

# **Introdução ao Data Warehouse**

**Texto Preliminar**

Geraldo Xexéo

29 de março de 2022 17:22

Copyright ©Geraldo Xexéo, 2021.

Todos os direitos reservados.

Esta é um pré-publicação com o texto parcial em trabalho.

Apenas os alunos da cadeira de Data Warehouse para Tomada de Decisão do DC-C/IM/UFRJ estão autorizados a ter uma cópia digital e uma cópia impressa deste livro.

Contato com o autor pelo e-mail [xexeo@cos.ufrj.br](mailto:xexeo@cos.ufrj.br).

# Sumário

<b>Lista de Figuras</b>	<b>vii</b>
<b>Lista de Tabelas</b>	<b>xiii</b>
<b>Prefácio</b>	<b>1</b>
<b>I. Revisão de Bancos de Dados</b>	<b>3</b>
<b>1. Modelo de Entidades e Relacionamentos</b>	<b>5</b>
1.1.    Modelo Conceitual . . . . .	5
1.1.1.    Modelo Lógico . . . . .	7
1.1.2.    Modelo Físico . . . . .	7
1.2.    Modelo de Entidades e Relacionamentos . . . . .	7
1.3.    Diagrama de Entidades e Relacionamentos . . . . .	10
1.4.    Desenvolvendo um Modelo Conceitual . . . . .	11
1.5.    Entidades . . . . .	11
1.5.1.    Onde encontrá-las . . . . .	14
1.5.2.    Descrevendo Entidades . . . . .	15
1.6.    Relacionamentos . . . . .	15
1.6.1.    O Relacionamento de Herança . . . . .	17
1.6.2.    Cardinalidade . . . . .	18
1.6.3.    Descrevendo Relacionamentos . . . . .	22
1.7.    Atributos . . . . .	23
1.7.1.    Descrevendo Atributos . . . . .	23
1.7.2.    Atributos Identificadores (Chaves Candidatas e Chaves Primárias) . . . . .	23
1.7.3.    Relacionamentos Identificadores . . . . .	24
1.8.    Descrição Gráfica do Modelo . . . . .	24

## Sumário

1.9.	Exemplos de notação da Engenharia de Informação . . . . .	25
1.10.	Exercícios . . . . .	26
<b>2. Bancos de Dados Relacionais e SQL</b>		<b>27</b>
2.1.	Definição Formal de um BD Relacional . . . . .	29
2.2.	Sistemas Gerenciadores de Bancos de Dados . . . . .	30
2.3.	SQL . . . . .	31
2.3.1.	Data Definition Language . . . . .	31
2.3.2.	Data Query Language . . . . .	33
2.3.3.	Uma estratégia para criar consultas SQL . . . . .	34
2.3.4.	Data Manipulation Language . . . . .	38
2.3.5.	Data Transaction Language . . . . .	40
2.3.6.	Data Control Language . . . . .	40
2.4.	O que mais . . . . .	40
<b>3. Bases Exemplo</b>		<b>41</b>
3.1.	O Bando de Dados Sakila . . . . .	41
3.2.	O Banco de Dados Northwind . . . . .	46
3.3.	A Cadeia de Valor da Northwind . . . . .	49
3.4.	O Banco de Dados Adventure Works . . . . .	49
<b>II. Conceitos de Sistemas de Informação</b>		<b>51</b>
3.5.	Conceituação Genérica de Valor . . . . .	53
3.6.	Valor na Economia . . . . .	54
3.6.1.	Utilidade . . . . .	55
3.6.2.	Microeconomia Moderna e o Custo de Oportunidade . . . . .	55
3.7.	Valor em Marketing . . . . .	56
3.7.1.	Os Elementos do Valor - B2C . . . . .	58
3.7.2.	Os Elementos do Valor - B2B . . . . .	59
3.7.3.	Outras Técnicas . . . . .	62
3.8.	Priorização Baseada em Valor para o Cliente . . . . .	62
3.8.1.	Técnicas simples de ordenação . . . . .	62
3.8.2.	Método MoSCoW . . . . .	63
3.8.3.	Análise de Kano . . . . .	64
3.8.4.	Prioridade e Consenso . . . . .	65
3.8.5.	Usando mais de uma dimensão . . . . .	66
3.8.6.	Prioridade e Dependências . . . . .	68
3.9.	Valor Financeiro de Projetos . . . . .	69
3.9.1.	Custo Total de Propriedade . . . . .	70
3.9.2.	Retorno do Investimento . . . . .	71
3.9.3.	Valor Presente Líquido . . . . .	72
3.9.4.	Taxa Interna de Retorno . . . . .	73

3.10.	Valor em Software . . . . .	74
3.10.1.	Triângulo do Valor . . . . .	75
3.10.2.	Valor em Métodos Ágeis . . . . .	75
3.11.	Conclusão . . . . .	76
3.12.	Exercícios . . . . .	76
<b>4.</b>	<b>5W2H</b>	<b>79</b>
4.1.	As Perguntas Certas . . . . .	80
4.2.	Usos do 5W2H . . . . .	81
4.2.1.	O <i>Framework de Zachman</i> . . . . .	82
4.3.	Onde Está o Valor? . . . . .	82
4.4.	Várias Perguntas para cada Palavra . . . . .	83
4.4.1.	Perguntas sobre o que . . . . .	83
4.4.2.	Perguntas sobre quem . . . . .	83
4.4.3.	Perguntas sobre quando . . . . .	84
4.4.4.	Perguntas sobre onde . . . . .	84
4.4.5.	Perguntas sobre como . . . . .	85
4.4.6.	Perguntas sobre por que . . . . .	85
4.4.7.	Perguntas sobre quanto . . . . .	86
4.5.	Exercícios . . . . .	86
<b>5.</b>	<b>Modelos e Abstrações</b>	<b>87</b>
5.1.	Modelos . . . . .	88
5.2.	Tipos de Abstrações . . . . .	91
5.2.1.	Ocultação da Informação . . . . .	91
5.2.2.	Classificação . . . . .	91
5.2.3.	Generalização . . . . .	93
5.2.4.	Composição ou Agregação . . . . .	93
5.2.5.	Identificação . . . . .	94
5.2.6.	Simplificação pelo Caso Normal . . . . .	95
5.2.7.	Foco e Inibição . . . . .	95
5.2.8.	Refinamento Sucessivo . . . . .	95
5.2.9.	Separação de Interesses . . . . .	96
5.3.	Trabalhando com as abstrações . . . . .	96
5.4.	Exercícios . . . . .	96
<b>6.</b>	<b>Dados e Informação</b>	<b>97</b>
6.1.	Usando Significados Diferentes . . . . .	98
6.2.	Conclusão . . . . .	99
<b>7.</b>	<b>Sistema</b>	<b>101</b>
7.1.	Tipos de Sistemas . . . . .	103
7.2.	Sistemas Automatizados . . . . .	105
7.3.	Princípios Gerais de Sistemas . . . . .	105

<b>8. Organizações</b>	<b>107</b>
8.1. Funções Típicas de uma Organização . . . . .	109
8.2. Estrutura das Organizações . . . . .	109
8.3. Estratégia Organizacional . . . . .	111
8.4. A Necessidade de Sistemas de Informação . . . . .	111
8.5. Processo de Informatização nas Organizações . . . . .	112
8.6. A necessidade centralização dos dados . . . . .	114
8.7. Organograma . . . . .	114
8.7.1. Relações em um organograma . . . . .	115
8.7.2. Outros formatos de organograma . . . . .	116
8.8. Conclusão . . . . .	120
8.9. Exercícios . . . . .	120
<b>9. Sistemas de Informação</b>	<b>121</b>
9.1. Características dos Sistemas de Informação . . . . .	123
9.2. Sistemas de Informação Típicos e a Organização . . . . .	124
9.3. Tipos de Projetos de Sistemas de Informação . . . . .	124
9.4. Porque são feitos projetos de SI . . . . .	125
9.4.1. Benefícios do Sistema . . . . .	125
9.5. O Poder está com o usuário . . . . .	126
9.6. Exercícios . . . . .	127
<b>III. Data Warehouse</b>	<b>129</b>
<b>10.A Situação das Informações nas Organizações</b>	<b>131</b>
10.1. A necessidade de informação das organizações . . . . .	131
10.2. Processo de Informatização nas Organizações . . . . .	132
10.3. A necessidade centralização dos dados . . . . .	133
10.4. Momento Histórico da Proposta dos Data Warehouse . . . . .	135
10.5. O Data Warehouse . . . . .	135
10.6. Data Warehouse e Bancos de Dados . . . . .	137
10.7. Data Warehouses ainda são necessárias? . . . . .	137
10.8. Mais de um Data Warehouse? . . . . .	138
<b>11.Data Warehouse</b>	<b>139</b>
11.1. O Debate Kimball x Inmon . . . . .	139
11.2. Definição de Inmon para Data Warehouse . . . . .	140
11.2.1. Orientação a Assunto . . . . .	140
<b>12.Componentes do Data Warehouse</b>	<b>143</b>
12.1. Sistemas Fonte . . . . .	144
12.2. Data Staging Area . . . . .	145
12.3. Data Warehouse e Data Marts . . . . .	145

12.4. Sistemas que usam o Data Warehouse . . . . .	145
12.5. O Processo de ETL . . . . .	145
<b>13. OLAP</b>	<b>147</b>
13.1. O Cubo de Dados . . . . .	149
13.2. OLAP com Pivot Table no Excel™ . . . . .	149
13.3. Slice and Dice . . . . .	152
13.3.1. Filtros ( <i>Slice and Dice</i> ) no Pivot Table do Excel™ . . . . .	154
13.4. Drill-down e Roll-up . . . . .	155
13.4.1. Drill down e Roll Up . . . . .	157
13.5. Pivot . . . . .	160
13.5.1. Pivot no Excel™ . . . . .	160
<b>14. Modelagem Dimensional</b>	<b>163</b>
14.1. O Modelo Estrela . . . . .	164
14.1.1. A Tabela Fato . . . . .	164
<b>15. O Método Beam*</b>	<b>167</b>
15.1. Eventos de Negócio OU Histórias de Dados . . . . .	170
15.2. Descobrindo as Histórias . . . . .	171
15.2.1. Prática para Levantamento e Seleção de Eventos . . . . .	173
15.3. Passo a Passo da Modelagem de Eventos de Negócio . . . . .	173
15.3.1. A sessão de criação da Tabela BEAM* . . . . .	175
15.3.2. Tabela BEAM* Inicial . . . . .	176
15.3.3. Modelando o quando . . . . .	177
15.3.4. Primeiros Exemplos . . . . .	177
15.3.5. Refazendo o ciclo “quem-o que -como” . . . . .	180
15.3.6. Definindo onde . . . . .	181
15.3.7. Encontrando quantos . . . . .	182
15.4. Definindo por que . . . . .	186
15.5. Definindo como . . . . .	186
15.5.1. Escolhendo a Granularidade . . . . .	187
15.5.2. Dando nome ao evento . . . . .	188
15.5.3. Resumo do passo a passo . . . . .	188
15.6. Modelando Dimensões . . . . .	189
15.6.1. Codificação Usada . . . . .	190
15.6.2. Início da tabela . . . . .	191
15.6.3. Granularidade e chaves . . . . .	192
15.6.4. Os Atributos de Dimensão . . . . .	192
15.6.5. Prevendo mudanças . . . . .	195
15.6.6. Construindo a hierarquia da dimensão . . . . .	195
15.7. Matriz de Evento . . . . .	197

*Sumário*

<b>A. Exercício Um</b>	<b>209</b>
A.1. Algumas questões . . . . .	211
A.2. Orientações . . . . .	211
A.3. Dicas de implementação . . . . .	211

# **Lista de Figuras**

1.1.	As três camadas de modelo de dados. . . . .	6
1.2.	Um diagrama de entidades e relacionamentos simples, sem mostrar atributos . . . . .	8
1.3.	Um modelo de entidades e relacionamentos escrito na notação da Engenharia de Informação . . . . .	10
1.4.	Exemplo de modelo de entidades e relacionamento usando cores de acordo com o proposto por Coad, Luca e Lefebvre (1999) . . . . .	14
1.5.	Modelo ER só com entidades e relacionamentos, notação da Engenharia da Informação . . . . .	26
1.6.	Modelo ER com atributos, notação Engenharia da Informação . . . . .	26
2.1.	Exemplo de um banco de dados relacional simples . . . . .	28
2.2.	Diagrama de entidades e relacionamentos relativo ao banco de dados da Figura 2.1 . . . . .	28
3.1.	O modelo de dados da base sakila . . . . .	42
3.2.	O modelo de dados da base Sakila com os nomes de campo . . . . .	44
3.3.	O modelo de dados da base Sakila com nomes de campo e seus tipos .	45
3.4.	O modelo de dados da base Northwind só com as entidades . . . . .	46
3.5.	O modelo de dados da base Northwind com nomes de campo . . . . .	47
3.6.	O modelo de dados da base Northwind com nomes de campo e seus tipos	48
3.7.	O processo Pedir Produto da Northwind . . . . .	49
3.8.	Elementos do Valor B2C. Fonte: Almquist, Senior e Bloch, 2016 . . . . .	58
3.9.	Hierarquia das Necessidades de Maslow. Fonte: Wikipedia Commons por Felipe Sanchez (CC-BY-SA 3.0) e J. Finkelstein (GFDL) . . . . .	59
3.10.	Elementos do Valor B2B. Fonte: Almquist, Cleghorn e Sherer, 2018 .	60
3.11.	. . . . .	65

## *Lista de Figuras*

3.12.	Gráfico com quadrantes para comparar prioridades usando benefício e custo. . . . .	67
3.13.	Gráfico com quadrantes, determinados arbitrariamente pelo usuário, para os itens de projeto listados na Tabela 3.2. As setas mostram uma possível sequência de prioridade. Os itens C e D foram considerados desnecessários ao projeto. . . . .	68
3.14.	Cálculo do valor futuro de um investimento de R\$10.000,00 por mês com uma taxa de juros de 1%. . . . .	70
3.15.	Gráfico do TCO de impressoras em função da quantidade de páginas impressas. Percebe-se que após aproximadamente 3500 páginas é melhor comprar uma impressora a laser mais cara, caso esse fosse o único fator de análise. . . . .	72
3.16.	O triângulo do valor para a maioria dos projetos. . . . .	75
4.1.	O Framework de Zachman fornece uma visão da arquitetura empresarial a partir de 6 dimensões definidas pelo 5WH. Reprinted with permission by John A. Zachman, Zachman International. . . . .	82
5.1.	Diferentes tipos de mapas, ou seja, modelos, cada um com um nível diferente de abstração e dedicado a uma utilização distinta. . . . .	89
5.2.	Exemplo de Diagrama de Casos de Uso. . . . .	90
5.3.	Um brinquedo é composto de suas partes. Foto do autor . . . . .	94
6.1.	Pirâmide da hierarquia do DIKW . . . . .	99
7.1.	O Sistema Solar é um sistema natural. Imagem por Don Cloud no Pixabay  . . . . .	103
7.2.	Um <i>smartphone</i> é um sistema artificial. Imagem  . . . . .	104
8.1.	Um organograma simples, de uma empresa hipotética. . . . .	115
8.2.	Um exemplo de um organograma de um pedaço de uma empresa hipotética escrito em ARIS. . . . .	116
8.3.	A relação de subordinação, normalmente mantida na vertical, entre o gerente geral e seus gerentes subordinados, que aparecem em um mesmo nível. . . . .	117
8.4.	A relação de assessoria é normalmente feita com uma barra horizontal, como a Assessorial Internacional aparece no diagrama acima. . . . .	117
8.5.	Um exemplo de organograma radial, com uma espécie de “diagrama de Venn” indicando que os diretores e o presidente compõe a diretoria colegiada. . . . .	118
8.6.	Organograma do Poder Judiciário, em forma inversa, criado pelo site Guia de Direitos, deixado em Copyleft. . . . .	119
9.1.	Uma tela de um sistema de informações real . . . . .	122

10.1.	Diferentes cargas exigidas de sistemas OLTP e OLAP. Fonte: (Inmon, 2005) . . . . .	136
10.2.	Número de data warehouses por empresa(D. Wells e Nahari, 2019). . . . .	138
11.1.	A cadeia de valor de um mercado, descrição em ARIS. . . . .	140
11.2.	Diversas fontes de onde pode ser obtido o assunto cliente . . . . .	141
12.1.	Componentes do Data Warehouse. Inspirada em Ralph Kimball (2013)	144
13.1.	Pequena fotografia de uma planilha com os micro-dados do ENEM 1998. Fonte (INEP, 2016). . . . .	148
13.2.	Cubo OLAP simplificado para rede de lojas de calçado. . . . .	149
13.3.	Diálogo do Pivot Table do Microsoft Excel™365. . . . .	150
13.4.	Nesse quadro são controladas que colunas aparecem em que posição na pivot table . . . . .	150
13.5.	Na planilha, a pivot table vai se alterando dinamicamente para atender a configuração dada no quadro de controle. . . . .	151
13.6.	Escolhendo “Sum” como função de agregação de dados para a presença dos candidatos. . . . .	151
13.7.	Efeito do uso de um filtro selecionando apenas os dados da cidade de São Paulo, realizando um <i>slice</i> . . . . .	152
13.8.	Aplicando mais de um filtro, obtemos um <i>dice</i> . . . . .	152
13.9.	Usando o filtro para cortar algumas idades . . . . .	154
13.10.	Tabela de presenças no ENEM 1998 por estado para maiores de 17 anos. Fonte (INEP, 2016) . . . . .	155
13.11.	Hierarquias de conceitos. No caso, na hierarquia de períodos de tempo, temos dois caminhos paralelos. . . . .	156
13.12.	Fazendo um <i>drill down</i> se obtém um novo cubo mais detalhado. . . . .	157
13.13.	Pequena fotografia de uma planilha com uma pivot table verificando as presenças por cidade e estado do ENEM 1998. Fonte (INEP, 2016). . . . .	158
13.14.	Pivot table com algumas informações suprimidas por meio de um <i>roll up</i> nos estados de Acre, Alagoas, Amazonas, Amapá e Bahia. Fonte (INEP, 2016). . . . .	159
13.15.	Operação de <i>pivot</i> no cubo OLAP troca a posição das dimensões. É necessário comparar com a Figura 13.2. . . . .	160
13.16.	Posição dos dados antes da operação <i>pivot</i> . . . . .	161
13.17.	Após o <i>pivot</i> . . . . .	161
15.1.	Descrição gráfica do <i>Framework 7W</i> . Fonte:(Corr e Stagnitto, 2012) . . . . .	168
15.2.	Descrição gráfica do Modelstorming. Fonte:(Corr e Stagnitto, 2012) . . . . .	169
15.3.	Descrição do Modelstorming em ARIS-EPC . . . . .	170
15.4.	Primeiro passo da ordem de descoberta. Fonte:(Corr e Stagnitto, 2012) . . . . .	172
15.5.	O processo Modelar Detalhes de Evento, descrito em ARIS-EPC . . . . .	174
15.6.	Definir Quem-O Que-Quando, descrito em ARIS-EPC . . . . .	175

## *Lista de Figuras*

15.7.	Configuração inicial da tabela BEAM* para o evento Cliente Encorda Produto. É necessário deixar bastante espaço para continuar a atividade. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	176
15.8.	Ordem de descoberta das informações do modelo. Fonte:(Corr e Stagnitto, 2012) . . . . .	177
15.9.	Configuração da tabela BEAM* com o o detalhe “quando”. Aparece a preposição utilizada (em+a) e um sustantivo. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	178
15.10.	Configuração da tabela BEAM* com o primeiro exemplo, mostrando uma história típica. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	179
15.11.	Configuração da tabela BEAM* com o segundo exemplo, mostrando uma outra história típica, diferente da primeira. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	180
15.12.	Configuração da tabela BEAM* com um exemplo de repetição de um item. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	181
15.13.	Configuração da tabela BEAM* com um dados faltando. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	182
15.14.	Configuração da tabela BEAM* ao final deste passo. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	183
15.15.	Os três primeiros passos do processo podem ser repetidos para exaurir os conceitos de “quem” e “quanto”. Adaptado de:(Corr e Stagnitto, 2012) . . . . .	183
15.16.	Uma tabela BEAM* com dois “quem” e dois “quando”. (Corr e Stagnitto, 2012) . . . . .	184
15.17.	Uma tabela BEAM* descreve uma história que acontece em dois lugares, um em cada ponto no tempo. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012) . . . . .	184
15.18.	Introduzindo na tabela BEAM* as informações quantitativas. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012) . . . . .	185
15.19.	Introduzindo na tabela BEAM* as informações sobre por que. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012) . . . . .	186
15.20.	Introduzindo na tabela BEAM* as informações sobre como. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012) . . . . .	187
15.21.	Finalizando a Tabela BEAM* com nome e tipo. Fonte:(Corr e Stagnitto, 2012) . . . . .	188
15.22.	Modelar Dimensão descrito em ARIS-EPC. . . . .	189
15.23.	Início da definição da dimensão Cliente. Fonte: (Corr e Stagnitto, 2012)	191
15.24.	Definição da chave de negócios para a dimensão Cliente. Fonte: (Corr e Stagnitto, 2012) . . . . .	192
15.25.	Definição de um tipo para a Dimensão Cliente. Fonte: (Corr e Stagnitto, 2012) . . . . .	194

*Lista de Figuras*

15.26.	Definição dos atributos exclusivos. Fonte: (Corr e Stagnitto, 2012) . . .	195
15.27.	Tipos de hierarquias para dimensões. Fonte: (Corr e Stagnitto, 2012)	196
15.28.	Hierarquias para dimensões de data, com duas versões, e produto. Fonte: (Corr e Stagnitto, 2012) . . . . .	196
15.29.	Notações para hierarquias com loops. Fonte: (Corr e Stagnitto, 2012)	197
15.30.	Notações para consultas perpassando hierarquias. Fonte: (Corr e Stagnitto, 2012) . . . . .	197
15.31.	Exemplo de Matriz de Evento. Fonte: (Corr e Stagnitto, 2012) . . . .	198
A.1.	Modelo estrela para a Data Warehouse do exercício . . . . .	210



# **Lista de Tabelas**

2.1. Dois tipos de domínio para o mesmo conceito . . . . .	30
3.1. Resolvendo a classe de Kano por meio do cruzamento de duas perguntas (Moorman, 2012) . . . . .	64
3.2. Benefício e esforço para itens de um project backlog imaginário. . . . .	67
3.3. Valores usados para calcular o TCO de uma impressora. . . . .	71
3.4. Tabela de cálculo do Valor Presente Líquido, considerando saldos diferentes em cada período. Foi usada uma taxa de 5% a cada período. A fórmula para cada Valor Presente é $\text{Caixa}_i / (1 + \text{Taxa})^{\text{Período}}$ . . . . .	73
4.1. Perguntas e respostas a partir da notícia. . . . .	81
5.1. Sequência de imagens que mostram abstrações crescentes de uma imagem inicial que representa uma mulher. Foto por Michael Jatremski. . . . .	88
5.2. Exemplos de Classificação . . . . .	92
5.3. Exemplos de Generalização . . . . .	93
5.4. Exemplos de Composição . . . . .	94
11.1. Sistemas em um pequeno mercado. . . . .	140
13.1. Presenças por moradores dos estados no ENEM de 1998, Fonte: (INEP, 2016). . . . .	153
13.2. Presenças por moradores dos estados no ENEM de 1998, Fonte: (INEP, 2016). . . . .	154
13.3. Presenças por moradores das cidades do Acre no ENEM de 1998. Fonte: (INEP, 2016) . . . . .	157
15.1. Equivalência dos tipos de eventos do método BEAM* e os Tipos de Tabela Fato de Kimball. Fonte: Corr e Stagnitto, 2012 . . . . .	171
15.2. Tipos de eventos e seus códigos. Baseado em:(Corr e Stagnitto, 2012) . .	188

*Lista de Tabelas*

15.3. Códigos usados na modelagem dos campos da dimensão . . . . .	190
A.1. Relação entre as tabelas do DW e as tabelas da base Sakila. . . . .	209
A.2. Campos a serem calculados . . . . .	210

# Prefácio

Neste livro, a teoria de Inmon é usada, na maior parte, para explicar o papel do data warehouse na empresa e características do processo de Data Warehousing, enquanto as práticas de Kimball, e outras nele baseadas, são usadas para explicar em mais detalhes como construí-los.

