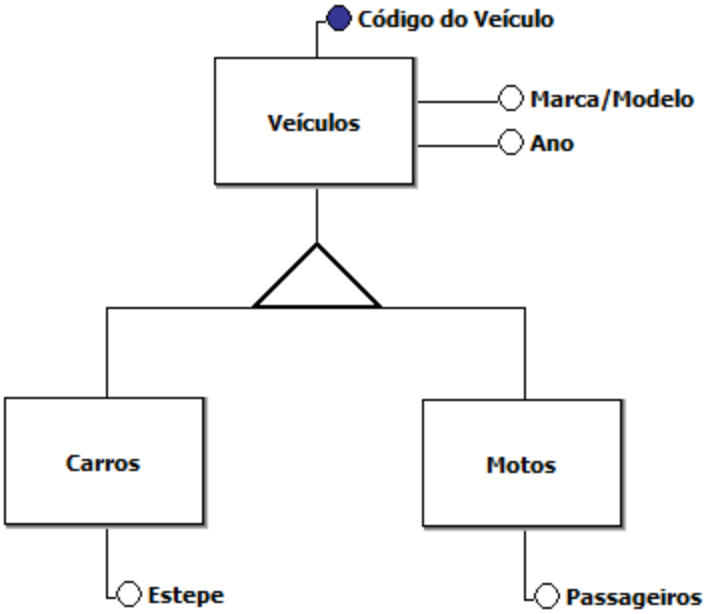
Passo 1: Quebra-se o particionamento e surgem duas novas entidades ligadas à partição principal, por dois novos relacionamentos 1:1.

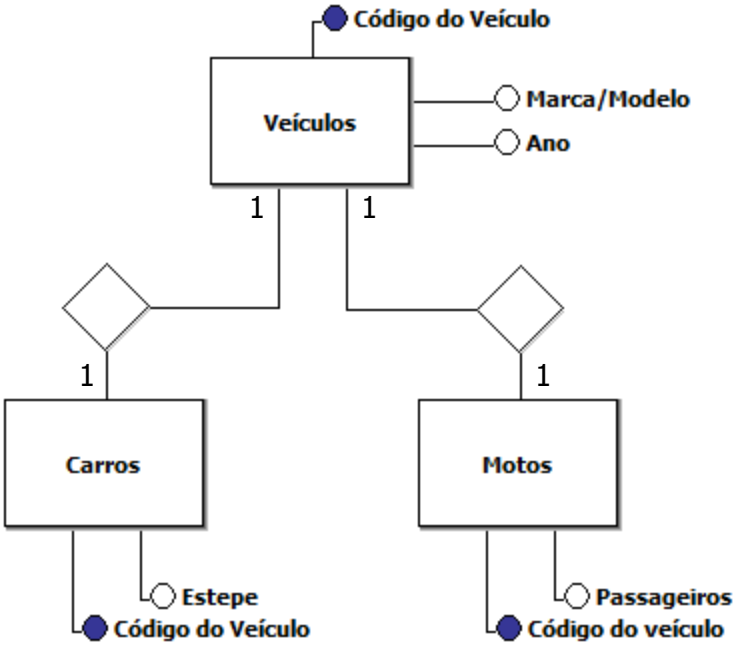


Passo 2: O atributo determinante da partição principal se repete nas novas entidades, também como atributos determinantes.



Exemplo:


Código do Veículo
Veículos
Marca/Modelo
Ano
Carros
Motos
Estepe
Passageiros


Código do Veículo
Veículos
Marca/Modelo
Ano
1
1
1
1
Carros
Motos
Estepe
Código do Veículo
Passageiros
Código do veículo

Para melhorar o entendimento dessa decomposição, vamos ver exemplos de instâncias da entidade.

Veículos

Código do Veículo	Marca/Modelo	Ano
1	Fusca/1600	1976
2	Fiat Uno Mile	1996
3	Lada Laika	1982
4	Fiat Fiorino	1986
5	Honda Fan	2002
6	Honda Pop	2009