Sinceramente, a adoção de uma ou de outra obra em partes deste texto é claramente uma preferência do autor, baseada no entendimento que os métodos de Kimball, em especial a Modelagem Dimensional, e os deles derivados, geram resultados mais imediatos e que isso é um grande fator de sucesso de projeto.

Como comentário em benefício do leitor, o livro mais famoso de Inmon, *Building the Data Warehouse*, apresenta uma abordagem mais de comentar as características importantes do processo, o que torna difícil seu uso imediato para o desenvolvimento de um projeto, enquanto os livros de Ralph Kimball, *The Data Warehouse Toolkit*, e de Kimball et al., *The Data Warehouse Lifecycle Toolkit: Practical Techniques for Building Data Warehouse and Business Intelligence Systems* dão uma abordagem mais objetiva para desenvolver um Data Warehouse, melhor definindo o passo a passo a ser feito. Ambos os autores possuem outras publicações que podem auxiliar no processo.

Ambos os autores possuem sites bastante interessantes:

- <https://www.kimballgroup.com/>
- <https://www.forestrimtech.com/>



## **Parte I.**

# **Revisão de Bancos de Dados**



# 1

## Modelo de Entidades e Relacionamentos

### Conteúdo

---

1.1.	Modelo Conceitual . . . . .	5
1.2.	Modelo de Entidades e Relacionamentos . . . . .	7
1.3.	Diagrama de Entidades e Relacionamentos . . . . .	10
1.4.	Desenvolvendo um Modelo Conceitual . . . . .	11
1.5.	Entidades . . . . .	11
1.6.	Relacionamentos . . . . .	15
1.7.	Atributos . . . . .	23
1.8.	Descrição Gráfica do Modelo . . . . .	24
1.9.	Exemplos de notação da Engenharia de Informação . . . . .	25
1.10.	Exercícios . . . . .	26

### 1.1. Modelo Conceitual

O objetivo da modelagem conceitual é fornecer aos desenvolvedores uma descrição abstrata de alto nível, independente de tecnologia, da informação contida no sistema. Essa descrição é conhecida como o esquema conceitual da base de dados.

## 1. Modelo de Entidades e Relacionamentos

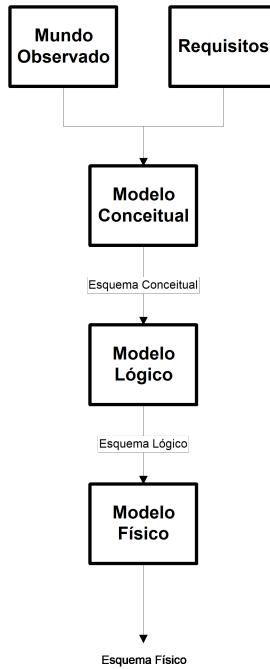


Figura 1.1.: As três camadas de modelo de dados.

A Modelagem Conceitual de Dados pode ser feita de muitas formas, algumas vezes com sutis diferenças. Alguns autores defendem a “modelagem do domínio”, onde tentamos descrever o domínio de aplicação sendo utilizados, outros tratam diretamente do sistema sendo desenvolvido.

A principal forma de construir um modelo conceitual de dados, aplicado a dados estruturados e SGDB, é usar o Modelo de Entidades e Relacionamentos (MER) (Bertini, Ceri e Navathe, 1992), na forma do Diagrama de Entidades e Relacionamentos (DER), ambos formando o tema deste capítulo. Outro modelo comum é o Modelo Dimensional (Kimball et al., 2008).

Um dos subsídios mais importantes para a criação do DER é o conjunto de regras de negócio levantadas. Muitas das regras de negócio são representadas diretamente no modelo conceitual. Veremos mais tarde que termos e fatos são candidatos naturais para serem objetos nos nossos modelos conceituais. Não devemos confundir, porém, regras de negócio com modelos de dados. Um relacionamento em uma regra de negócio pode representar uma função do sistema, enquanto um relacionamento no MER representa algo que deve pertencer à memória do sistema.

### 1.1.1. **Modelo Lógico**

O modelo lógico descreve a informação contida no sistema de acordo com uma tecnologia adotada, sem utilizar, porém, detalhes de implementação. Ele descreve a estrutura do banco de dados que será processado por um SGDB.

Atualmente, o modelo mais utilizado é o modelo relacional. Além dele, alguns modelos distintos podem ser encontrados em aplicações especiais, como data-warehousing e sistemas de informação geográfica.

### 1.1.2. **Modelo Físico**

No modelo físico devemos levar em conta não só a tecnologia sendo utilizada, mas também os produtos específicos e a interação do sistema com o ambiente de desenvolvimento e operação. É nessa etapa que nos preocupamos com as principais questões de desempenho, como escolha de índices, particionamento, etc.

## 1.2. **Modelo de Entidades e Relacionamentos**

O **Modelo de Entidades e Relacionamentos** (MER), segundo Cougo (1999), descreve o mundo como: "...cheio de coisas que possuem características próprias e que se relacionam entre si". Essas descrições são feitas sobre mundos restritos, e se referem a informação necessária para que um sistema de informações cumpra suas funções. Normalmente a descrição é feita na forma de um diagrama, o diagrama de entidades e relacionamentos.

Existem várias propostas de representação do MER. Em quase todas elas, no diagrama de entidade e relacionamentos cada tipo de entidade é representado por um retângulo, identificado pelo nome da entidade.

A Figura 1.2 mostra um diagrama de entidades e relacionamentos simples, usando uma evolução da notação original de Chen (1990) proposta por Bertini, Ceri e Navathe (1992). Esse diagrama, que descreve um mini-mundo sobre novelas, mostra as entidades Novela, Capítulo, Ator, Diretor, Ator Horista e Horas. Ele também apresenta os relacionamentos Dirige, Compõe, Atua, Pode ser e Trabalha, que ligam, cada um, duas dessas entidades. Os números indicam a cardinalidade da participação das entidades nos relacionamentos. Assim, a leitura do modelo da Figura 1.2 é:

- Entidades do modelo: Diretor, Novela, Capítulo, Ator, Ator Horista e Hora
- Um diretor dirige no máximo uma novela, podendo não dirigir nenhuma, e uma novela é dirigida por um e apenas um diretor.
- Um capítulo compõe uma e apenas uma novela e uma novela tem no mínimo um capítulo, podendo ter um número ilimitado deles.

## 1. Modelo de Entidades e Relacionamentos

- Um ator atua em uma novela, podendo não atuar em nenhuma e uma novela tem ao menos um ator, podendo ter vários.
- Um ator pode ser um ator horista e um ator horista é obrigatoriamente um ator.
- O registro do trabalho de ator horista é feito por uma entidade chamada Horas, que indica a quantidade de horas trabalhadas.

Algumas observações podem ser feitas nesse modelo. A primeira é que os capítulos não estão sequenciados. Na prática, algum atributo deverá ser adicionado ou a “Capítulo”, ou ao relacionamento “compõe” para caracterizar o número do capítulo. A segunda é que a entidade “Horas” não assume valores, devendo ter atributos que contenham ou permitam calcular a quantidade de horas trabalhadas.

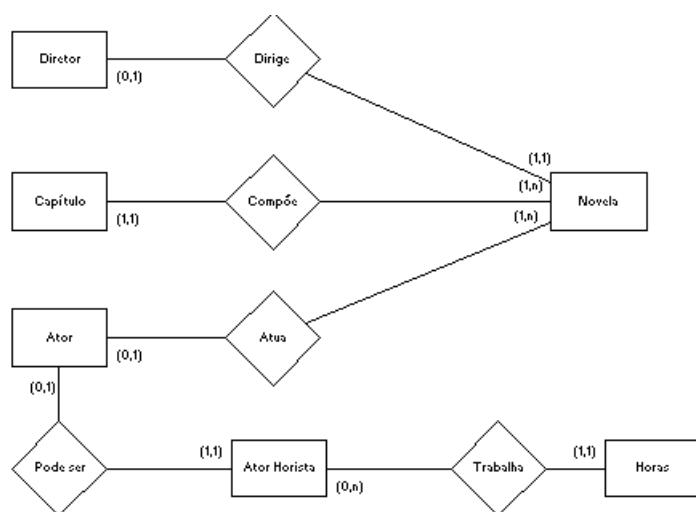


Figura 1.2.: Um diagrama de entidades e relacionamentos simples, sem mostrar atributos

As coisas as quais o MER se refere podem ser pessoas, objetos, conceitos, eventos, etc., existentes ou imaginárias. Elas são classificadas em **entidades**. Alguns autores preferem o termo **tipo de entidade** ao termo entidade. Este texto usa os termos entidade e tipo de entidade indiferentemente, para representar a classe.

*A priori*, só exigimos de uma entidade que cada um dos seus membros possa ser identificado distintamente, isso é, tenha identidade própria. Cada coisa distintamente identificada é uma instância.

Para representar os objetos específicos, os membros da classe, é adotado o termo **instância**, ou **instância de entidade**. Porém, no discurso normal, a palavra entidade também é muitas vezes usada no lugar de instância.

Uma entidade representa uma classe de objetos do universo de discurso do modelo. Por exemplo, em uma universidade podemos encontrar um funcionário chamado funcionário José e uma aluna chamada Maria. José uma instância da entidade funcionário, enquanto

## 1.2. Modelo de Entidades e Relacionamentos

Maria é uma instância da entidade aluna. Funcionário e aluno são os tipos de entidade. Cada instância, então, deve poder ser identificada unicamente.

A classificação consiste em resumir uma quantidade de características comuns de objetos que tem identidade própria por meio da criação de uma classe, ou tipo, que os descreva de alguma forma. Assim, é possível saber que todos os funcionários, por serem instâncias de um mesmo tipo, possuem características comuns (como trabalhar na universidade, ter um salário, etc.).

Matematicamente, entidades, ou tipos de entidade, são conjuntos. Os elementos desse conjunto são as instâncias. Em um modelo, as entidades representam todas as instâncias possíveis. No banco de dados que implementa esse modelo estarão apenas as instâncias que interessam ao sistema.

Apenas algumas entidades do mundo real (ou imaginário) são de interesse para o sistema. Durante a modelagem conceitual nos preocupamos com as “coisas” que o sistema deve lembrar e colocamos essas “coisas” no modelo de entidade e relacionamentos. Uma entidade deve ser relevante para o objetivo do negócio e necessária para a sua operação.

Cada entidade tem dois tipos de características importantes: seus atributos e seus relacionamentos.

Os **atributos** são características que toda a instância de um tipo possui, mas que podem variar entre as instâncias. Uma instância do tipo “aluno” tem os atributos “nome” e “ano de matrícula”, por exemplo.

Atributos caracterizam a informação que deve ser guardada sobre uma entidade. Só devemos colocar como atributos aquelas informações que o sistema precisa lembrar em algum momento. Assim, uma instância de “aluno” não precisa ter o atributo “nome do animal de estimação” em um sistema acadêmico, pois apesar de ser algo importante para o “aluno” propriamente dito, não tem importância alguma para o sistema.

A Figura 1.3 mostra um diagrama de entidades e relacionamentos sobre uma locadora de fitas de vídeo<sup>1</sup>.

Cada característica deve possuir um **domínio**. O domínio indica o conjunto de valores válidos para a característica. No caso de “nome”, geralmente aceitamos qualquer sequência de caracteres, enquanto no caso de “altura” podemos aceitar apenas valores reais positivos menores que 2,5.

Atributos eram originalmente descritos por círculos no modelo E-R. As notações mais modernas anotam os atributos dentro dos retângulos da entidade a que pertencem, como na Figura 1.3.

Finalmente, como indica o nome do modelo, entidades podem se relacionar entre si. Essa característica é a principal força do modelo de entidades e relacionamentos, pois permite que, de certa forma, “naveguemos” no modelo.

---

<sup>1</sup>O que mostra a idade do autor.

## 1. Modelo de Entidades e Relacionamentos

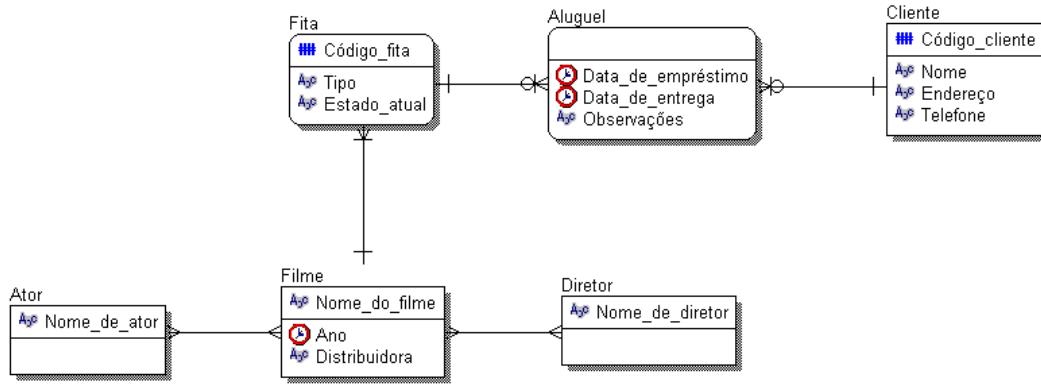


Figura 1.3.: Um modelo de entidades e relacionamentos escrito na notação da Engenharia de Informação

Podemos indicar relacionamentos apenas pelas entidades envolvidas, como “cliente-pedido”, ou usar um termo como “solicita” que descreva o relacionamento “cliente solicita pedido”.

Modelos de Entidades e Relacionamentos para serem completos exigem também um conjunto de restrições. Algumas dessas restrições, como a cardinalidade dos relacionamentos que veremos a seguir, podem ser descritas em algumas (ou todas) notações. Porém, a maioria das descrições é muito complexa para ser descrita em um diagrama. Nesse caso são necessárias anotações ao diagrama descrevendo as descrições. Isso pode ser feito em linguagem natural ou em alguma notação formal específica, dependendo de escolhas da equipe de projeto ou do método utilizado.

### 1.3. Diagrama de Entidades e Relacionamentos

Existem muitas notações para Diagrama de Entidades e Relacionamentos. A notação original foi proposta por Chen e é composta de entidades (retângulos), relacionamentos (losangos), atributos (círculos) e linhas de conexão (linhas) que indicam a cardinalidade de uma entidade em um relacionamento. Chen ainda propõe símbolos para entidades fracas e entidades associativas.

As notações modernas abandonaram o uso de símbolos especiais para atributos, incluindo a lista de atributos, de alguma forma, no símbolo da entidade. Consideraremos as notações mais interessantes na atualidade:

- Engenharia de Informação, também conhecida como *crow's foot* ou notação de Martin, bastante difundida e também presente como notação alternativa no ERWIN, exemplificada na Figura 1.3;

#### *1.4. Desenvolvendo um Modelo Conceitual*

- Notação de Bertini, Ceri e Navathe (1992), pouco difundida, mas com aspectos teóricos interessantes, ou
- Uso da UML para representar modelos de dados não-orientados a objetos.

Além disso, a notação IDEF1X, utilizada pela ferramenta ERWIN, é bastante difundida no mercado(NIST, 1993);

Toda a notação moderna tem como característica importante definir a cardinalidade mínima e máxima em uma relação, não utilizar um símbolo especial para relacionamentos, mas apenas a linha, e descrever atributos dentro do símbolo de entidade.

## **1.4. Desenvolvendo um Modelo Conceitual**

Desenvolver um modelo conceitual correto para um sistema, completo e consistente, não é uma tarefa fácil. Já desenvolver um modelo conceitual razoavelmente correto, que dê uma idéia do sistema e do negócio e que, de forma evolutiva, resulte em um modelo conceitual correto, é uma tarefa razoavelmente fácil para um analista experiente. Para o analista de sistemas iniciante, porém, parece uma tarefa extremamente difícil.

Isso acontece porque o modelo conceitual de dados exige duas coisas: um alto grau de abstração e a internalização, pelo analista, de conceitos bastante vagos, como “entidades” e “relacionamentos”. Assim, o analista iniciante precisa seguir algumas estratégias para entender melhor como desenvolver um modelo conceitual.

A primeira estratégia é a estratégia dos exercícios e exemplos. Nada é tão útil ao aprendizado como colocar a mão na massa. Os exemplos, por seu lado, servem não só como orientação geral, mas também como exemplos de pontos específicos da modelagem e de como especialistas resolvem problemas de modelagem, sejam eles simples ou complicados.

A segunda estratégia é desenvolver uma lista de dicas de trabalho. Essas dicas funcionam para o analista como as pistas funcionam para um detetive, mostrando que caminho seguir até encontrar a solução do problema.

## **1.5. Entidades**

Uma **entidade** é uma pessoa, objeto, local, animal, acontecimento, organização ou outra ideia abstrata sobre a qual o sistema deve se lembrar alguma coisa.

Cada instância de uma determinada entidade tem características similares (mas não iguais), o mesmo comportamento e uma identidade própria.

O primeiro passo na determinação das entidades é o levantamento dos candidatos à entidade. Durante as entrevistas e reuniões de análise de sistema, vários objetos e conceitos serão descritos como parte do sistema. Algumas vezes esses objetos são bastante concretos, como um “produto” dentro de um “estoque”, outras vezes são descritos como

## 1. Modelo de Entidades e Relacionamentos

documentos que guardam alguma informação, como uma “nota fiscal”, outras vezes são abstratos, como um “curso”.

No discurso fluente durante uma entrevista, entidades são geralmente substantivos ocupando o papel de sujeito ou objeto, enquanto relacionamentos geralmente são encontrados na forma de verbo. Muitas vezes uma regra de negócio, como “alunos cursam turmas” ou “clientes fazem pedidos”, nos indica entidades e relacionamentos .

Outro sinal importante da necessidade de uma entidade é o fato de algo que precisa ser lembrado representar um conceito ou ideia completa. Em um sistema acadêmico precisamos nos lembrar do nome do aluno, da data de matrícula do aluno, do curso em que está o aluno, etc. O “aluno” é a nossa ideia completa que aparece várias vezes, algumas vezes caracterizado por seu nome, outras vezes por seu curso. É um bom candidato a entidade.

Alguns autores propõem uma determinação *bottom-up* das entidades, sugerindo que elas sejam construídas pela partição de todos os dados atômicos que o sistema deve lembrar (nome de aluno, data de matrícula do aluno, etc.). Assim, construiríamos uma lista de atributos para depois agrupá-los em entidades. Preferimos, porém, uma abordagem de busca direta das entidades. Um sistema com poucas dezenas de entidades pode ter centenas de atributos, o que torna a estratégia *bottom-up* mais confusa.

Aproveitando da cultura da orientação a objetos, podemos adotar a divisão proposta por Shlaer e Mellor (1999), uma entidade pode estar em cinco grandes categorias:

- **objetos tangíveis;**
- **papéis exercidos;**
- **eventos;**
- **interações;**
- **especificações;**

Podemos facilmente ver porque objetos tangíveis são bons candidatos a entidades: normalmente, sistemas de informação falam em algum momento de objetos tangíveis, como produtos e equipamentos. Algumas vezes, porém, um objeto tangível, como uma pessoa, assume uma função ou papel específico, como aluno ou professor.

Eventos, ou interações, acontecem em algum momento do tempo e representam classes importantes de entidades. Um exemplo de evento é uma “reunião” em uma agenda . Normalmente eventos exigem atributos como data e duração.

Exemplos típicos de interações são: “contratação de serviço” ou “venda de produto”. Interações são semelhantes a relacionamentos ou a objetos tangíveis ou eventos, sendo muitas vezes representadas dessa forma.

Finalmente, especificação são tipos especiais de entidades que classificam outra entidade. Um bom exemplo é “fábrica”, que é uma especificação para “automóvel”. Geralmente, especificações também podem ser implementadas como um atributo na entidade especificada, sendo essa uma decisão de análise.

Já Coad, Luca e Lefebvre (1999), ao buscarem uma forma mais objetiva de modelar objetos, sugerem que busquemos inicialmente 4 tipos de objetos (que podemos entender como entidades):

- **momentos ou intervalos** - cor rosa: um momento ou um intervalo representa qualquer coisa que precisa ser acompanhada, por motivos de negócio ou legais, e que acontecem em um instante de tempo ou por um período de tempo. Muitas vezes pode ser mais fácil começar nossa análise por esse tipo de entidade, pois estamos tratando de atividades de negócio que devem exigir a participação das outras entidades. Exemplos são: aulas, consultas, contratação, etc.
- **papéis** - cor amarela: representam papéis assumidos pelas pessoas que estão envolvidas com o sistema sendo analisado. Cuidado, pois não são apenas os usuários, nem representam os cargos que as pessoas ocupam nas empresas necessariamente. Exemplos são: aluno, professor.
- **pessoas, locais ou coisas** - cor verde: representam os objetos tangíveis e localidades. Exemplos são: sala, automóvel.
- **descrições** - cor azul: são basicamente as especificações propostas por **DESCO-BRIR**. Modelos de um produto é um bom exemplo.

Os quatro estereótipos coloridos do modelo de Coad, Luca e Lefebvre (1999) ainda trazem um padrão de uso que é bastante interessante: relacionamento entre as pessoas e os intervalos podem ser mediados por um papel da pessoa no intervalo. Assim, um contrato de aluguel é um intervalo, onde uma pessoa é o locatário e outra o locador, ou seja, existe uma entidade pessoa que assume dois possíveis papéis.

A Figura 1.4 mostra um exemplo do uso dessa notação e do padrão descrito.

J. Robertson e S. Robertson (1998) sugerem algumas regras para que verifiquemos se um conceito deve ser realmente escolhido como uma entidade:

- toda entidade deve ter um papel único e definido no negócio, se você não pode explicá-la, provavelmente não precisa se lembrar dela;
- entidades devem ter ao menos um atributo que as descrevam, e é preferível que tenham vários;
- entidades devem ter mais de uma instância. Se a instância é única, então não deve ser uma entidade, mas uma informação constante, que é parte do negócio da empresa (uma regra de negócio?);
- entidades devem possuir instâncias unicamente identificáveis;
- entidades não possuem valores, apenas atributos possuem valores;
- pessoas e organizações que interagem com o sistema são candidatos a entidade quando precisamos nos lembrar alguma coisa específica sobre elas, para gerar relatórios ou processar dados entrados. Isso não se aplica a “logons” ou “passwords” utilizados para a segurança do sistema, pois segurança é um problema tratado no projeto físico. Devemos aplicar essa regra em relação à necessidade de identificação e endereçamento, por exemplo;

## 1. Modelo de Entidades e Relacionamentos

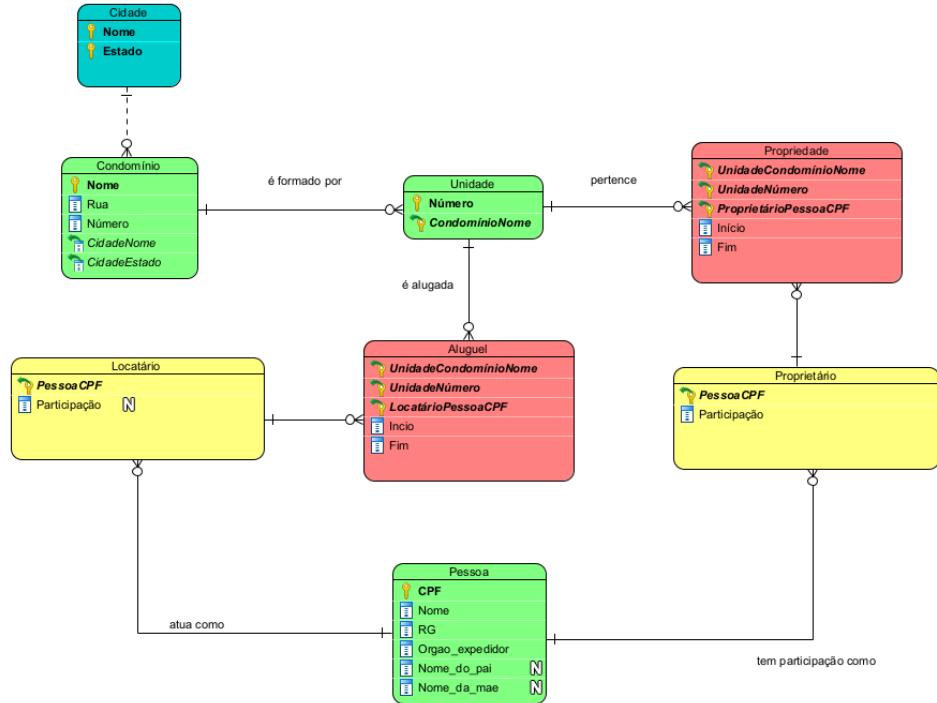


Figura 1.4.: Exemplo de modelo de entidades e relacionamento usando cores de acordo com o proposto por Coad, Luca e Lefebvre (1999)

- relatórios raramente são entidades. Normalmente eles são apenas os resultados de um processo que acessa várias entidades;
- linhas de relatório geralmente são entidades;
- nomes de colunas indicam entidades ou seus atributos;
- nenhum valor calculado ou derivado é atributo ou entidade;
- substantivos em regras de negócio são normalmente entidades;
- produtos, quando não são únicos, são normalmente entidades;
- papéis, como funcionário, atendente, apostador, etc., são bons candidatos para entidade, e
- um grupo de dados que se repete em uma entrada ou saída de dados é normalmente uma entidade (ou mais).

### 1.5.1. Onde encontrá-las

Além de encontrá-las em entrevistas e em regras de negócio, podemos utilizar alguns documentos para encontrar entidades:

- relatórios;
- formulários de entrada de dados;
- arquivos, tanto de papel quanto no computador;

- fichas, como fichas de cadastro, de empréstimo, etc;
- pedidos, requisições e documentos do gênero;
- documentos contábeis e fiscais, como nota fiscal;
- planilhas de dados, em papel ou eletrônicas;
- listagens, registros, agendas, protocolos e outros documentos de trabalho;
- sistemas já existentes,e
- bancos de dados já existentes.

Outra forma de encontrar entidades é buscar sistemas semelhantes já resolvidos e padrões de projeto ou padrões internacionais sobre o assunto sendo tratado.

### 1.5.2. Descrevendo Entidades

É extremamente importante a descrição precisa de cada entidade , pois sua descrição não serve só de documentação, mas também de teste para verificar se entendemos realmente sua presença no sistema. Uma boa descrição de entidade conter os seguintes itens:

- nome, incluindo uma listagem de sinônimos e homônimos;
- definição;
- exemplos;
- atributos (veremos a seguir);
- relacionamentos (veremos a seguir);
- eventos que a utilizam (veremos no próximo capítulo);
- correlação, descrevendo outras partes da análise que se referem a ela;
- regras e exceções relacionadas a essa entidade, incluindo regras de negócio;
- outros comentários e observações, e
- uma idéia da quantidade esperada de instâncias no sistema.

Durante a definição devemos tentar responder várias perguntas, procurando deixar claro o porquê dessa entidade fazer parte do sistema. Assim devemos nos preocupar em dizer o que é essa entidade, o que faz e para que está no sistema, quando algo é ou não é uma dessas entidades, quando passa a ser ou deixa de ser, ou se é permanentemente.

Quando algum elemento passa de uma entidade para outra devemos tomar bastante cuidado para descrever as ações necessárias para tal fato.

## 1.6. Relacionamentos

A principal característica das entidades que compõe um sistema é se relacionarem umas com as outras. É impossível imaginar uma entidade isolada em um sistema de informação. Toda entidade deve possuir ao menos um relacionamento que a coloque em contato com as outras entidades do sistema.

## *1. Modelo de Entidades e Relacionamentos*

Relacionamentos representam que existe alguma conexão entre entidades dentro do negócio sendo analisado [B34]. Cada relacionamento deve ser também uma regra de negócio e é utilizado em pelo menos um processo que lida com as entidades envolvidas.

Relacionamentos indicam a possibilidade de buscar um grupo de entidades a partir de outra entidade. Assim, permite encontrar os “visitantes” que “emprestaram” um “livro” específico (navegando de livro para clientes), ou descobrir que “produtos” um “cliente” “pediu” (navegando de clientes para livros). Indicam também que precisamos nos lembrar de algo que envolve, simultaneamente, duas ou mais entidades do sistema, e que essa lembrança só faz sentido quando todas as instâncias envolvidas são recuperáveis simultaneamente ou seqüencialmente.

Existem muitos relacionamentos comuns, encontrados em muitos sistemas, como “compõe” (peças compõe máquinas), “é um” (bicicleta “é um” meio de transporte), “faz” ou “gera” (cliente faz ou gera pedido), “atende” (visita atende solicitação de reparo), “usa” (cliente “usa” produto), etc.

O relacionamento “é um” é tão comum, e tão útil, que foi escolhido como um relacionamento especial em muitos métodos derivados da Modelagem Entidade e Relacionamento original. É a relação de herança, onde dizemos que uma entidade “herda” todas as características de outra entidade. A herança equivale à abstração de generalização/especificação e será discutida mais adiante.

Outro relacionamento tão comum que mereceu um tratamento especial em muitos métodos é o relacionamento “é parte de”. Esse relacionamento equivale à abstração de composição. É normal que utilizemos esse relacionamento apenas quando a parte só existe em função do todo, porém não é uma exigência muito forte.

Relacionamentos podem unir indiferentemente entidades do mesmo tipo ou entidades de tipos diferentes. Quando relaciona entidades do mesmo tipo, por exemplo, pessoas com pessoas, dizemos que é um auto-relacionamento. Ao especificar um auto-relacionamento devemos ter mais cuidado em declarar os papéis das entidades no relacionamento, além de atentar para não produzir um loop infinito no relacionamento. Para isso, algum lado do relacionamento tem que ser opcional.

Apesar de ser possível trabalhar com relacionamentos múltiplos, isto é, relacionamentos contendo mais de duas entidades, normalmente trabalhamos apenas com relacionamentos entre duas entidades. É comum transformar um relacionamento múltiplo em alguma entidade que o represente.

O mesmo acontece com o uso de atributos no relacionamento. Apesar do método original permitir, atualmente criamos uma entidade para representar esse relacionamento. Bertini, Ceri e Navathe (1992) mostram algumas operações que, aplicadas a um modelo e-r, criam diagramas diferentes que podem representar uma mesma realidade. Assim, algo que foi representado como uma entidade em um modelo pode ser representado como duas em outro, ou um relacionamento em um modelo pode ser transformado em uma entidade que se relaciona com as entidades originais (ou vice-versa) sem que haja uma

representação falsa da realidade. Pode acontecer de uma ou outra representação ser mais interessante em um contexto.

Relacionamentos podem ser condicionais ou incondicionais, isto é, uma entidade pode ser obrigada a ter um relacionamento com outra ou não. Por exemplo: um automóvel é obrigatoriamente fabricado em uma fábrica, mas nem todos os livros em uma livraria já foram vendidos. Como veremos adiante, o fato de um relacionamento ser opcional é representado pela definição da cardinalidade mínima do relacionamento, que pode ser 0 ou 1.

Também é importante notar que existem também relações que ocorrem entre relacionamentos. Dois relacionamentos podem ocorrer sempre juntos (contingentes) ou nunca ocorrer juntos (mutuamente exclusivos). Existem métodos que permitem anotar diretamente no diagrama essas características, porém são pouco utilizados.

Tudo que não puder ser anotado no diagrama deverá ser anotado em um documento associado. O principal tipo de anotações são as regras de negócio que funcionam como restrições, como “um professor só pode dar aulas para alunos da escola em que trabalha”. Restrições são geralmente impossíveis de desenhar diretamente no diagrama .

Normalmente associamos restrições a ciclos no diagrama. Por exemplo, se temos que fazer pedidos de livros para uma editora, então temos um relacionamento entre livro e pedido, um entre livro e editora e um entre editora e pedido, formando um ciclo. A restrição é que “um pedido só pode conter livros da editora indicada no pedido”. É possível desenhar o diagrama sem ciclos, eliminando, por exemplo, a ligação entre pedido e editora, porém aconteceriam duas coisas: primeiro teríamos que escrever uma restrição que é possivelmente mais complexa (“um pedido só pode conter livros da mesma editora”), segundo não teríamos nenhuma indicação no diagrama que o pedido é feito para a editora, exigindo uma nova regra. Finalmente, a falta do ciclo funciona também como falta de indicação que existe uma restrição, pois todo ciclo é um aviso de restrição .

### 1.6.1. O Relacionamento de Herança

Existem duas formas básicas de herança. Na herança exclusiva dividimos uma classe em categorias. Essa forma de herança traz poucas dificuldades na modelagem e é conhecida como separação em categorias. Uma pessoa, por exemplo, pode ser dividida em duas categorias, a dos homens e a das mulheres. Quando a divisão não é exclusiva, ou seja, quando é possível que uma instância de uma entidade específica seja classificada em duas (ou mais) de suas subclasses, temos alguns problemas que devem ser resolvidos na modelagem lógica. Por exemplo, uma pessoa pode ser aluno e professor simultaneamente em uma faculdade.

Também é possível que existam instâncias que não fazem parte de nenhum dos tipos de entidade especializados, mas fazem parte do tipo geral. Isso também exige um tratamento especial durante a modelagem lógica.

## 1. Modelo de Entidades e Relacionamentos

Nós dizemos então que:

- A cobertura é total, se cada elemento da classe genérica é mapeado em ao menos um elemento das subclasses.
- A cobertura é parcial, se existe ao menos um elemento da classe genérica não mapeado nas estruturas das subclasses.
- A cobertura é exclusiva, se cada elemento da superclasse é mapeado em no máximo um elemento das subclasses.
- A cobertura é sobreposta, se existe um elemento da superclasse mapeado em mais de um elemento das subclasses.

Devemos tentar obter apenas heranças totais e exclusivas, pois são mais fáceis de serem tratadas. Dado um grupo de entidades candidatas a construir um relacionamento de herança, devemos analisar se existe um atributo ou relacionamento que é aplicável a apenas um subconjunto dessas entidades, se simplificamos o modelo e se aumentamos sua compreensão. Ou seja, devemos usar a herança para aumentar a semântica do modelo sem causar excesso de informação.

### 1.6.2. Cardinalidade

Para bem representar um relacionamento, devemos indicar a cardinalidade desse relacionamento, isto é, quantas vezes uma instância da entidade pode se relacionar com instâncias de outras entidades.

Veja por exemplo o relacionamento “mãe-filha”. Uma filha só pode ter uma mãe, mas uma mãe pode ter várias filhas. Existem três tipos básicos de relacionamentos: o 1:1, um para um, o 1:N, um para muitos, e o N:M, muitos para muitos. Nesse caso só estamos falando da cardinalidade máxima. A cardinalidade máxima indica quantas vezes uma entidade pode aparecer em um relacionamento.

No relacionamento 1:1 cada entidade só pode se relacionar com uma entidade do outro conjunto. Geralmente indica semelhança, igualdade, utilização conjunta, etc.

No relacionamento 1:N cada entidade de um conjunto pode ser relacionar com várias entidades do outro conjunto, mas as entidades do segundo conjunto só podem se relacionar com uma entidade do primeiro conjunto. Geralmente indicam relações de posse, hierarquia ou de composição.

No relacionamento N:M qualquer número de relacionamentos é válido. Podem indicar várias coisas, como eventos, contratos, acordos, ligações temporárias como empréstimos e aluguéis, etc. É normal aparecerem também quando o relacionamento é do tipo 1:1 ou 1:N em certo momento ou período (como o aluguel de uma fita de vídeo), mas se deseja manter a história de todos os relacionamentos. Quando falamos também da cardinalidade mínima usamos notação de par ordenado, (0,1):(1,N) por exemplo, onde o primeiro número do par indica a cardinalidade mínima e o segundo a máxima. A cardinalidade mínima indica uma exigência da participação de uma instância da entidade

em relacionamentos. A cardinalidade mínima 0 em ambos os lados indica a existência própria de ambos os objetos. A cardinalidade mínima 1 pode indicar a necessidade de um objeto pertencer ou ser criado por outro.

É comum evitarmos, na prática, relacionamentos onde ambos os lados exijam como cardinalidade mínima “1”. O motivo é que um par de entidades só pode ser colocado na memória do sistema em uma mesma transação, não permitindo que primeiro coloquemos a instância de uma entidade na memória e depois uma instância relacionada da outra entidade. Obviamente os SGBDs permitem criar uma transação longa, mas mesmo assim isso cria uma dificuldade adicional.

Temos então os seguintes tipos de relacionamentos:

- Relacionamentos um para um.
  - O relacionamento 1x1 é menos comum em um modelo, já que a restrição que ele aplica a relação entre dois conjuntos é forte: uma instância só pode se relacionar com uma instância apenas.
  - Seus sub-tipos são:
    - ◊ (0,1):(0,1)
      - Esse relacionamento significa que uma instância do primeiro conjunto pode ou não se relacionar com uma instância do segundo conjunto, porém pode ter apenas um relacionamento. O mesmo vale do segundo conjunto para o primeiro.
      - Esse relacionamento é encontrado quando é possível formar pares entre duas entidades, mas esses pares são opcionais.
      - Um exemplo seria o caso da alocação cabines e reboques de caminhões em uma empresa de aluguel de viaturas. Cabines e reboques podem ser trocados arbitrariamente. Em certo momento, cada cabine só pode ter um reboque e vice-versa. Além disso, algumas vezes uma das partes fica guardada na garagem enquanto a outra é utilizada.
      - Outro caso que podem mostrar são auto-relacionamentos desse tipo. Uma igreja ou um templo, por exemplo, pode ter um catálogo de frequentadores e querer saber quem é casado com quem (e quem é solteiro, ou seja, não tem nenhum relacionamento).
    - ◊ (1,1) :(0,1)
      - Esse relacionamento significa que a primeira entidade obrigatoriamente tem um relacionamento, mas ele é opcional para a segunda entidade. Em ambos os casos apenas um relacionamento é permitido.
      - Esse relacionamento é encontrado quando uma entidade possui ou controla de alguma forma outra. Em alguns casos as duas entidades podem ser unidas em uma só.
      - Ele é menos comum que o relacionamento (1,1):(0,N).
      - Um exemplo seria uma distribuição de papéis de uma peça de teatro em uma companhia de atores. Cada ator só pode fazer um papel,

## 1. Modelo de Entidades e Relacionamentos

alguns atores podem não ter papel, mas todos os papéis têm um ator, e apenas um ator.

- ◊ (0,1):(1,1), similar ao anterior
- ◊ (1,1):(1,1)
  - Esse relacionamento é pouco comum, pois indica que uma entidade não pode existir sem estar relacionada com outra, e tudo isso apenas uma vez. Normalmente pode ser substituído pela unificação das duas entidades em uma só.
  - Algumas vezes é utilizado para diferenciar aspectos diferentes da mesma entidade. Por exemplo, um avião é uma entidade que tem visões comerciais, de mecânica, de operação, etc. Fica muito complicado, em um modelo ER, colocar todos os atributos, que chegam a centenas, em uma só entidade, assim podem ser criados relacionamentos (1,1):(1,1) para tratar essa modelagem.
  - Esse relacionamento não é recomendado, pois exige que ambas as entidades sejam sempre criadas juntas.

- Relacionamentos 1 para N

- Os relacionamentos 1xN são o típico relacionamento pai-filhos, ou mão-filhas. Ele aplica uma restrição a Relação entre os conjuntos de entidades: as instâncias de um dos conjuntos só podem aparecer uma vez na relação, como um filho só pode ter um pai.
- Outro uso comum de relacionamentos 1xN é quando queremos guardar a história de relacionamentos 1x1, por exemplo, em nosso caso anterior dos caminhões e reboques, não queremos apenas saber que caminhão está usando que reboque nesse instante, mas toda a história de uso.
- Na prática, esses relacionamentos também são usados nos modelos mais modernos quando devemos guardar dados sobre os relacionamentos NxM, porque não temos mais o losango do modelo original de Peter Chen para guardar as entidades. Então o relacionamento NxM se transforma em dois relacionamentos 1xN para outra entidade, que registra a informação. Muitas vezes essa entidade é do tipo “evento ou intervalo”, como um contrato entre as partes.
- Seus sub-tipos são:
  - ◊ (0,1):(0,N)
    - A primeira entidade pode ou não participar do relacionamento, mas apenas uma vez. A segunda entidade pode ou não participar do relacionamento, e ainda pode fazê-lo várias vezes.
    - Esse é um relacionamento muito comum. Normalmente significa que dois objetos que não possuem nenhum relacionamento de propriedade ou restrição de existência podem ser colocados em uma relação hierárquica.
    - Exemplo: esse tipo de relacionamento pode ser encontrado em locais onde temos um estoque de objetos que são alocados a departamentos da empresa, por exemplo, computadores. Um computador só pode

estar alocado em um departamento ou pode estar no estoque (sem alocação). Um departamento pode ter 0, 1 ou vários computadores alocados para si.