Carros

Código do Veículo	Estepe
1	Careca
2	Não tem
3	Furado
4	OK

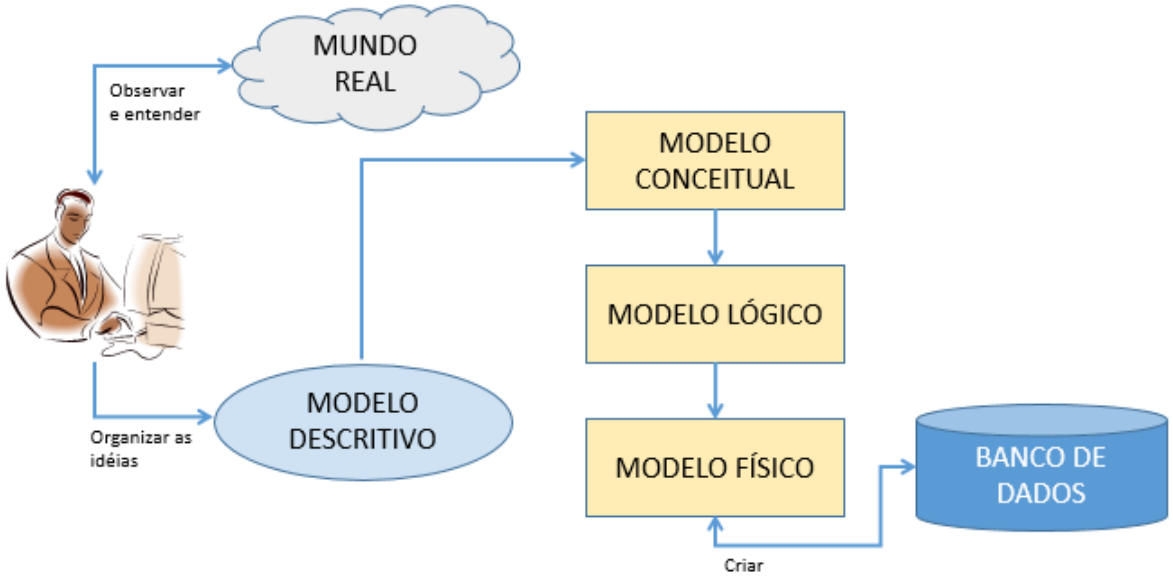
Motos

Código do Veículo	Passageiros
5	2
6	1

9. MODELO FÍSICO

Terminada a decomposição de todas as regras do **Modelo Conceitual**, temos o **Modelo Lógico** pronto. Lembre-se, se você fez tudo certinho até aqui, seu modelo não poderá ter relacionamentos N:N, agregações, atributos em multivaloração ou redundância funcional. Garantindo isto, podemos começar a transformação do modelo lógico em físico.

Reveja o esquema de todos os modelos que percorremos até aqui.



Como vemos na figura acima, o **Modelo Físico** já é muito próximo ao banco de dados. Para este novo modelo não existem muitas novidades e transformações, afinal, o trabalho grosso já foi feito. O que é essencial nesta etapa agora é escolher qual SGBD irá receber o banco de dados, pois este Modelo é específico para cada fabricante. Em outras palavras, de acordo com o SGBD escolhido – MySQL, SQL Server, o modelo poderá ter diferentes formas. Para os exemplos aqui contidos, usaremos nomenclaturas do Microsoft SQL Server.

Para fazer a conversão, mudaremos alguns termos, e este modelo assumirá o formato de tabelas e seus relacionamentos (e não mais entidades). Veja abaixo as regras para realizar a conversão entre os modelos:

- Toda entidade (ou entidade associativa) se transformará em uma **tabela**;
- Todo atributo (simples ou determinante) virará uma **coluna** desta tabela;
- Atributos determinantes serão marcados como **chaves primárias** das tabelas;
- Atributos resultantes da decomposição do 1:N serão **chaves estrangeiras**. São assim chamadas pois "apontando" para chaves primárias de outras tabelas.

Algo também importante de mencionar é que cada atributo tem o seu tipo específico de dados. No caso do SQL Server, os principais tipos de dados são:

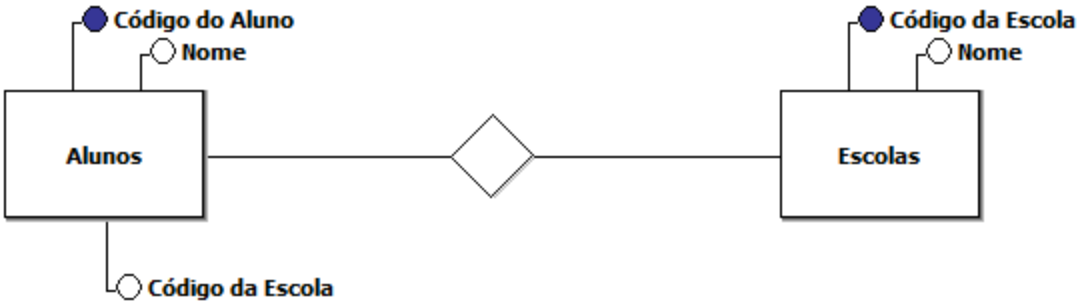
- INT – número inteiros;
- DECIMAL(*8,2*) – número com casa decimal, armazenando *8* digitos, sendo *2* digitos depois da vírgula. É alterável;
- VARCHAR(*50*) – texto com largura máxima de *50* caracteres. É alterável;
- TEXT – texto longo;
- DATE – data completa;
- DATETIME – data e hora.

Como este modelo será utilizado já diretamente pelo SGBD na criação do banco de dados, já é uma boa prática utilizar as seguintes normas de nomenclatura para todos os elementos do modelo (tabelas, colunas, chaves, etc):

- Padronize: ou escreva tudo em maiúsculas ou em minúsculas;
- Não usar acentos, ç, espaços em branco ou caracteres especiais no nome dos elementos.
- É permitido usar o "underline" _, caso deseje.

Por exemplo: "Marca/Modelo" poderia virar MARCA_E_MODELO ou simplesmente MARCAEMODELO. Locação viraria LOCACAO. São apenas sugestões para evitar problemas de compatibilidade entre servidores, que geralmente podem estar configurados em diferentes idiomas e sistemas operacionais.

Para facilitar a aprendizagem, veja abaixo um exemplo de conversão entre modelos, saindo do Modelo Lógico para o Modelo Físico.



10. BIBLIOGRAFIA

SETZER, Valdemar W. " Bancos de Dados", Editora Edgard Blucher LTDA, 1989.

SILBERSCHATZ, Abraham.Horth, Henry F., Sudarshan. S.Sistema de Bancos de Dados. Makron Books.