- ◊ (0,N):(0,1), similar ao anterior
- ◊ (0,1):(1,N)
  - Esse relacionamento normalmente também indica uma relação hierárquica.. O primeiro objeto pode opcionalmente pertencer a essa relação, enquanto o segundo objeto obrigatoriamente pertence a relação.
  - Não é muito comum, pois exige que uma instância tenha no mínimo uma “filha”, mas as filhas podem existir de forma independente.
  - Pode ser usado quando algo para existir deve ter ao menos uma parte, mas estas partes tem vida própria, mesmo só podendo ser usadas em um lugar.
  - Isso pode ser encontrado, por exemplo, no relacionamento entre uma venda e os itens (quando únicos) vendidos. Uma loja de carros novos, por exemplo, pode em uma mesma venda negociar vários carros, mas necessariamente a venda contém um carro. Já o carro pode ter sido vendido ou não.
- ◊ (1,N): (0,1), similar ao anterior
- ◊ (1,1):(0,N)
  - Indica uma “maternidade” da segunda entidade em relação à primeira, ou seja, cada instância da primeira entidade é obrigada a possuir uma “mãe”, e apenas uma, que seja instância da segunda entidade.
  - É um dos relacionamentos mais comuns.
  - Pode ser encontrado, por exemplo, na relação entre automóveis de uma empresa e multas recebidas. Cada multa é de apenas um automóvel, mas podem existir automóveis com 0, 1 ou mais multas.
- ◊ (0,N) :(1,1), similar ao anterior
- ◊ (1,1):(1,N)
  - Indica uma “maternidade” da segunda entidade em relação à primeira, ou seja, cada instância da primeira entidade é obrigada a possuir uma “mãe”, e apenas uma, que seja instância da segunda entidade. Além disso, obrigatoriamente a “mãe” deve possuir uma filha.
  - Esse relacionamento apresenta o inconveniente de exigir criar uma entidade “filha” para criar a entidade “mãe”.
  - Pode ser encontrado, por exemplo, em um cadastro de pessoas jurídicas, que devem possuir um endereço, mas podem possuir mais de um.
  - Também é encontrado na modelagem normatizada de objetos que contém listas que obrigatoriamente possuem um item, como uma nota fiscal.
- ◊ (1,N):(1,1) , similar ao anterior

- Relacionamentos N para M

## 1. Modelo de Entidades e Relacionamentos

- Os relacionamentos NxM são a forma mais geral, que menos restrições oferecem a relação entre os dois conjuntos de entidades envolvidos, porém não pode ser implementado diretamente em um SGDB, exigindo a criação de uma tabela onde são anotados os pares de instâncias de entidades envolvidos.
- Muitas ferramentas não desenham esse tipo de relacionamento, por não ser possível sua implementação direta nos SGDBs.
- Seus sub-tipos são:
  - ◊ (0,N):(0,N)
    - Esse relacionamento é muito comum. Representa a forma mais geral de relacionamento, opcional e com todas as possibilidades para ambos os lados.
    - Pode ser encontrado, por exemplo, na relação entre alunos e cursos oferecidos em um semestre em uma universidade. Alguns cursos não recebem inscrição, alguns alunos não fazem inscrição em nenhum curso.
  - ◊ (0,N):(1,N)
    - Semelhante ao (0,N):(0,N). Também muito comum, porém agora exigimos que haja pelo menos um relacionamento na segunda entidade.
    - Pode ser encontrado, por exemplo, na relação entre músicas e CDs onde estão gravadas, para controle de uma discoteca. Uma mesma música pode estar em vários CDs, mas não é possível registrar um CD sem músicas (deve existir pelo menos uma). Porém uma música pode nunca ter sido gravada.
  - ◊ (1,N):(0,N), similar ao anterior
  - ◊ (1,N):(1,N)
    - Aqui temos um relacionamento múltiplo que deve existir pelo menos uma vez.
    - Um exemplo é o relacionamento entre salas de uma empresa e móveis colocados nessa sala.
    - Essa representação muitas vezes é verdadeira, mas é evitada, sendo trocada pelo relacionamento (0,N):(1,N), pois exige que ambas as entidades, quando estão sendo criadas, sejam sempre criadas juntas, ou que existam algumas entidades na base como “semente”.

### 1.6.3. Descrevendo Relacionamentos

Dependendo da notação, relacionamentos podem ser descritos por linhas ligando duas entidades ou por um losango ligado por linhas às entidades. Em ambos os casos é possível anotar os relacionamentos com nomes e com a sua cardinalidade (ver exemplos mais a frente).

O nome escolhido para o relacionamento pode estar na voz ativa (mãe gera filho) ou na voz passiva (filho é gerado por mãe). Algumas notações permitem que se usem os dois

nomes (um por cima e um por baixo da linha de relacionamento). Geralmente se usa o nome que permite a leitura do relacionamento da esquerda para a direta na parte de cima da linha (ou se dá preferência a esse nome quando apenas um pode ser utilizado).

Relacionamentos também devem ser descritos e comentados, sendo importante responder qual sua função no sistema, o que eles representam, como e quando são estabelecidos ou destruídos.

## 1.7. Atributos

Todo atributo descreve de alguma forma a instância da entidade. Alguns atributos são especiais e definem a entidade, mesmo que não de forma unívoca. Esses são os atributos nominativos. Outros atributos permitem definir outro objeto que não é o sendo tratado, são os atributos referenciais. Um exemplo de atributo referencial é “fábrica” para “automóvel”, referenciando a fábrica onde foi construído. É uma opção do analista criar ou não entidades que permitem a substituição de um atributo referencial por um relacionamento.

### 1.7.1. Descrevendo Atributos

Devemos definir as seguintes características:

- Nome
- Descrição
- Domínio (valores válidos, como inteiro, real, string ou uma lista de valores, ou ainda tipos criados pelo projetista).
- Tipos de nulos aceitos Exemplo

Na descrição devemos nos preocupar em explicar qual a finalidade do atributo, como são atribuídos os valores, o que significa cada valor, quem define a escolha do valor, quando, por que e por quem o valor é atribuído ou alterado, etc. Atributos são atualmente denotados no mesmo retângulo da entidade, como mostrado nos exemplos a seguir.

### 1.7.2. Atributos Identificadores (Chaves Candidatas e Chaves Primárias)

Alguns atributos têm o poder de distinguir as ocorrências das entidades, isto é, servem para identificar univocamente uma instância de entidade à instância do mundo real. Definido o valor desse atributo, os outros valores são dependentes e não podem ser escolhidos, mas sim devem possuir um valor exato seguindo a realidade.

Um atributo identificador típico em sistemas financeiros é o CPF de uma pessoa física ou o CNPJ de uma pessoa jurídica. Definido o CNPJ, a empresa, e todos os seus dados,

## 1. Modelo de Entidades e Relacionamentos

estão univocamente definidos (nome fantasia, endereço, etc.) no mundo real, e assim deve seguir o sistema que estamos construindo.

Muitas vezes precisamos de mais de um atributo identificador para realmente identificar uma instância. Dizemos então que a chave primária é composta. Se usarmos apenas um atributo como identificador, então dizemos que a chave primária é simples.

### 1.7.3. Relacionamentos Identificadores

Algumas instâncias são identificadas também, ou até mesmo unicamente, por seus relacionamentos. Uma forma de denotar isso é utilizar uma linha mais grossa no relacionamento ou algum símbolo específico. Alguns autores chamam as entidades que são identificadas por seu relacionamento com outras entidades de “entidades fracas” ou “entidades dependentes”. Atualmente esses nomes são considerados derogatórios para entidades que podem ser muito importantes em um modelo. Os alunos também, muitas vezes, tendem a achar que não devemos modelar “entidades fracas”, conclusão que está absolutamente errada.

#### Chaves Estrangeiras

No modelo conceitual não existe o conceito de chave estrangeira, que é uma característica do modelo relacional. Uma chave estrangeira é uma chave de outra tabela que usamos em uma tabela para indicar o relacionamento. Porém, é comum que as ferramentas de modelagem copiem as chaves estrangeiras automaticamente. Em benefício da prática atual, e em detrimento da pureza teórica, mostramos a seguir algumas possibilidades da notação de relacionamento.

## 1.8. Descrição Gráfica do Modelo

Várias são as notações existentes para o modelo de entidade e relacionamento. Usaremos nesse texto a notação de Martin, também conhecida como Information Engineering, fornecida pelo software Erwin.

Nessa notação não temos um símbolo para relacionamentos, apenas um retângulo para entidades. Os relacionamentos são indicados por linhas. Uma linha cheia indica um relacionamento identificador. Uma linha tracejada indica um relacionamento não identificador. Por isso, não podemos usar relacionamentos com atributos, necessitando de uma nova entidade nesse caso. Também não podemos criar relacionamentos múltiplos, necessitando de criar entidades para representá-los.

Apesar de parecer que temos um modelo menos poderoso, temos na verdade apenas uma sintaxe mais simples, com o mesmo poder de modelagem. Algumas decisões também ficam tomadas automaticamente também. Por exemplo, não precisamos decidir se um

### 1.9. Exemplos de notação da Engenharia de Informação

“objeto com atributo” é um relacionamento ou uma entidade, pois relacionamentos não têm atributos em nosso modelo. Acreditamos que a modelagem segundo as regras do IDEF1X ou da Engenharia de Informação possibilita encontrar mais facilmente um modelo essencial do sistema que as regras tradicionais de Chen ou ainda extensões as mesmas.

A cardinalidade é indicada por três símbolos usados na ponta da linha que indica o relacionamento: uma linha indica 1, um círculo indica 0 (zero), e um “pé-de-galinha” indica n. Dessa forma podemos anotar o mínimo e o máximo da cardinalidade usando dois símbolos em cada ponta.

O nome do relacionamento é colocado acima (à esquerda) da linha que o indica, sendo o nome do relacionamento inverso colocado abaixo (à direita). Normalmente se lê a notação partindo de uma entidade, lendo o nome do relacionamento, lendo a cardinalidade da ponta oposta e finalmente o nome da segunda entidade.

Nessa notação os atributos podem ser colocados dentro da caixa que representa as entidades, como apresentado na próxima seção.

## 1.9. Exemplos de notação da Engenharia de Informação

Vamos descrever um modelo na notação da Engenharia da Informação, também conhecida no Brasil como “pés de galinha”.

Apresentaremos o modelo de uma locadora de vídeo. A locadora trabalha com fitas de vídeo. Cada fita de vídeo contém um filme, porém cada fita deve ser identificada unicamente, pois elas podem ser dubladas ou legendadas. As fitas são emprestadas para clientes em um dia e hora específico. Um cliente pode ficar com várias fitas, ou nenhuma. Uma fita pode estar com apenas um cliente, ou estar na loja e não estar com cliente nenhum. É importante saber para quem cada fita específica foi emprestada, para auditar clientes que estragam fitas, por isso todas as fitas são numeradas com um código único. Os filmes são dirigidos por diretores e contém atores.

Observamos que o cliente é um papel assumido por uma pessoa, a fita de vídeo é um objeto físico, existente, o filme é uma obra de arte que está representada na fita (um conceito), diretor e ator são também papéis assumidos por pessoas dentro da idéia de filme e que um aluguel é um contrato entre o cliente e a locadora.

Atenção para outro detalhe: não existe a entidade locadora, pois este sistema é destinado a uma só locadora. Seria uma entidade única, que claramente é uma constante do sistema. A presença de entidades desse tipo é um erro comum nos modelos feitos por principiantes. Porém, se tivéssemos um software para uma rede de locadoras, seria interessante guardar em que locadora está cada fita, o que exigiria essa entidade.

## 1. Modelo de Entidades e Relacionamentos

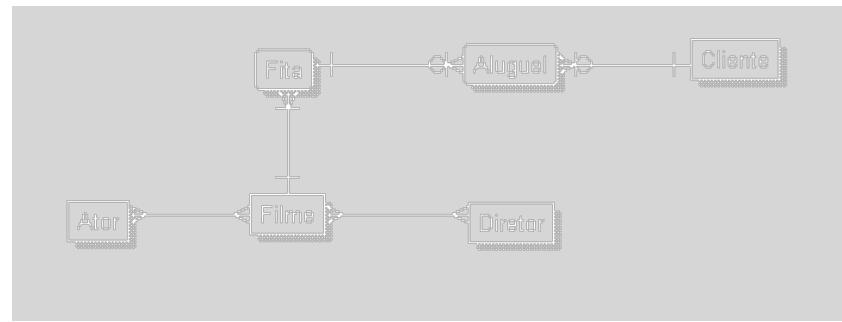


Figura 1.5.: Modelo ER só com entidades e relacionamentos, notação da Engenharia da Informação

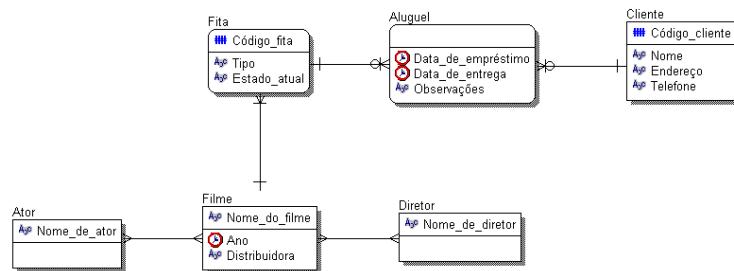


Figura 1.6.: Modelo ER com atributos, notação Engenharia da Informação

## 1.10. Exercícios

**Exercício 1.1:** Escreva um modelo de entidades e relacionamentos para um aplicativo de celular que sirva como relógio e despertador, permitindo vários alarmes.

**Exercício 1.2:** Vá para o site <http://jogodeanalisedesistemas.xexo.net/> e visite a Livraria Resolve. A partir da sua visita faça um modelo de entidades e relacionamentos para o sistema proposto.

## Bancos de Dados Relacionais e SQL

O objetivo deste capítulo é servir de uma pequena introdução ou revisão aos conceitos básicos de bancos de dados relacionais e SQL, que permita o leitor entender o restante do material apresentado no texto. O leitor encontrará cobertura bem mais completa em livros dedicados aos assuntos, como (Elmasri e Navathe, 2016).

Um **Banco de Dados Relacional** é um repositório de dados composto unicamente por tabelas, destinado a registrar as informações que representam o estado de uma sistemas de informação.

Cada tabela é formada por linhas e colunas. As linhas representam fatos de um mesmo tipo sobre o qual se deseja guardar informação, enquanto as colunas representam atributos a serem registrados sobre esses fatos, ou seja, as informações específicas que são guardadas sobre eleElmasri e Navathe (2016). Esses fatos se referem a objetos do mundo real, eventos, momentos, contratos, etc., e são diretamente ligados as instâncias de entidades ou relacionamentos.

Uma célula  $a_{i,j}$  da tabela  $A$  indica o valor do atributo  $j$  para um fato  $i$ . Essa maneira de representar o mundo é conhecida como **Modelo Relacional**.

A Figura 2.1 mostra informalmente uma pequena parte de um banco de dados relacional com três tabelas, criado a partir do modelo ER da 2.2, tratando de avaliações de especialistas sobre o custo de recuperação de terrenos com problemas de contaminação. Cada especialista é descrito por seu nome, em que empresa trabalha e qual sua especialidade. Cada local é descrito por um código, o tipo de terreno e o desenvolvimento da região. Uma terceira tabela indica qual o custo de recuperação de um local de acordo com um especialista. Pela forma como foi construída, cada especialista só pode dar uma avaliação por local, porém todas as combinações de local e especialistas são possíveis, porém apenas algumas são verdadeiras e são registradas na tabela.

Essa é uma forma de explicar o que é um Banco de Dados Relacional em linguagem corrente. Os termos tabelas, linhas e colunas são muito fácil de entender e levam ao usuário a pensar em coisas com que tem o hábito de trabalhar.

## 2. Bancos de Dados Relacionais e SQL

**Especialista**

nome	empresa	especialidade
João	ServAmb	Qualidade de Água
Carlos	EcoConsult	Produtos Químicos
Ana	ServAmb	Vazamentos
...	...	...

**Local**

site_id	terreno	desenvolvimento
L171	Pantanoso	Baixo
M123	Subúrbio	Moderado
...	...	...

**Avaliação**

nome	site_id	custos
João	L171	100K
Paulo	M123	20K
Ana	L171	80K
...	...	...

Figura 2.1.: Exemplo de um banco de dados relacional simples

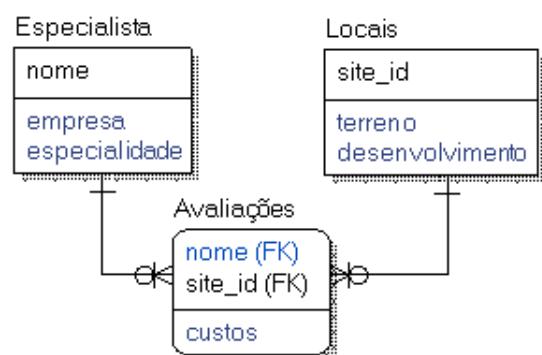


Figura 2.2.: Diagrama de entidades e relacionamentos relativo ao banco de dados da Figura 2.1

Bancos de dados relacionais se assemelham de muita formas com o modelo de entidades e relacionamentos, sendo que a transformação de um MER em um MR é razoavelmente direta. Em linhas gerais, relacionamentos  $N \times M$  devem ser eliminados, sendo transformados em novas entidades e relacionamentos  $1 \times N$ , e então todas as entidades são convertidas em tabelas, enquanto os relacionamentos são indicados por um campo adicional em uma das tabelas. Heuser (2001) apresenta uma descrição de alta qualidade de como fazer essa transformação.

## 2.1. Definição Formal de um BD Relacional

É interessante também possuir uma noção da definição formal de um banco de dados relacional, como a apresentada por Elmasri e Navathe (2016).

O Modelo Relacional usa o conceito de relação matemática como base de sua construção. Ele foi proposto por Codd (1970).

Um **domínio**  $D$  é um conjunto de valores atômicos. Normalmente um domínio é especificado de acordo com um **tipo de dados**, que define todos os valores possíveis dos valores desse domínio.

Exemplos de domínios são: datas, números de telefone, nomes de pessoa, quantidade em dinheiro, etc.

Também é comum que seja especificado um formato para cada domínio. Por exemplo, um formato válido para número de telefone pode ser “+999 999 99999-9999”, indicando três números para o código do país, três para o código de cidade e nove para o número do telefone. Já um domínio relacionado a dinheiro pode ter seu valor máximo limitado e uma indicação de como será representado.

Sendo assim, para definir um domínio são necessários:

- um nome,
- um tipo de dados,
- um formato e
- outras informações adicionais que auxiliem a interpretação do domínio

A definição do domínio é arbitrária, mas segue a lógica de um conceito desejado em um contexto específico. A Tabela 2.1 apresenta a definição de dois domínios diferentes para o mesmo conceito “sexo”, para serem usados em contextos diferentes.

No primeiro caso, o sexo é definido de acordo com o padrão ISO/IEC 5218 (ISO/IEC, 2004). No segundo, segundo as necessidades de um consultório médico.

Um **esquema de relação** é denotado por

$$R(A_1, A_2, \dots, A_n) \tag{2.1}$$

e contém um nome de relação  $R$  e uma lista finita de **atributos**  $A_1, A_2, \dots, A_n$  onde cada atributo  $A_i$  é um papel que algum domínio assume na relação  $R$ .

## 2. Bancos de Dados Relacionais e SQL

Tabela 2.1.: Dois tipos de domínio para o mesmo conceito

	ISO/IEC 5218	Médico
Nome	Sexo	Sexo
Tipo de Dados	{0,1,2,9}	F ou M
Formato	o número em Unicode	a letra em Unicode
Significado	0 - não informado 1 - masculino 2 - feminino 9 - não se aplica	F - Feminino M - masculino

$D$  é chamado o domínio de  $A_i$ ,  $D = \text{dom}(A_i)$ . O **grau da relação** é o número de atributos  $n$ . O esquema da relação descreve uma **relação** chamada  $R$ .

Os esquemas de relação da Figura 2.1 são:

- Especialista(nome, empresa, especialidade)
- Local(site\_id, terreno, desenvolvimento)
- Avaliação(nome, site\_id, custos).

Uma **relação** do esquema de relação  $R(A_1, A_2, \dots, A_n)$ , denotada por  $r(R)$  é um conjunto de  $n$ -tuplas  $r = \{t_1, t_2, \dots, t_k\}$ , onde cada tupla é uma lista ordenada de valores  $t = <v_1, v_2, \dots, v_n>$  onde cada  $v_j$ ,  $1 \leq j \leq n \Rightarrow v_j \in \text{dom}(A_j) \vee v_j = \text{nulo}$ .

Ou seja, na definição formal, um esquema de relação explica como é organizada a relação, que é a tabela, já preenchida com os dados, da definição informal.

Para um mesmo esquema de relação podem existir várias, possivelmente infinitas, relações possíveis. Em um certo instante do tempo, porém, provavelmente apenas uma relação representa corretamente a informação necessária para um sistema de informação.

Como uma relação não é ordenada, então as tuplas de uma relação não são ordenadas.

É importante notar que dentro do Modelo Relacional só existem tabelas. Assim, para consultar a base de dados são feitas operações com tabelas que geram outras tabelas, com o resultado desejado da consulta imaginada. Dentro da teoria foram desenvolvidos tanto uma álgebra quanto um cálculo relacional, que definem operações que sempre geram tabelas como resultado Elmasri e Navathe (2016). A linguagem SQL é uma tentativa de implementar a álgebra relacional por meio de uma linguagem declarativa.

## 2.2. Sistemas Gerenciadores de Bancos de Dados

Bancos de dados relacionais se tornaram ubíquos em sistemas computacionais porque criam uma camada de abstração entre vários sistemas e um conjunto comum de dados para a organização.

Os dados são armazenados, gerenciados e acessados por meio de **Sistemas Gerenciadores de Banco de Dados, SGDB**, que permitem o controle e acesso aos dados por meio de uma linguagem padronizada conhecida como SQL.

Assim, o que a grande maioria das aplicações faz é garantir a visualização adequada e a lógica correta de negócio e consultar e guardar os dados em um SGDB Relacional, por meio de comandos nessa linguagem.

Existem muitos SGDB no mercado, boa parte deles seguindo o modelo relacional. Entre os mais conhecidos e usados estão: MySQL, PostgreSQL, Microsoft SQL Server, e Oracle Database. Os dois primeiros são open-source e possuem licenças para uso gratuito. O Microsoft SQL Server e o Oracle são proprietários, mas tem uma licenças gratuitas para desenvolvimento e uma versão Express para pequenas aplicações, também grátis<sup>1</sup>.

## 2.3. SQL

**SQL**<sup>2</sup>, sigla que significa **Structured Query Language**, é uma linguagem de consulta e manipulação de banco de dados onde o programador explicita o que ele deseja e não como ele deseja que a operação seja feita. Dessa forma, SQL é uma linguagem declarativa e padronizada, apesar de cada vendedor possuir algumas extensões.

A linguagem pode ser dividida em Várias partes. A maioria dos autores registra duas partes principais: **Data Manipulation Language ( DML)**, **Data Definition Language(DDL)**.

Outros autores ou manuais de sistemas específicos apresentam ainda variações com os seguintes subconjuntos: **Data Query Language (DQL)**, **Data Transaction Language (DTL)** e **Data Control Language(DCL)**. Este texto usa a divisão mais detalhada para melhor explicar o espectro total de SQL.

A seguir será feita uma breve introdução a algumas instruções da linguagem SQL, com mais foco na instrução SELECT. Para maiores informações devem ser procurados livros textos, como (Elmasri e Navathe, 2016) e sites e manuais das ferramentas específicas. Nesta introdução é usada a linguagem do sistema MySQL(Oracle Corporation, 2019b), devido a seu largo uso na criação de web sites e na academia.

### 2.3.1. Data Definition Language

A função da **Data Definition Language** é permitir definir e alterar a estrutura do banco de dados, isto é, criar, alterar e modificar tabelas e outros objetos do banco de dados. Deste modo, a DDL trata dos esquemas de relação, ou seja, das estruturas das tabelas.

---

<sup>1</sup>Informações válidas em novembro de 2019

<sup>2</sup>Deve ser pronunciada S-Q-L, letra a letra, em português ou como a palavra *sequel* em inglês.

## 2. Bancos de Dados Relacionais e SQL

Esse subconjunto de SQL é formado pelos comandos CREATE, ALTER, DROP, RENAME, TRUNCATE.

Neste texto serão tratados as instruções CREATE, DROP que criam e apagam tabela e outros elementos de um banco de dados.

### A instrução CREATE TABLE

A instrução **CREATE TABLE** é usada para criar uma tabela. Sua sintaxe permite definir o nome da tabela, seus campos, os tipos de dados dos campos, chaves e índices.

O formato mais simples da instrução é o seguinte:

```
CREATE TABLE nome_da_tabela (
    nome_da_coluna_1 tipo_de_dados PRIMARY KEY,
    nome_da_coluna_2 tipo_de_dados,
    nome_da_coluna_3 tipo_de_dados,
    ...
);
```

Para criar as tabelas da Figura 2.1 seriam necessários os comandos:

```
CREATE TABLE Especialista
(
    nome           CHAR(255) PRIMARY KEY,
    empresa        CHAR(255),
    especialidade CHAR(255)
);

CREATE TABLE Locais
(
    site_id        VARCHAR(20) PRIMARY KEY,
    terreno         VARCHAR(20),
    desenvolvimento VARCHAR(20)
);

CREATE TABLE Avaliacoes
(
    nome           CHAR(255) PRIMARY KEY,
    site_id        VARCHAR(20) PRIMARY KEY,
    custos          DECIMAL(19,4),
);
```

A instrução CREATE TABLE fornece variações poderosas que normalmente podem ser encontradas no manual de cada ferramenta. Além disso, para otimizar a base seria necessário criar restrições adicionais e índices. Sua sintaxe descrita no manual

do MySQL(Oracle Corporation, 2019b) contém 147 linhas e ainda se refere a outras listagens.

Outras instruções do tipo CREATE existem em SQL, mas não serão tratadas neste texto.

## A instrução DROP TABLE

A instrução **DROP TABLE** simplesmente apaga uma tabela do sistema, tanto os dados como sua estrutura. Sua forma mais comum é: “DROP TABLE NOME\_DA\_TABELA”. Um exemplo que destruiria a tabela “Local” é:

```
|DROP TABLE Local;
```

### 2.3.2. Data Query Language

A **Data Query Language** tem como finalidade fornecer funções de consulta a base de dados.

Esse subconjunto de SQL é formado pelos comandos SHOW e SELECT, que é o comando de consulta as informação no SGDB.

O comando SELECT é um dos mais poderosos de toda a linguagem. Sua forma mais simples retorna uma tabela inteira. Por exemplo, todos as linhas da tabela “Local” seriam retornadas com a instrução:

```
|SELECT * FROM Local;
```

Se apenas alguns campos forem necessários, então a consulta poderia ser:

```
|SELECT site_id, terreno FROM Local;
```

Porém, o maior poder da instrução vem de algumas capacidades adicionais. A primeira é a seleção de linha por meio de valores, como em:

```
|SELECT site_id, terreno FROM Local WHERE desenvolvimento = "baixo";
```

Mais ainda, é possível listar novas tabelas a partir de outras, realizando operações conhecidas como **junção**, como em:

```
|SELECT Local.site_id, terreno FROM Local , Avaliacao WHERE desenvolvimento = "baixo" and
```

Para deixar clara a complexidade da instrução SELECT, o texto a seguir apresenta a sintaxe do mesmo em MySQL(Oracle Corporation, 2019b).

```
|SELECT
  [ALL | DISTINCT | DISTINCTROW ]
  [HIGH_PRIORITY]
```

## 2. Bancos de Dados Relacionais e SQL

```
[STRAIGHT_JOIN]
[SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
[SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
select_expr [, select_expr ...]
[FROM table_references
    [PARTITION partition_list]
[WHERE where_condition]
[GROUP BY {col_name | expr | position}, ... [WITH ROLLUP]]
[HAVING where_condition]
[WINDOW window_name AS (window_spec)
    [, window_name AS (window_spec)] ...]
[ORDER BY {col_name | expr | position}
    [ASC | DESC], ... [WITH ROLLUP]]
[LIMIT {[offset,] row_count | row_count OFFSET offset}]
[INTO OUTFILE 'file_name'
    [CHARACTER SET charset_name]
    export_options
    | INTO DUMPFILE 'file_name'
    | INTO var_name [, var_name]]
[FOR {UPDATE | SHARE} [OF tbl_name [, tbl_name] ...] [NOWAIT | SKIP LOCKED]
    | LOCK IN SHARE MODE]]
```

Essa sintaxe esconde muita complexidade, principalmente nos termos `table_references` e `where_condition`.

O exemplo a seguir mostra uma sentença SELECT contendo outra cláusula SELECT dentro dela, que responde com o nome de todos os especialistas que não fizeram nenhuma avaliação de mais de R\$ 100.000,00 reais.

```
SELECT nome
    FROM Especialista
    WHERE nore NOT IN (SELECT nome
        FROM Avaliacoes
        WHERE custos > 100000);
```

### 2.3.3. Uma estratégia para criar consultas SQL

Tendo em vista a importância da instrução SELECT na criação de consultas SQL para o analista de dados, apresento aqui uma estratégia de 8 passos para criar um consulta:

1. escolher as tabelas (cláusula FROM);
2. juntar as tabelas em uma só (cláusulas JOIN);
3. selecionar campos (nomes dos campos);
4. agrupar (cláusula GROUP BY);
5. ordenar (cláusula ORDER BY);

6. filtrar pelos campos das tabelas (cláusula WHERE);
7. filtrar pelos campos agrupados (cláusula HAVING), e
8. limitar o tamanho da resposta (cláusula LIMIT)

Chamamos a atenção que essa ordem é arbitrária, e a construção de expressões SQL complicadas pode ser feita de maneira iterativa e incremental, de forma que fique claro para o autor que fazem o que ele deseja.

A ideia básica é saber em que tabelas estão os dados necessários, e a partir dessas tabelas construir a consulta necessária.

Por exemplo, vamos supor que está sendo usada o banco de dados da base SAKILA, apresentada no capítulo 3, na figura 3.2.

Vamos supor que a consulta desejada pode ser descrita como: listar o valor total obtido com aluguel para cada filme com rating “R”, caso o valor seja maior que 100 dólares, ordenado por valor total, para os 10 filmes que mais arrecadaram.

## **Escolher as tabelas**

Uma observação do modelo da base indica que será necessária usar as tabelas: film, inventory, rental e payment. Nossa esqueleto de consulta fica sendo: SELECT \* FROM FILM, INVENTORY, RENTAL , PAYMENT. Essa consulta não executa ainda.

## **Juntar as tabelas em uma só**

O segundo passo é juntar as tabelas. Isso pode ser feito com a cláusula WHERE, mas prefirimos usar as junções (JOINS). Quatro tipo de junções são possíveis: INNER, LEFT, RIGHT e FULL<sup>3</sup>.

O INNER JOIN exige que cada linha das tabelas juntadas tenham valores que casam, sendo o *join* mais comum, e não aceitando os nulos. O outros garantem que todos as linhas da tabela da esquerda, da direita e das duas apareçam, mesmo que não haja um casamento perfeito.

No nosso caso queremos juntar as tabelas em uma ordem, criando uma consulta já executável:

```
SELECT * FROM film
INNER JOIN inventory ON film.film_id=inventory.film_id
INNER JOIN rental ON inventory.inventory_id=rental.inventory_id
INNER JOIN payment ON rental.rental_id=payment.rental_id;
```

---

<sup>3</sup>Elas também são conhecidas como LEFT OUTER, RIGHT OUTER e FULL OUTER.

## 2. Bancos de Dados Relacionais e SQL

### Selecionar campos

Vamos ficar apenas com o título do filme e o valor recebido como pagamento, criando a seguinte consulta:

```
SELECT film.title , amount FROM film
INNER JOIN inventory ON film.film_id=inventory.film_id
INNER JOIN rental ON inventory.inventory_id=rental.inventory_id
INNER JOIN payment ON rental.rental_id=payment.rental_id;
```

A seleção de campos permite também a mudança de nomes, como por exemplo em:

```
SELECT film.title as titulo FROM film
```

### Agrupar se necessário

Nós queremos agrupar, por valor total de todas os aluguéis, então nossa consulta passa a ser:

```
SELECT film.title , SUM(amount) as total FROM film
INNER JOIN inventory ON film.film_id=inventory.film_id
INNER JOIN rental ON inventory.inventory_id=rental.inventory_id
INNER JOIN payment ON rental.rental_id=payment.rental_id
GROUP BY film.title;
```

A cláusula GROUP BY pode ser feita com várias colunas. Normalmente as colunas não agrupadas que aparecem na seleção devem ser alteradas para conter uma operação de agregação, como SUM, MAX,etc.

### Ordenar se necessário

Vamos ordenar pelo total, em ordem descendente:

```
SELECT film.title , SUM(amount) as total FROM film
INNER JOIN inventory ON film.film_id=inventory.film_id
INNER JOIN rental ON inventory.inventory_id=rental.inventory_id
INNER JOIN payment ON rental.rental_id=payment.rental_id
GROUP BY film.title
ORDER BY total DESC;
```

A cláusula ORDER BY também permite o uso de várias colunas, e a ordem será usada na ordenação, e pode ser feita de forma descendente (DESC) ou ascendente (ASC)

### Filtrar se necessário

Vamos tratar só de filmes com rating igual a “R”:

```

SELECT film.title , rating , SUM(amount) as total  FROM film
INNER JOIN inventory ON film.film_id=inventory.film_id
INNER JOIN rental ON inventory.inventory_id=rental.inventory_id
INNER JOIN payment ON rental.rental_id=payment.rental_id
WHERE film.rating = "R"
GROUP BY film.title
ORDER BY total DESC;

```

A cláusula WHERE é das mais poderosas do SELECT, permitindo uma longa construção de operações lógicas, sendo também usada para fazer INNER JOINs implicitamente. Devemos tomar cuidado porque campos criados com operações de agregação são filtrados com a cláusula HAVING.

### Filtrar os valores agrupados se necessário

Vamos filtrar para os totais maiores que 100 dólares:

```

SELECT film.title , SUM(amount) as total  FROM film
INNER JOIN inventory ON film.film_id=inventory.film_id
INNER JOIN rental ON inventory.inventory_id=rental.inventory_id
INNER JOIN payment ON rental.rental_id=payment.rental_id
WHERE film.rating = "R"
GROUP BY film.title
HAVING total>100
ORDER BY total DESC;

```

A cláusula HAVING é dedicada a filtrar por campos criados com funções de agregação.

### Limitar o tamanho da saída

Limitando a 10 resultados:

```

SELECT film.title , SUM(amount) as total  FROM film
INNER JOIN inventory ON film.film_id=inventory.film_id
INNER JOIN rental ON inventory.inventory_id=rental.inventory_id
INNER JOIN payment ON rental.rental_id=payment.rental_id
WHERE film.rating = "R"
GROUP BY film.title HAVING total>100
ORDER BY total DESC
LIMIT 10;

```

A cláusula LIMIT é auto explicativa.

## 2. Bancos de Dados Relacionais e SQL

### 2.3.4. Data Manipulation Language

Esse subconjunto de SQL é formado pelos comandos INSERT, UPDATE, DELETE, MERGE, CALL, EXPLAIN PLAN e LOCK TABLE.

Neste texto serão tratados os comandos INSERT, UPDATE e DELETE são responsáveis por colocar, alterar e apagar informações das tabelas de um SGDB.

Essas instruções alteram as tabelas e nunca devem ser usadas por um analista de dados nas tabelas fontes. O ideal é que ele não tenha nem mesmo permissão para fazê-lo, a não ser em tarefas de correção de dados e após todos os testes necessários e um backup de segurança ter sido feito antes da operação.

#### **DELETE**

A instrução delete é a mais simples de definir. Para isso, basta normalmente usar algo como:

```
| DELETE FROM Local WHERE terreno = "Arenoso";
```

Sua sintaxe completa no MySQL é(Oracle Corporation, 2019b):

```
| DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name [[AS] tbl_alias]
|   [PARTITION (partition_name [, partition_name] ...)]
|   [WHERE where_condition]
|   [ORDER BY ...]
|   [LIMIT row_count]
```

Cuidado, porém, pois um DELETE errado pode fazer você perder os dados totalmente, a menos da existência de um backup. Não recomendo apagar nenhum dado nas fontes, na verdade, recomendo que um analista de dados nem tenha a permissão para fazer isso no SGDB, para evitar erros.

#### **INSERT**

O objetivo dessa instrução é inserir dados em uma tabela. Sua forma mais usada é simplesmente inserir uma linha em uma tabela listando os valores de todos os seus campos na ordem correta(Oracle Corporation, 2019b), como em:

```
| INSERT INTO Local VALUES("s123-a","pantanoso","alto");
```

Sua sintaxe, em MySQL(Oracle Corporation, 2019b), permite variações:

```
| INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
|   [INTO] tbl_name
|   [PARTITION (partition_name [, partition_name] ...)]
|   [(col_name [, col_name] ...)]
```

```

{VALUES | VALUE} (value_list) [, (value_list)] ...
[ON DUPLICATE KEY UPDATE assignment_list]

INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE]
[INTO] tbl_name
[PARTITION (partition_name [, partition_name] ...)]
SET assignment_list
[ON DUPLICATE KEY UPDATE assignment_list]

INSERT [LOW_PRIORITY | HIGH_PRIORITY] [IGNORE]
[INTO] tbl_name
[PARTITION (partition_name [, partition_name] ...)]
[(col_name [, col_name] ...)]
SELECT ...
[ON DUPLICATE KEY UPDATE assignment_list]

value:
{expr | DEFAULT}

value_list:
value [, value] ...

assignment:
col_name = value

assignment_list:
assignment [, assignment] ...

```

## UPDATE

A instrução update permite alterar o valor de células específicas de acordo com uma condição. Por exemplo para alterar o valor do custo de descontaminação do site L171 em mais 10%, o seguinte comando seria válido.

```
UPDATE Avaliacoes SET custos=custos*1.1 WHERE site_id="L171";
```

A sintaxe completa da instrução UPDATE para MySQL(Oracle Corporation, 2019b) é:

```

UPDATE [LOW_PRIORITY] [IGNORE] table_reference
SET assignment_list
[WHERE where_condition]
[ORDER BY ...]
[LIMIT row_count]

```

## 2. Bancos de Dados Relacionais e SQL

```
value:  
  {expr | DEFAULT}  
  
assignment:  
  col_name = value  
  
assignment_list:  
  assignment [, assignment]  
  
UPDATE [LOW_PRIORITY] [IGNORE] table_references  
  SET assignment_list  
  [WHERE where_condition]
```

### 2.3.5. Data Transaction Language

Esse subconjunto de SQL é formado pelos comandos BEGIN TRANSACTION, COMMIT E ROLLBACK. Esses comandos permitem que o início, o fim e ainda abortar uma transação composta de várias operações no banco de dados.

### 2.3.6. Data Control Language

Esse subconjunto de SQL é formado pelos comandos GRANT, DENY e REVOKE. Esses comandos controlam os privilégios de acesso às tabelas do SGDB e não são tratados neste texto.

## 2.4. O que mais

Existe muito mais a falar sobre bancos de dados relacionais e SQL. Milhares de textos já foram escritos sobre o assunto. Este capítulo tem como finalidade apenas ser uma visão rápida, em especial para o leitor que já viu o tema e pode ter se esquecido de algumas coisas.

Havendo mais interesse, os livros *Fundamentals of Database Systems* de Elmasri e Navathe (2016), *An introduction to database systems* de Date (2004) e *Joe Celko's SQL for smarties: advanced SQL programming* Celko (2005) são boas referências. Uma boa introdução a SQL pode ser encontrada também no site W3School (<https://www.w3schools.com/sql>). Há muito mais material de boa qualidade disponível em livros, artigos, tutoriais e sites.

# 3

## Bases Exemplo

Neste capítulo vamos mostrar dois bancos de dados de exemplo, descritos com o Modelo de Entidades e Relacionamentos. A primeira é a base Sakila e a segunda a NorthWind.

### 3.1. O Bando de Dados Sakila

O modelo de dados da Figura 3.1, escrito na notação de Engenharia de Informação, descreve o banco de dados conhecido como **Sakila**(Oracle Corporation, 2019c). Esse banco é disponibilizado pela Oracle como um banco de dados exemplo de MySQL, licenciado com uma licença New BSD(Oracle Corporation, 2019a).

O modelo descreve uma empresa que possui várias lojas que alugam filmes, sem discutir o formato dos mesmos (DVDs?). Apesar de ser um tipo de negócio em decadência, ainda serve para o nosso propósito.

A informação central do modelo, que aparece na Figura 3.2 está na tabela que registra os aluguéis (*rental*) e na que registra os pagamentos (*payment*).

Essas tabelas se referem a basicamente três áreas: os clientes (*customer*), que alugam (*rental*) itens de estoque (*inventory*), sendo atendidos por um funcionário (*staff*). Esse itens são cópias de filmes (*film*) e os funcionários trabalham em lojas (*store*).

Os pagamentos são feitos pelos clientes aos funcionários para pagar um aluguel.

Clientes, funcionários e lojas possuem endereço (*address*), que é uma tabela única para todos. Endereços estão em cidades (*city*) que estão, por sua vez, em países (*country*).

Todos os funcionários estão alocados em alguma loja, e alguns funcionários são gerentes de uma ou mais lojas.

Os filmes são classificados (*film\_category*) em categorias (*category*), e também organizados (*film\_actor*) pelos atores (*actor*) que trabalham nele. Todo filme também possui

### 3. Bases Exemplo

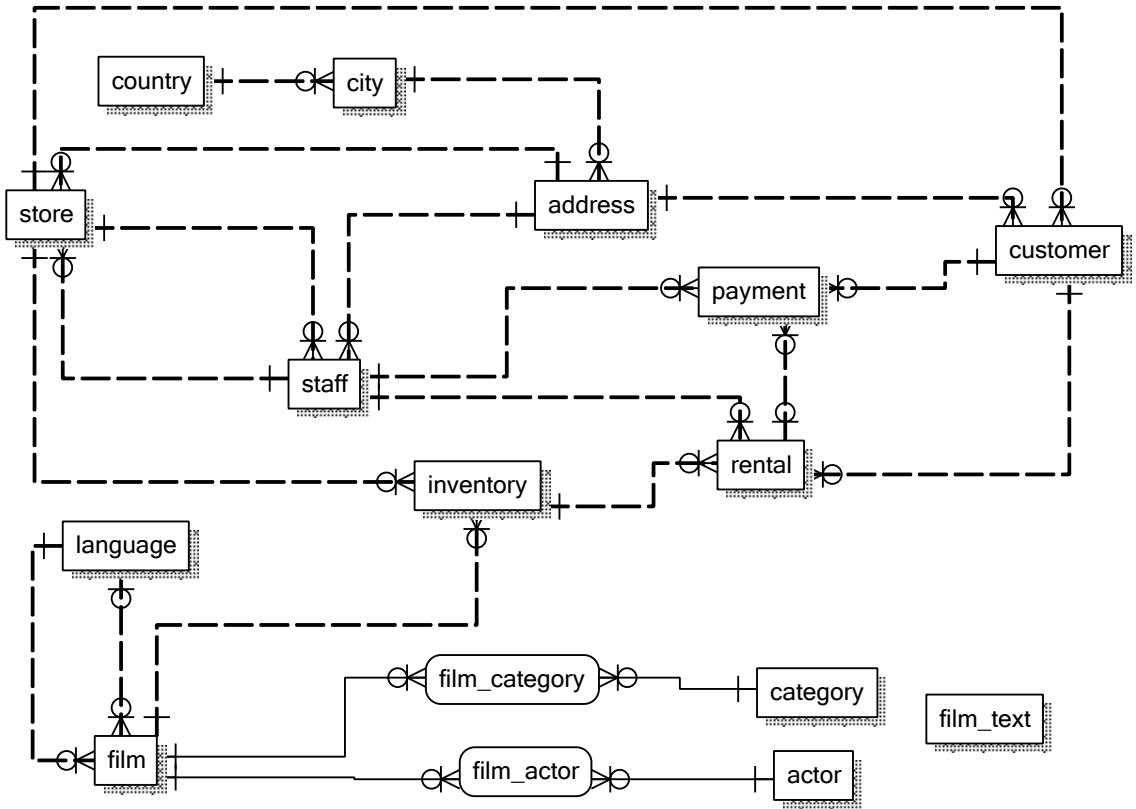


Figura 3.1.: O modelo de dados da base sakila

uma pela linguagem (*language*) original e linguagem disponível o que implica em dois relacionamentos distintos.

A Figura 3.2 apresenta o modelo mais detalhado, com os nomes dos campos, onde as chaves estrangeiras permitem entender como os relacionamentos foram implementados.

Com os nomes dos campos, e mesmo sem os tipos, podemos ver que a principal informação de faturamento do negócio está na tabela *payment* e se refere ao valor (*amount*) do aluguel.

É importante ver que nessa base, toda criação ou atualização de uma linha de tabela é anotada no campo *last\_update*, que aparece em todas as tabelas.

Também existe uma tabela extra que repete alguns dados dos filmes (*film\_text*).

A base Sakila contém apenas duas lojas e dois funcionários. Também possui 599 clientes, 16005 aluguéis, 16086 pagamentos, 1000 filmes, 4581 itens de inventário, 10 categorias, 603 endereços e 200 atores cadastrados.

Finalmente a Figura 3.3 mostra o modelo de dados com seus campos e os tipos dos campos. Vemos que os valores financeiros são guardados com o tipo *decimal*. As datas

### 3.1. O Bando de Dados Sakila

usam o tipo *datetime*, mas algumas informações, como *last\_update*, que aparece em todas as tabelas usam o tipo *timestamp*.

### 3. Bases Exemplo

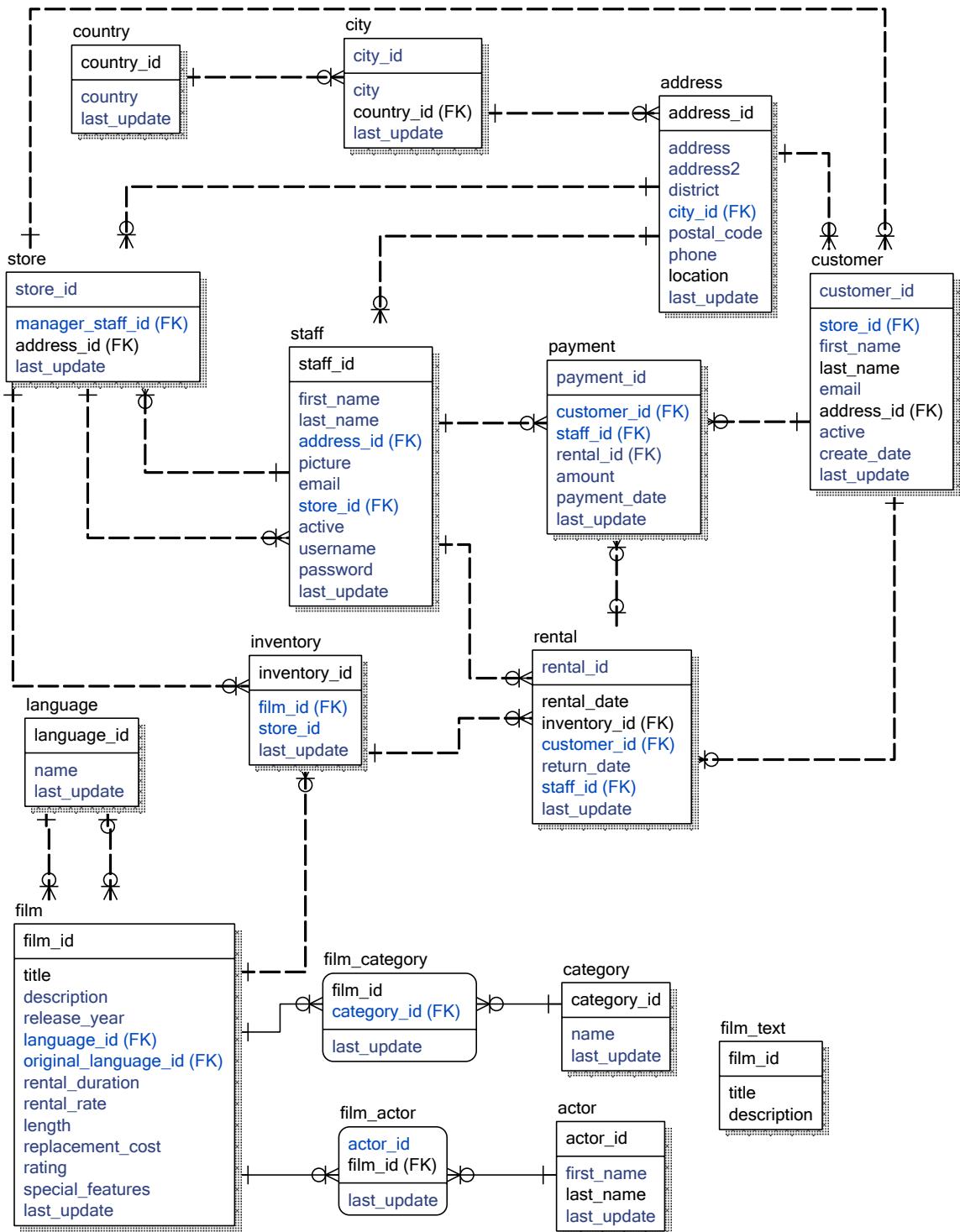


Figura 3.2.: O modelo de dados da base Sakila com os nomes de campo

### 3.1. O Bando de Dados Sakila

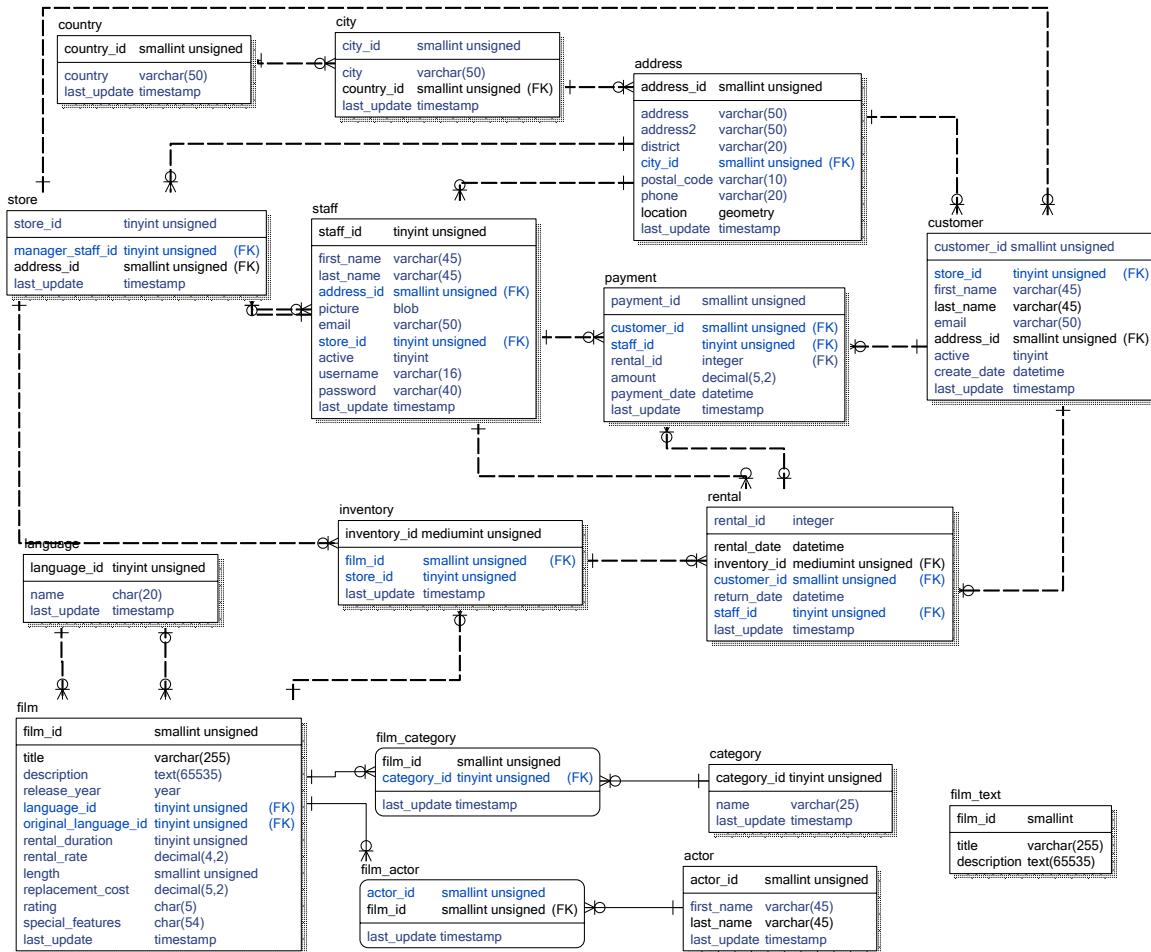


Figura 3.3.: O modelo de dados da base Sakila com nomes de campo e seus tipos

### 3. Bases Exemplo

## 3.2. O Banco de Dados Northwind

O banco de dados Northwind é um banco de dados fornecido pela Microsoft e que simula informações de um ERP de uma empresadir . A versão usada é convertida para MySQL por Scott (2016), e ainda existe em outros formatos, como Postgres. É um banco ainda pequeno e fácil de entender, pois tem poucas tabelas.

A Figura 3.4 mostra o modelo de dados da base Northwind só com suas entidades, na notação da Engenharia da Informação.

Nesse modelo clientes (*customers*) fazem pedidos (*orders*). Os pedidos tem um estado (*order\_status*), são enviados por um transportador (*shippers*), tem um estado do imposto (*orders\_tax\_status*) e tem uma fatura (*invoices*) correspondente.

Os pedidos são compostos de itens de pedidos, que descrevem (*order\_details*) de produtos (*products*), que possuem um estado próprio (*order\_details\_status*).

Os produtos estão em um estoque, do qual entram e saem por meio de transações (*inventory\_transactions*) de certos tipos (*inventory\_transaction\_types*). Eles são comprados por meio de ordens de compras (*purchase\_orders*), que também estão descritas por seus detalhes (*purchase\_order\_details*) e tem um estado (*purchase\_order\_status*). Ordens de compra são feitas para fornecedores (*suppliers*).

Quem faz pedidos e ordens de compra são os funcionários (*employees*), que possuem privilégios (*employee\_privileges*) de certos tipos (*privileges*).

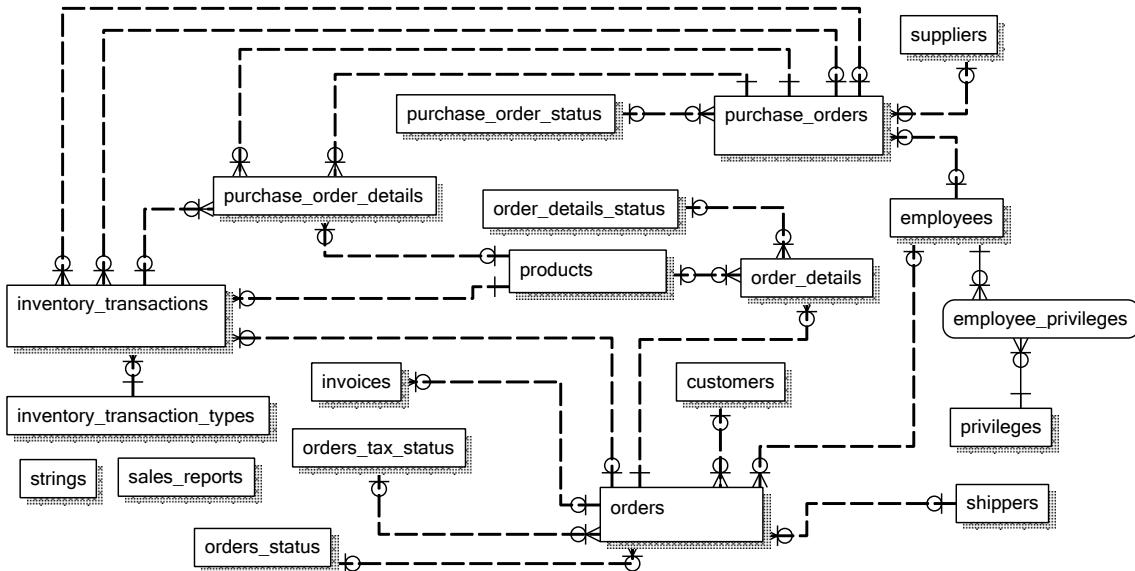


Figura 3.4.: O modelo de dados da base Northwind só com as entidades

A Figura 3.5 já mostra o modelo da mesma base com atributos.

### 3.2. O Banco de Dados Northwind

A base Northwind oferece 45 produtos, 28 ordens de compra, 9 empregados, 48 pedidos e 29 clientes. Sua tabela mais longa é a de transações do estoque, como 102 linhas.

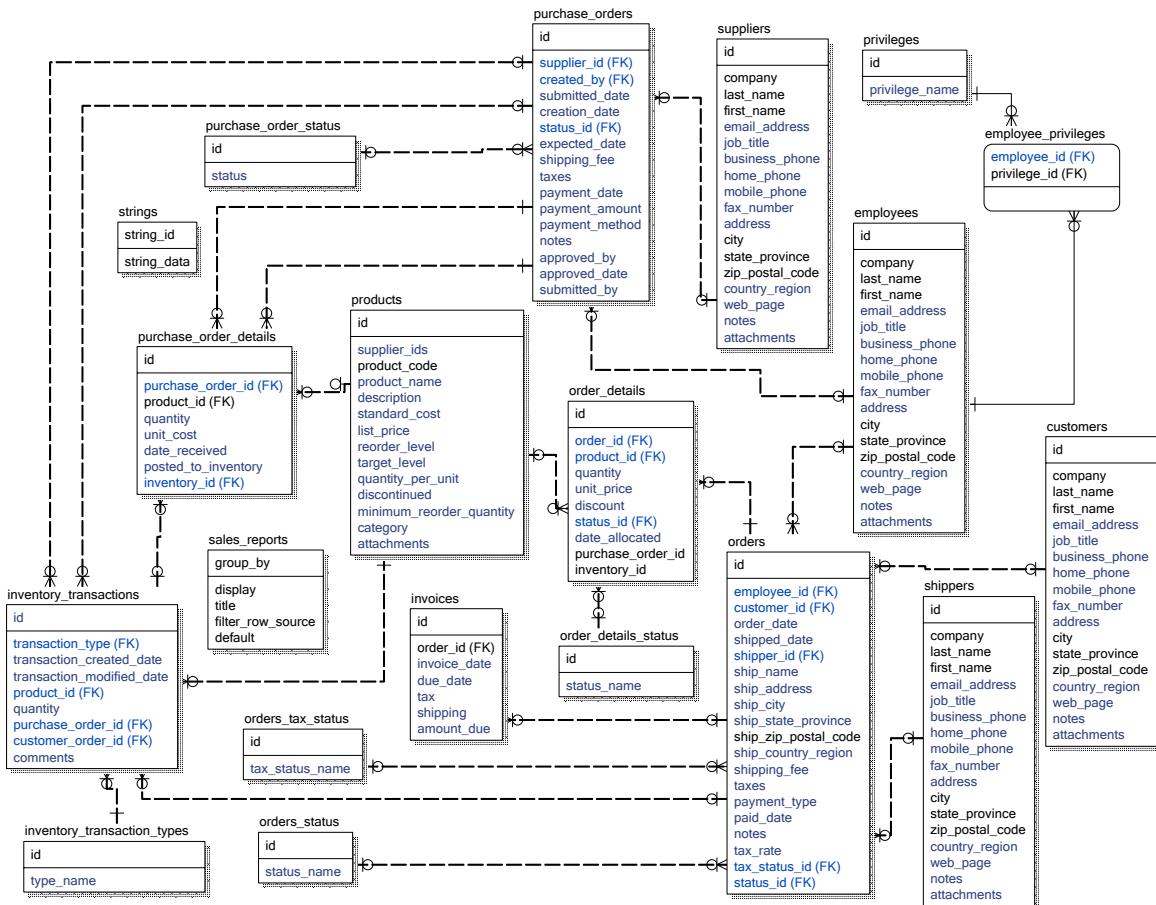


Figura 3.5.: O modelo de dados da base Northwind com nomes de campo

Já a Figura 3.6 mostra não só os atributos como os tipos dos atributos. Aqui é importante notar que os valores financeiro e quantidades estão normalmente descritos como *decimal*.

### 3. Bases Exemplo

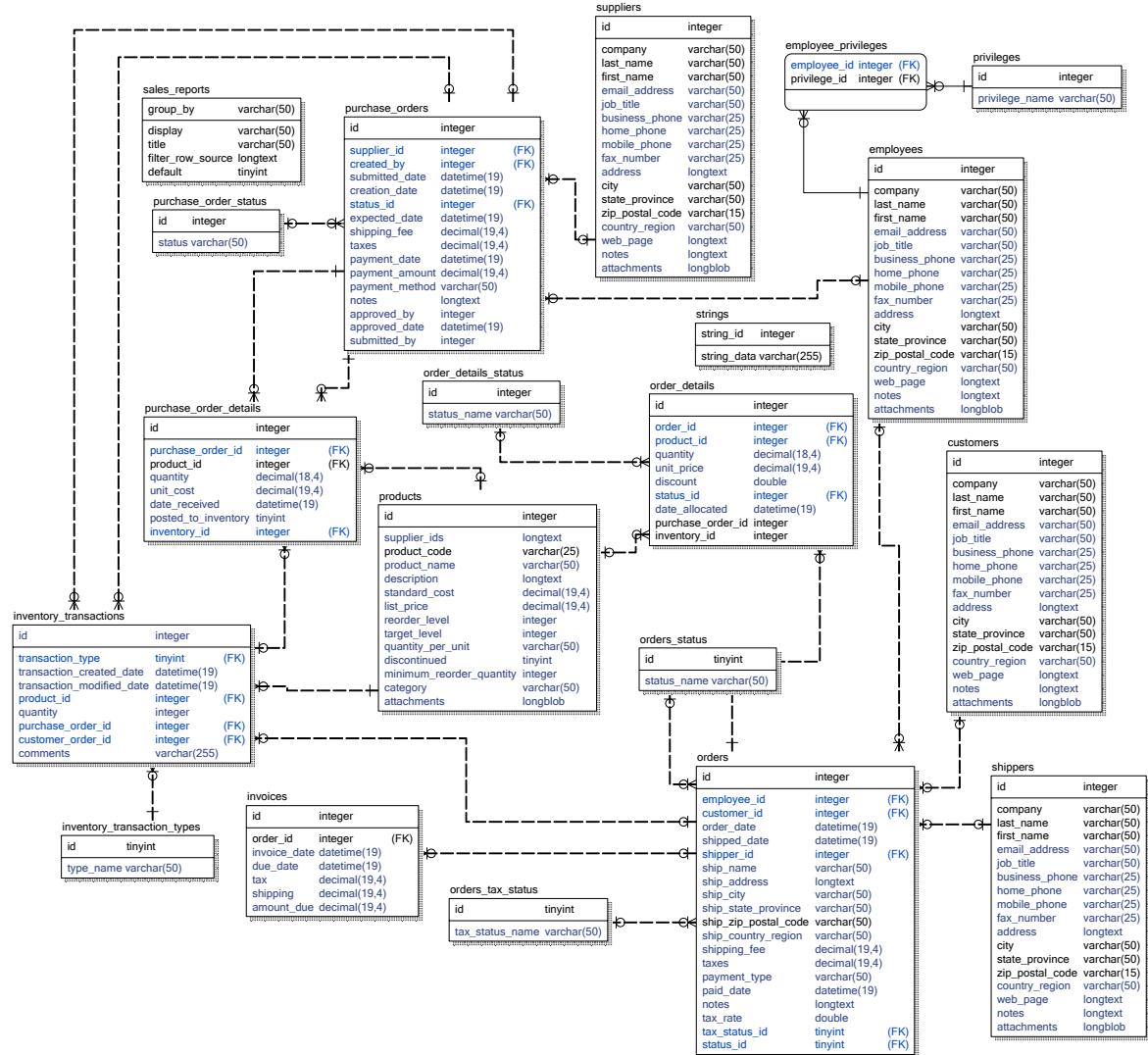


Figura 3.6.: O modelo de dados da base Northwind com nomes de campo e seus tipos

### 3.3. A Cadeia de Valor da Northwind

O processo Pedir Produto, na Figura 3.7, possui 4 passos. Primeiro é necessário que alguém submeta uma necessidade de compra. A seguir a ordem de compra é criada. Criada, a ordem de compra fica esperando a aprovação pelo Vice-Presidente de Vendas. Depois dessa aprovação ela pode ser enviada para o fornecedor.

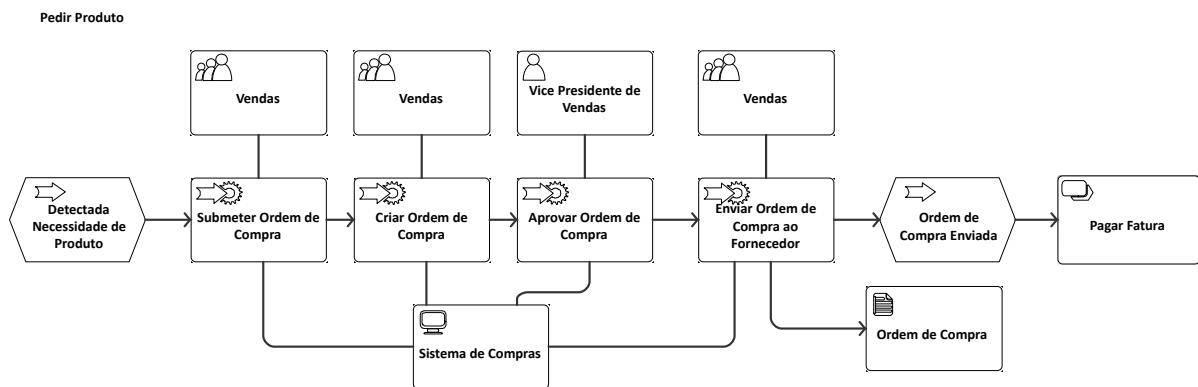


Figura 3.7.: O processo Pedir Produto da Northwind

### 3.4. O Banco de Dados Adventure Works

O banco de dados Adventure Works 2014 (OLTP version) é “Uma base de dados exemplo para o Microsoft SQL Server, que substituiu a base exemplo Northwind, que foi lançada anteriormente. O banco de dados é sobre uma fabricante multinacional de bicicletas, fictícia, chamada *Adventure Works Cycles*.(Motl, 2019)

Esse banco contém bastante entradas e serve para alguns testes mais práticos e capacidade de Data Warehousing.



## **Parte II.**

# **Conceitos de Sistemas de Informação**



O objetivo de qualquer projeto é **entregar valor às partes interessadas**. O conceito de valor, porém, tem várias acepções. Mesmo que muitos métodos e autores se baseiem neste objetivo, raramente valor é definido. Assim, fornecedores e clientes se perguntam: o que é valor? Como posso medir valor?

A resposta pode depender da escolha de uma visão comum do significado de valor que depende de todas as partes interessadas do sistema, logo, é influenciada também por todo o contexto onde o projeto está inserido, incluindo o *background* das partes interessadas.

Este capítulo busca mostrar as várias facetas da ideia de valor. Começa com uma abordagem genérica e depois mostra alguns entendimentos da palavra valor na economia, no marketing e na área de software. Obviamente há alguma similaridade entre as definições, mas os diferentes interesses entre as áreas também fornecem um conhecimento mais amplo do que é valor e de como as pessoas entendem o que é valor.

## 3.5. Conceituação Genérica de Valor

A palavra **valor** pode ser usada em vários sentidos (Cairncross, 1951, pg. 140):

- o **valor moral**, por exemplo, consideramos um *valor* a liberdade, a honestidade;
- o **valor estético**, por exemplo, damos *valor* a obra de Camões;
- o **valor em uso**, ou seja, ligado a utilidade, a como uma coisa atende as necessidades de alguém, como o valor da água, que é alto para a existência de vida, mesmo ela sendo barata;
- o **valor de troca**, por exemplo, o preço de uma casa a ser vendida, e
- o **valor ideal de troca**, o preço que um comprador imaginário pagaria pela casa.

Procuramos entender a acepção de valor com relação aos assuntos ligados aos negócios, em busca de entender **o que é valor para uma organização que está adquirindo ou desenvolvendo um software**. Isto não significa, porém, que a presença de atributos éticos e estéticos em um software não agregue valor, justamente o contrário. É possível, inclusive, que uma organização procure um produto que faça as mesmas funções que o produto que ela já possui, mas que seja mais agradável de usar, valor estético, ou garanta algumas propriedades ligadas ao valor ético, como privacidade.

Em todo caso, mesmo que a organização busque originalmente outro tipo de valor, no momento do projeto é mais provável que a interpretação dada ao conceito seja, na prática, ligada a fatores econômicos, financeiros ou legais, de acordo com a situação.

Da forma mais geral, é possível entender valor como **uma qualidade atribuída a algo ou alguém**, mas essa definição é muito ampla. Beleza, por exemplo, também é uma qualidade atribuída. Neste quadro em especial, valor é uma qualidade relacionada com a **utilidade que um produto ou serviço tem para os que o consomem ou usufruem**.

A princípio, valor econômico é “a importância que um indivíduo dá a determinado bem ou serviço, seja para uso pessoal ou organizacional, seja para a troca” (Holanda Ferreira, 1986).

Essa definição do Dicionário Aurélio é interessante para projeto , porque já diz que o valor é ligado a importância dada ao bem ou serviço que está sendo recebido para o uso, ou sendo produzido para a troca.

Nas próximas seções veremos definições de valor nas seguintes áreas:

- Economia;
- Marketing;
- Projetos, e
- Software.

### 3.6. Valor na Economia

As tentativas de definir algo que pode ser chamado de valor na Economia tem, historicamente, também relação com o entendimento do preço dos objetos e serviços.

Na Economia Clássica, Smith, em 1776, primeiro reconhece que valor pode ter dois significados diferentes, um ligado a utilidade de um objeto particular, e outro ao poder de comprar outro bens. O primeiro seria o “valor de uso”, o segundo o “valor de troca”(Smith, 2003, Livro I, Cap. 4, para. 13)

Smith ainda discutiu valor, no sentido de preço, ou o preço relativo entre bens e serviços (Cairncross, 1951), com três abordagens (King e McLure, 2014), sendo que a terceira antecipou a teoria subjetiva de valor que apareceria mais tarde:

- o trabalho incorporado, que seria adequado as sociedades primitivas;
- a soma dos custos de produção, incluindo terra, capital e trabalho, mais adequado ao capitalismo, e
- a quantidade trabalho e o incômodo de adquirir, ou que é pougado e pode ser imposto a outro, que é seu valor de troca (Strathern, 2003).

Podemos dizer que muitas empresas usam abordagens como essa para calcular o preço de seu software: calcular todos os custos de produção e somar um lucro, ou **markup**, sobre esse preço.

Também as organizações ou pessoas que estão pagando por um produto de software podem ser associadas a definição mais evoluída de Adam Smith: elas pagam para evitar algum incômodo.

Já em 1821, Ricardo (1996) propôs que para se saber o valor é necessário saber a utilidade, e que existiria um preço primário e natural, dado pela quantidade comparativa de trabalho necessário para a produção (King e McLure, 2014).

Novamente, a interpretação clássica de valor tem relação com a prática de desenvolvimento de software: utilidade para o usuário, custo de produção para o desenvolvedor.

### 3.6.1. Utilidade

**Utilidade** é o grau satisfação que obtemos do consumo de bens e serviços (Krugman e R. Wells, 2013). Não é possível medir utilidade de forma prática, ou seja, dizer que algo tem 100 ou 2354 de utilidade para alguém. Porém, teoricamente, se discute a **função utilidade**, que é individual. Isso significa que cada pessoa tem uma satisfação diferente consumindo um produto, e não é possível comparar utilidade entre pessoas.

De acordo com o **princípio da diminuição da utilidade marginal**, cada unidade adicional de bens ou serviços adiciona menos a utilidade total do que a unidade prévia. Esse é um princípio genérico. A verdade é que existem produtos que tem utilidade marginal que aumenta, por exemplo, se você precisa de uma quantidade  $X$  para atingir um objetivo mínimo, a utilidade marginal não vai diminuir até você atingir  $X$  unidades, e pode até aumentar se for uma solução pior ter  $X - 1$  unidades (Krugman e R. Wells, 2013). Um exemplo seria construir um muro de tamanho pré-determinado com tijolos, a utilidade dos tijolos não vai diminuir até atingir a quantidade necessária para o muro. Porém se você quer um muro que garanta privacidade, sem especificar a altura, a utilidade dos tijolos vai começar a diminuir quando certo tamanho é atingido, referente a uma privacidade necessária, até que a privacidade máxima seja alcançada e a utilidade marginal seja zero.

Software é um desses casos. Para um software funcionar, é necessária uma funcionalidade mínima que permita que ele comece a funcionar, porém após essa funcionalidade mínima, cada funcionalidade a mais vai seguir esse princípio, até o ponto que o cliente não mais vai querer pagar por uma funcionalidade, porque não vai valer a pena, já que o investimento vai ser maior que o retorno.

Utilidade é uma coisa importante no valor de produtos de software. Basicamente, em todas metodologias que usam o conceito de valor, o que se pode deduzir das explicações dadas é que **o mais útil em um certo contexto tem mais valor e deve ser priorizado**. Em software, utilidade pode ser um sinônimo de valor em alguns projetos.

### 3.6.2. Microeconomia Moderna e o Custo de Oportunidade

O **custo de oportunidade** indica “do que alguém deve desistir para obter o que deseja”(Greenlaw, Shapiro e Taylor, 2017), ou seja, indica a oportunidade perdida de se consumir ou usufruir outra coisa quando se consome ou usufrui de algo. É o “custo da próxima melhor alternativa” ao que está sendo consumido (Greenlaw, Shapiro e Taylor, 2017).

Segundo Krugman e R. Wells (2013), “o verdadeiro custo de algo é o seu custo de oportunidade”. Se alguém deseja um software, qual o verdadeiro custo disso? Isso envolve então o custo de oportunidade de uma segunda opção, por exemplo, não desenvolver o software e arcar com as consequências e investir o dinheiro.

Assim, novamente em software, o custo de oportunidade é um bom significado de valor. Perguntas que podem ser feitas, em caso de discussões de valor, estão ligadas alternativa de não fazer o produto, ou não corrigir um erro. E, no cálculo de um valor financeiro, o preço das outras ofertas que vão ser recebidas pelo cliente.

O conceito de valor na Economia, como tratado aqui, nos permite trabalhar com algumas ideias onde o homem é visto como um ser racional, que faz cálculos e toma a decisão de forma acertada baseada nesses cálculos. Porém o homem também é um animal emocional, e sua visão de valor envolve necessidades e desejos, como veremos na abordagem que o Marketing dá ao conceito de valor.

### 3.7. Valor em Marketing

O estudo de valor em marketing se inicia com o entendimento do que são **necessidades**, ou seja, “os requisitos básicos dos seres humanos”: água, comida, recreação, educação, etc. (Kotler e Keller, 2012). Necessidades são estados onde se sente uma privação (Kotler, Armstrong et al., 2017). Em uma empresa, comprar um software que emite nota fiscal segundo a legislação vigente é uma necessidade.

Necessidades se transformam em **desejos** “quando são direcionadas para objetivos específicos que podem satisfazê-las”(Kotler e Keller, 2012). Desejos “são a forma que as necessidades tomam quando moldadas pela cultura e personalidade individual”(Kotler, Armstrong et al., 2017).

Por exemplo, a fome gera uma necessidade de se alimentar, que pode se configurar como o desejo de um hambúrguer para um americano, ou um prato feito de arroz, feijão, bife e batata frita para um brasileiro.

Da mesma forma, dada uma necessidade de um tipo de software, uma organização pode desejar características específicas adicionais, como ser feita de código aberto.

As necessidades podem ser (Kotler e Keller, 2012):

- **declaradas**, que são ditas pelo cliente, mas não são necessariamente verdade;
- **reais**, que são o que o cliente realmente precisa;
- **não declaradas**, que são o que o cliente espera receber;
- **de algo mais (delight)**, que são o que o cliente gostaria, e
- **secretas**, que são como o cliente deseja ser visto pela sociedade.

Kotler e Keller (2012) ainda lembram que responder apenas a necessidade declarada pode não ser o bastante, e isso é verdade especialmente **em projetos de software, onde muitas vezes o cliente nem mesmo sabe suas necessidades reais**.

Uma **demand**a é um “desejo por um determinado produto, suportado por uma capacidade de pagar”(Kotler e Keller, 2012). Muitos querem lagosta no almoço, mas só podem pagar por algo mais barato.

Em software, a questão financeira é importante. É possível cotar um projeto que descrito em um parágrafo como algo entre 30 mil reais e quatro milhões de reais, dependendo da extensão do produto e da capacidade de operação necessária, como servidores e previsão de milhões de usuários.

Uma demanda pode ter oito estados (Kotler e Keller, 2012) em relação a um, ou mais, produtos ou serviços:

1. **demand negativa**, quando os consumidores não gostam do produto e podem até pagar para evitá-lo;
2. **demand não existente**, quando os consumidores não sabem que o produto existe ou não tem interesse no mesmo;
3. **demand latente**, quando os consumidores tem um necessidade grande que não é satisfeita por nenhum produto existente;
4. **demand declinante**, quando os consumidores passam a comprar menos ou deixam de comprar um produto;
5. **demand irregular**, quando existe uma variação na compra do produto, em pequena escala de tempo (horas) ou grandes (estações do ano, meses);
6. **demand plena**, onde os consumidores estão comprando todos os produtos postos no mercado;
7. **demand excessiva**, onde os consumidores gostariam de comprar mais produtos do que podem ser fornecidos, e
8. **demand indesejada**, onde os consumidores são atraídos por produtos com consequências sociais indesejadas.

Compreendendo essa sequência que começa com a necessidade, passa para o desejo e chega a demanda, é possível então falar de valor.

Para o marketing, o valor é a “a somatória dos benefícios tangíveis e intangíveis proporcionados pelo produto subtraída da somatória dos custos financeiros e emocionais envolvidos na aquisição desse produto”(Kotler e Keller, 2013). O valor é então uma “combinação de qualidade, serviço e preço”(Kotler e Keller, 2013). As percepções de valor “aumentam com a qualidade e o serviço, mas diminuem com o preço”(Kotler e Keller, 2013).

Segundo Drucker (1974) o valor, para um cliente, é a satisfação de um desejo. Um dos objetivos do marketing é aumentar esse valor, gerando satisfação no cliente. A **satisfação** é então um “julgamento comparativo de uma pessoa sobre o desempenho percebido de um produto em relação as expectativas”(Kotler e Keller, 2013).

### 3.7.1. Os Elementos do Valor - B2C

Almquist, Senior e Bloch (2016) identificaram 30 elementos que fornecem valor para **consumidores**, i.e., pessoas, divididos em uma pirâmide<sup>1</sup> de quatro camadas<sup>2</sup>. Essa pirâmide é apresentada a seguir em forma de lista.

- mais alta*
- **Impacto social:** auto-transcendência (ajudar outros ou a sociedade de forma mais ampla).
  - **Mudança de vida:** fornecer esperança, auto-atualização, motivação, herança para as próximas gerações, afiliação/pertencimento.
  - **Emocional:** reduzir ansiedade, recompensa pessoal, nostalgia, design/estética, representação de status ou aspirações, bem estar, valor terapêutico, diversão/entretenimento, aumentar a atração pessoal, fornecer acesso a outros itens de valor.
- mais baixa*
- **Funcional:** poupar tempo, simplificar, fazer dinheiro, reduzir risco, organizar, integrar aspectos diferentes da vida, conectar com outras pessoas, reduzir esforço, evitar problemas, reduzir custo, qualidade, variedade de escolha, apelo sensorial, informar.



Figura 3.8.: Elementos do Valor B2C. Fonte:Almquist, Senior e Bloch, 2016

Essa pirâmide pode ser comparada com a pirâmide de Maslow, apresentada na Figura 3.9, que mostra também as necessidades das pessoas em camadas cada vez mais abstratas. Sua aplicação é apropriada ao analisar valor em software que vai ser vendido para pessoas.

Os elementos podem ter importância diferente por pessoas e por indústria. Por exemplo, os elementos mais importantes para consumidores de *smartphones* são, em

<sup>1</sup>A pirâmide é uma construção usada para passar a ideia que os níveis inferiores são essenciais para os níveis superiores poderem ser alcançados

<sup>2</sup>Uma versão interativa pode ser acessada em <https://media.bain.com/elements-of-value/>, link válido em 9/2/2020

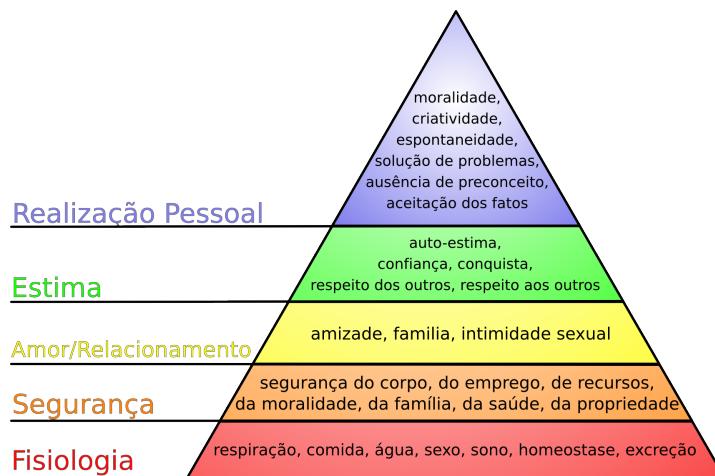


Figura 3.9.: Hierarquia das Necessidades de Maslow. Fonte: Wikipedia Commons por Felipe Sanchez (CC-BY-SA 3.0) e J. Finkelstein (GFDL)

ordem decrescente (Almquist, Senior e Bloch, 2016): qualidade, reduzir esforço, variedade de escolha, organizar e conectar com outras pessoas<sup>3</sup>.

### 3.7.2. Os Elementos do Valor - B2B

Mais tarde, Almquist, Cleghorn e Sherer (2018) desenvolveram modelo semelhante para negócios B2B, obtendo uma pirâmide um pouco mais complicada, com 40 elementos, onde há sub-níveis, representada na Figura 3.10 e na lista a seguir<sup>4</sup>. A importância de cada elemento novamente varia com a indústria. Essa segunda pirâmide é aplicável a software que vai ser vendido a organizações, porém leva em conta que as vendas são feitas para pessoas que participam das organizações. Os compromissos básicos são exigências para o vendedor.

#### mais alta • Valor inspiracional

- objetivo:
  - ◊ visão, ajuda o cliente a antecipar a direção do mercado;
  - ◊ esperança, dá aos clientes e usuários esperanças no futuro da empresa, e
  - ◊ responsabilidade social, ajuda o cliente a ser mais responsável socialmente.

#### • Valor individual:

- carreira:
  - ◊ expansão da rede (*network*), ajuda os usuários e colegas a expandir a rede profissional;
  - ◊ *marketability*, faz clientes e colegas mais *marketables* em seu campo, e

<sup>3</sup>Apesar de ser a missão original do *smartphone*, seu valor como conexão é só o quinto mais valorizado.

<sup>4</sup>Uma versão interativa pode ser acessada em <https://media.bain.com/b2b-eov/index.html>, link válido em 9/2/2020.



Figura 3.10.: Elementos do Valor B2B. Fonte: Almquist, Cleghorn e Sherer, 2018

- ◊ garantia de reputação, não atrapalha e pode aumentar a reputação do clientes no trabalho.
- pessoal:
  - ◊ design e estética, fornece produtos e serviços que são estéticamente agradáveis;
  - ◊ crescimento e desenvolvimento, ajuda usuários e colegas a crescer pessoalmente;
  - ◊ redução da ansiedade, ajuda clientes e outros na organização a se sentir mais seguros, e
  - ◊ diversão e vantagens, é agradável de interagir com ou dá recompensa de alguma forma.
- **Valor da facilidade de fazer negócios:**
  - produtividade:
    - ◊ poupar tempo;
    - ◊ reduzir esforço;
    - ◊ diminuir problemas;
    - ◊ informação, e
    - ◊ transparéncia, fornece uma visão clara da organização do cliente.
  - operacional:
    - ◊ organização;
    - ◊ simplificação, reduz complexidade;
    - ◊ conexão, conecta organização e usuários com outros interna e externamente, e
    - ◊ integração, ajuda o cliente a integrar diversas facetas do negócio.

- acesso:
  - ◊ disponibilidade, garante que o bem ou serviço está disponível quando e onde necessário;
  - ◊ variedade, fornece uma variedade para escolha, e
  - ◊ configurabilidade, permite configurar o bem ou serviço de acordo com as necessidades do cliente.
- relacionamento:
  - ◊ responsividade, responde rápida e profissionalmente as necessidades da organização;
  - ◊ perícia, fornece *know-how* para o mercado ou indústria relevante;
  - ◊ compromisso, mostra que está compromissado com o sucesso do cliente;
  - ◊ estabilidade, é uma empresa(bem ou produto) estável no futuro previsível, e
  - ◊ ajuste cultural, se encaixa na cultura do cliente.
- estratégico:
  - ◊ redução de risco, protegendo o cliente;
  - ◊ alcance, permite o cliente operar em mais locais ou segmentos do mercado;
  - ◊ flexibilidade, vai além dos bens ou padrões comuns para permitir customização, e
  - ◊ qualidade de componentes, melhora a qualidade percebida dos produtos e serviços do cliente.

● **Valor funcional:**

- econômico:
  - ◊ redução de custos, e
  - ◊ aumento de receitas.
- desempenho:
  - ◊ qualidade do produto;
  - ◊ escalabilidade, e
  - ◊ inovação.

mais baixa ● **Compromissos básicos (*table stakes*<sup>5</sup>)**

- satisfazer especificações;
- preço aceitável;
- conformidade regulatória, e
- padrões éticos.

Ambas as pirâmides tentam ser exaustivas, mas foram criadas a partir de pesquisas de campo, o que quer dizer que podem existir outros elementos não detectados, porém isso não é esperado em geral.

Além disso, elas devem ser usadas em função dos elementos mais desejados em uma indústria ou cliente específico. Por exemplo, investigando a contribuição proporcional dos elementos para os clientes de infraestrutura de TI, com uma soma de 100%, (Almquist,

---

<sup>5</sup>No linguajar de negócios em inglês, *table stakes* é o mínimo que você deve fazer para um mercado ou negócio específico.

Cleghorn e Sherer, 2018) encontraram que os três principais elementos são: qualidade do produto (7,8%), perícia (6,1%) e responsividade (5,5%). Poupar tempo, por exemplo, era décimo elemento na ordem e com proporção de apenas 3,0%. Na prática, a importância de cada elemento pode ser levantada com cada cliente.

### 3.7.3. Outras Técnicas

Tendo em vista que a finalidade do Marketing é engajar e gerenciar relações lucrativas com os clientes, é razoável que parte do trabalho de Marketing seja identificar o que é valor para o cliente, de forma que a organização possa atendê-lo com qualidade. Assim, muitas técnicas de Marketing se dedicam a isso e está além deste livro cobrir todas.

Outras técnicas interessantes provenientes do Marketing são o QFD (Franceschini, 2016), adequada tanto a software sob encomenda quanto a produtos a serem lançados no mercado, e a Estratégia do Oceano Azul (Kim e Mauborgne, 2005), adequada apenas a novos produtos a serem lançados no mercado.

## 3.8. Priorização Baseada em Valor para o Cliente

Em muito projetos é mais prático colocar os itens a serem realizados em uma lista priorizada do que dar um valor para cada item, sejam eles demandas do cliente, requisitos em português, casos de uso, histórias do usuário ou outros métodos.

A priorização é um processo para “determinar a importância relativa de informações”(IIBA, 2011). Ela permite várias abordagens:

- agrupamento, normalmente em poucas classes pré-determinadas;
- *ranking* ou ordenação;
- orçamento ou *time-boxing*, por meio da alocação de recursos finitos e
- negociação, em busca de um consenso.

### 3.8.1. Técnicas simples de ordenação

Algumas das técnicas que podem ser usadas para criar uma estimativa relativa de valor com clientes e outras partes interessadas são baseadas em criar uma pontuação ou separar os itens que têm o valor estimado em grupos de níveis de prioridade diferentes(Satpathy et al., 2016).

Métodos simples de priorização incluem:

- esquemas simples de pontuação, onde cada item do projeto analisado recebe um número entre uma variação pequena, como de 1 a 3 ou 1 a 5, ou uma rótulo como “Alto”, “Médio” ou “Baixo”, criando grupos ordenados;

- **dinheiro de brinquedo ou 100-pontos**, onde é dada uma quantidade de dinheiro de brincadeira<sup>6</sup>, como 100 moedas, ou pontos, para o cliente associar aos itens do projeto, algumas vezes submetido a algumas regras, como não ter X itens com o mesmo valor (Leffingwell e Widrig, 1999), criando uma ordem parcial baseada na alocação de recursos, e
- ordenação de cartões contendo um item cada (J. Robertson e S. Robertson, 1998), mas também realizada em planilhas eletrônicas, ou em softwares como *Trello*.

### 3.8.2. Método MoSCoW

Na priorização **MoSCoW**, cada item é associado a um conceito do grupo “*Must Have*”, precisa ter, “*Should Have*”, devia ter, “*Could Have*”, podia ter e “*Won’t Have*”, não vai ter (IIBA, 2011), criando grupos.

Há uma diferença clara entre esse método e separar em grupos que determinam um grau de prioridade: no método MoSCoW os grupos tem um significado específico.

Entre os “*Must Have*” devem estar apenas os itens essenciais ao projeto, ou seja, aqueles que, não existindo, colocam o sucesso do projeto em risco. São fatores críticos do sucesso do projeto.

Nos dois grupos seguintes, “*Should Have*” e “*Could Have*”, cai a necessidade. Espera-se que contenham funcionalidades desejadas, mas que não sejam essenciais ao sucesso. No caso do “*Could Have*”, poderiam ser apenas melhorias eventuais ou mesmo o que é conhecido como *gold plating*, isto é, decoração que não soma realmente valor.

Finalmente os “*Won’t Have*” mostram requisitos que foram levantados, mas não são necessários ou mesmo podem causar problemas ao projeto.

Ao longo do tempo, principalmente em projetos iterativos, como os projetos ágeis, é comum um item mudar de grupo de uma iteração para outra. Por exemplo, quando todos os itens “*Must Have*” estão realizados, pode acontecer de outros itens passar a ser “*Must Have*”. Mesmo itens que são indesejados (“*Won’t Have*”) em algum momento do projeto podem se tornar necessários, e o inverso também é verdadeiro.

Um exemplo de item indesejado que pode se tornar desejado em um projeto é a compra de um insumo ou equipamento muito caro, que passa a se tornar necessário em um certo ponto do projeto. Isso pode acontecer se a alternativa não atende às expectativas ou se o problema se tornou mais difícil do que planejado.

O método MoSCoW não é exclusivo da área de software, mas é parte importante de métodos de gestão de projetos e foi particularmente utilizado no método DSDM(Stapleton, 2003).

---

<sup>6</sup>Facilmente obtida em uma caixa de Banco Imobiliário

### 3.8.3. Análise de Kano

O modelo de Kano é um método para avaliar a reação emocional de clientes a características individuais de um produto. Kano detectou cinco respostas emocionais a essas características (Moorman, 2012):

- encantadores ou atrativos (*delighters*), “eu gosto disso”, que trazem satisfação se presentes, mas não trazem insatisfação se não estiverem presentes, normalmente porque são inesperadas e endereçam necessidades não reconhecidas;
- unidimensionais (*satisfiers*), que causam satisfação se estiverem presentes e insatisfação se não estiverem presentes;
- obrigatórios ou esperados (*dissatisfiers*), que causam insatisfação se não estiverem presentes, mas não causam satisfação se estiverem presentes;
- indiferentes (*indifferent*), ou
- indesejados.

Para classificar em uma dessas cinco respostas emocionais, são feitas duas perguntas ao cliente (Moorman, 2012):

- Como você vai se sentir se a funcionalidade estiver presente?
- Como você vai se sentir se a funcionalidade não estiver presente?

As respostas devem ser em uma escala ordinal:

- eu gosto disso;
- eu espero isso;
- eu sou neutro em relação a isso;
- eu posso tolerar isso, ou
- eu não gosto disso.

Finalmente, a classificação é feita cruzando as respostas na Tabela 3.1.

Tabela 3.1.: Resolvendo a classe de Kano por meio do cruzamento de duas perguntas  
(Moorman, 2012)

		Questão Negativa				
Questão Positiva	Gosto	Gosto	Espero	Neutro	Tolero	Não Gosto
	Gosto	Conflito!	Atrativo	Atrativo	Atrativo	Unidimensional
	Espero	Indesejado	Indiferente	Indiferente	Indiferente	Obrigatório
	Neutro	Indesejado	Indiferente	Indiferente	Indiferente	Obrigatório
	Tolero	Indesejado	Indiferente	Indiferente	Indiferente	Obrigatório
	Não Gosto	Indesejado	Indesejado	Indesejado	Indesejado	Conflito!

As categorias de Kano não são permanentes, e as características de um produto, ou tipo de produto, vão migrando de atrativas, para unidimensionais, e para obrigatórias, e podem até mesmo se tornar indesejadas. O tamanho do celular, por exemplo, sofreu essa mudança com o tempo.

### 3.8. Priorização Baseada em Valor para o Cliente

J. Robertson e S. Robertson (2006) usam o método de Kano no seu método Volare. Nesse método o valor para o cliente é definido em duas perguntas são: quão satisfeito você se sentirá se esse requisito for implementado e quão insatisfeito você se sentirá se esse requisito não for implementado. segundo o casal, os graus de satisfação e insatisfação são a melhor mecanismo para saber que requisitos o cliente considera com maior valor. J. Robertson e S. Robertson, porém, não se referem a Kano, mas a um autor que descreve suas técnicas. As notas dadas também são de 1 a 5.

O modelo de priorização do método Volare permite criar um gráfico de quatro quadrantes, como o da Figura 3.11. O mesmo princípio pode ser aplicado com as respostas do método de Kano.



Figura 3.11.

#### 3.8.4. Prioridade e Consenso

Todas as técnicas explicadas nesta seção foram descritas como se o cliente tivesse uma opinião única, porém isto não é sempre verdade. Como projetos possuem muitas partes interessadas, cada parte interessada pode trazer uma visão diferente da realidade.

Para isso existem técnicas que visam atingir o consenso. A questão é que algumas técnicas atingem um ponto que não é considerado realmente consenso. No sentido estrito, consenso significa que todos concordaram com algo, porém algumas técnicas tentam resolver problemas de opiniões diferentes com votação, sendo possível ser uma votação por maioria simples, ou seja, metade dos votos mais um, ou por maioria qualificada, onde se exigem mais votos, como dois terços dos votos. Outras formas de eleição podem ser mais relacionadas com o conceito de consenso, como o voto em múltiplas rodadas, onde itens menos votados são eliminados em cada rodada. O voto em dois turnos é a versão mais simples desse método. Métodos de votação preferencial tentam simular as

múltiplas rodadas. Nesses métodos o eleitor ordena os candidatos de acordo com sua preferência, eliminando o candidato com menor preferência e recontando os votos, agora sem esse candidato, até que um candidato se eleja por maioria da primeira preferência não eliminada de cada eleitor(Adiel Teixeira de Almeida, 2019)(Amy, 2000).

Algumas técnicas confundem o consenso com o consentimento. Consentir é concordar que a decisão foi tomada da melhor forma possível, mas não necessariamente concordar com ela.

Provavelmente a técnica mais famosa de atingir o consenso é o método de Delphi(Dalkey, 1966). Existem variações desse método com objetivo específico, como o *Planning Poker*(Cohn, 2005, p 56).

Também existem técnicas que permitem calcular o grau de consenso(Meskanen e Nurmi, 2006). Isso pode ser necessário, por exemplo, quando cada pessoa escolhe uma ordenação diferente. Nesse caso, havendo um valor específico que indique o consenso, é possível negociar até que ponto

O dot-voting é uma técnica típica do Design Thinking e dos métodos ágeis. Nessa técnica, os itens a serem priorizados são colocados em cartões em um quadro e cada participante recebe um número de votos, sendo 3 um número padrão na literatura, normalmente na forma de adesivos em forma de círculo. Os participantes então colam seus adesivos, isto é, dão votos não identificados, nos cartões.

A técnica é muito usada, mas alguns defeitos já foram reconhecidos. Um deles é o efeito em cascata, um item já votado tende a chamar atenção e receber mais votos. Outro é que é comum o excesso de escolhas. Além disso, se os itens estiverem em níveis diferentes de uma hierarquia, é possível que um item menos desejado seja escolhido, porque o item mais desejado estava representado em duas possibilidades. Existem soluções, e obviamente estamos falando aqui novamente de soluções ligadas a eleições.

### **3.8.5. Usando mais de uma dimensão**

Os métodos anteriores usavam apenas uma dimensão para determinar a prioridade, porém é possível utilizar outras dimensões. Um exemplo é usar uma dimensão que determine o benefício e outra que determine o custo, já que duas dimensões nos permitem novamente usar um diagrama 2D que pode ser dividido em quadrantes, sendo fácil de analisar. A dimensão que determina o benefício é sempre feita em função das partes interessadas principais, os clientes. Já a dimensão que determina o custo é melhor avaliada pelo equipe que executará o projeto, como a equipe de desenvolvimento. Nesse caso, é necessário utilizar técnicas de predição, como o Planning Poker Cohn, 2004 ou Pontos de Função (IFPUG, 2010).

Definido o benefício e o custo, é possível analisar o resultado em um diagrama com 4 quadrantes, como o da Figura X.

### 3.8. Priorização Baseada em Valor para o Cliente



Figura 3.12.: Gráfico com quadrantes para comparar prioridades usando benefício e custo.

Dada uma tabela com itens, onde tanto o benefício quanto o custo receberam um valor de 1 a 5, os itens podem ser posicionados na gráfico e uma ordem de prioridade pode ser determinada. 3.2. A Tabela 3.2 permite gerar o gráfico da Figura 3.13, onde as setas mostram uma sequência possível de priorização, AFHIEBJG, onde os itens C e D foram desprezados. Outra sequência possível seria AFHIEDJGC. Com essa visualização é possível melhorar a expectativa do cliente quanto a prazos de atendimento das suas reais necessidades.

Tabela 3.2.: Benefício e esforço para itens de um project backlog imaginário.

Funcionalidade	Benefício	Esforço
A	5	1
B	3	2
C	1	3
D	2	4
E	4	3
F	5	2
G	1	2
H	5	3
I	5	4
J	1	1

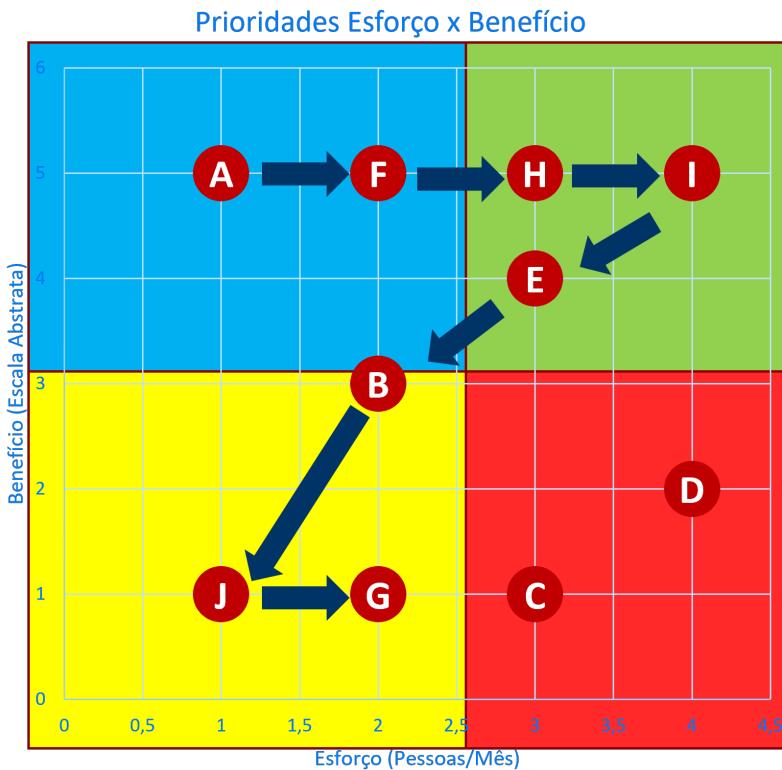


Figura 3.13.: Gráfico com quadrantes, determinados arbitrariamente pelo usuário, para os itens de projeto listados na Tabela 3.2. As setas mostram uma possível sequência de prioridade. Os itens C e D foram considerados desnecessários ao projeto.

### 3.8.6. Prioridade e Dependências

Suponha que um item A seja mais prioritário que um item B, porém A não possa ser usado sem que B esteja pronto. O que fazer?

Esse é um acontecimento comum, e mostra que há uma diferença entre as prioridades do negócio e as prioridades de um projeto que busca atender o negócio. Nesse caso, podem ser investigadas as seguintes soluções:

1. Alterar a forma como o software é desenvolvido, de forma que A funcione sem que B esteja pronto;
2. Dividir B em pedaços, de forma que apenas parte de B seja necessária para A funcionar;
3. Aceitar que B tem que ser feito antes de A.

### 3.9. Valor Financeiro de Projetos

Apesar de não ser o objetivo principal desse capítulo, é interessante observar as formas de calcular o valor financeiro de um projeto, ou subprojeto.

Antes de tratar das estimativas financeiras usadas em projetos, é importante conhecer os seguintes conceitos, como definidos por Dal Zot e Castro (2015):

- **principal** (P) é o capital inicial de um empréstimo ou uma aplicação financeira, também conhecido como **valor presente** (VP), **valor atual** (VA), **valor descontado**, ou por seu nome em inglês, ***present value*** (PV);
- **juro** (J) é a **remuneração** do capital emprestado, da parte de quem paga é uma despesa ou custo financeiro, da parte de quem receber é um rendimento ou renda financeira;
- **montante** (S) é o **saldo**, ou **valor futuro** (VF), ou ***future value*** (FV), de um empréstimo ou de uma aplicação financeira. Pode ser também chamado de **valor de resgate**, ver equação 3.1.
- **prazo** (n) é o período de tempo que dura o empréstimo ou a aplicação financeira, podendo ser medido em dias, meses, anos...
- **prestaçāo ou pagamento** (P,PGTO,PMT) se refere ao valor de pagamentos quando feitos em um número maior do que 1.
- **taxa de juros** (i) , ou simplesmente **taxa**, é o quociente entre juros e o principal, ver equação 3.2;

$$S = P + J \quad (3.1)$$

$$i = \frac{J}{P} \quad (3.2)$$

Analizando essas definições podemos detectar uma propriedade importante do dinheiro e que é a base da Matemática Financeira: o dinheiro muda de valor ao longo do tempo. Normalmente, como os cenários de inflação são muito mais comuns que os de deflação, R%100,00 hoje compram mais que comprarão daqui a um ano e menos do que compravam um ano atrás. Por meio de cálculos como o do Valor Presente, como veremos no caso do Valor Presente Líquido, podemos comparar opções de investimentos ou empréstimos.

Calculadores financeiros, como a HP-12C e o Excel permitem calcular facilmente o Valor Presente e o Valor Futuro com pagamentos, ou retiradas, fixas, em um período determinado, ao com uma taxa de juros fixas.

Por exemplo, se você tiver R\$10.000,00 por mês e puder aplicar em um projeto, deve procurar uma boa alternativa de investimento para poder saber se, financeiramente, vale a pena usar o dinheiro no projeto, ou se é melhor usar esse investimento. Com a fórmula do Valor Futuro (VF em português ou FV em inglês) esse cálculo pode ser facilmente feito no Excel, como mostrado na Figura 3.14.

Investimento Alternativo	
Taxa de Juros	1,00%
Período	12
Pagamento	R\$ 10.000,00
Valor Presente	R\$ 0,00
Valor Futuro	R\$ 126.825,03

(a) Valores e formatação

Investimento Alternativo	
Taxa de Juros	0,01
Períodos	12
Pagamento	10000
Valor Presente	0
Valor Futuro	=FV(Taxa_de_Juros;Períodos;Pagamento;Valor_Presente;0)*-1

(b) Fórmulas

Figura 3.14.: Cálculo do valor futuro de um investimento de R\$10.000,00 por mês com uma taxa de juros de 1%.

Existem várias práticas que são utilizadas para calcular o valor financeiro de um projeto, entre elas (Satpathy et al., 2016):

- Custo Total de Propriedade (TCO - *Total Cost of Ownership*), que calcula todo custo de um serviço ou produto, incluindo a realização e a o uso do mesmo.
- Retorno do Investimento (ROI - *Return on Investment*), o método mais usado que é basicamente o lucro sobre o custo;
- Valor Presente Líquido - VPL (NPV - *Net Peesent Value*), onde se calcula a diferença entre a receita do projeto e seus custo ao longo do tempo, corrigido para o tempo presente, e
- Taxa Interna de Retorno - TIR (IRR - *Internal Rate of Return*) é a taxa de desconto de um investimento que torna o NPV zero para todos os fluxos de caixa de um projeto, não podendo ser calculado analiticamente, quanto maior for, mais desejável é o projeto (Hayes, 2019).

Devemos ter cuidado com projetos com resultados subjetivos, que não podem ser medidos diretamente pelo valor financeiro. Essa questão não será discutida nesse livro.

### 3.9.1. Custo Total de Propriedade

Quanto se analisa o custo de um sistema é normal falar de Custo Total de Propriedade, conhecido pela sigla em inglês TCO (Total Cost of Ownership). O TCO é o valor presente de todos os custos a serem feitos durante a vida de um sistema, produto, serviço ou equipamento(Anklesaria, 2008).

Tabela 3.3.: Valores usados para calcular o TCO de uma impressora.

Impressora	Laser A	Ink A	Laser B
Preço	R\$ 880,00	R\$ 400,00	R\$ 2.100,00
Preço Toner	R\$ 480,00	R\$ 105,00	R\$ 539,00
Páginas Toner	1000	480	12.000

Por exemplo, se decidirmos trocar todo o parque computacional de uma empresa que usa Windows para Linux, mesmo que o custo do Linux seja zero, o TCO é bem alto, pois envolve o processo de troca, novos profissionais, treinamento, etc... Outro exemplo comum é o da compra de uma impressora. Seu TCO não envolve apenas o custo da impressora, mas também o custo do material consumível, quando uma certa produção é prevista. Por isso é que grandes empresas compram menos impressoras, porém impressoras maiores e mais caras, para baixar o TCO, já que o custo por impressão delas acaba saindo menor.

Para o software desenvolvido vale o mesmo conceito. Qual seu TCO? Envolve o preço pago pelo software mais tudo que vai ser pago para possibilitar a implantação e utilização do produto (instalação, cursos de treinamento, manutenção mensal, etc...). Espera-se, em uma decisão racional, que o TCO seja menor que o benefício trazido pelo sistema.

Um exemplo típico é a análise a ser feita na compra de uma impressora. Impressoras mais caras normalmente tem o preço por página impressa mais barato, porém só a partir de um ponto de uma quantidade grande de impressões.

A Tabela 3.3 e a Figura 3.15 mostram como isso acontece. Nelas vemos qual a melhor alternativa, considerando apenas o preço da impressora e do tipo de impressão, qual a impressão mais barata. É possível, por exemplo, somar o gasto de energia a essa conta.

Anklesaria (2008) divide os custos em possíveis classes:

- **preço de compra**, e
- custos internos, que se dividem em
  - **custos de aquisição**, como transporte, inspeção, armazenagem;
  - **custos de uso**, como manutenção, garantias, e
  - **custos do fim de vida**, como custos para desmontar um projeto, etc.

### 3.9.2. Retorno do Investimento

O cálculo do ROI depende de haver uma previsão para a receita corrente  $R$  e para o custo  $C$  do projeto. Sua fórmula é:

$$ROI = \frac{R - C}{C} \quad (3.3)$$

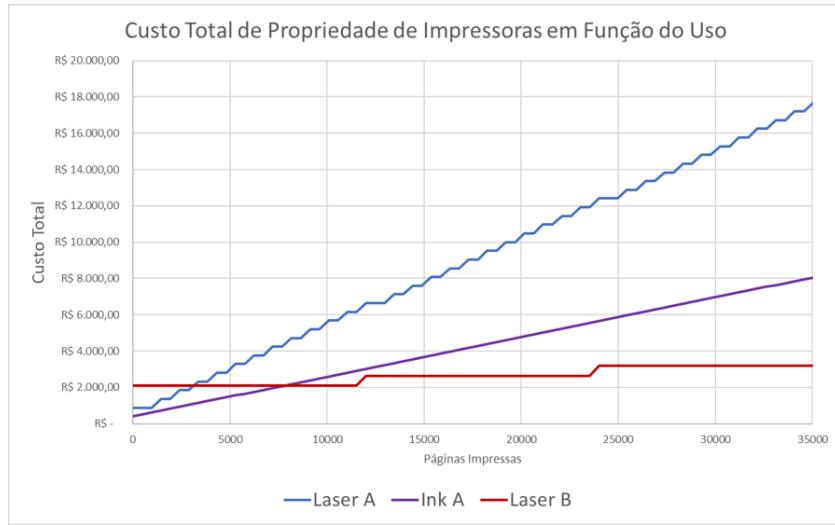


Figura 3.15.: Gráfico do TCO de impressoras em função da quantidade de páginas impressas. Percebe-se que após aproximadamente 3500 páginas é melhor comprar uma impressora a laser mais cara, caso esse fosse o único fator de análise.

O cálculo do ROI de um projeto que está previsto custar R\$ 500.000,00 e pretende obter como benefício um valor de R\$ 700.000,00 é:

$$ROI = \frac{700 - 500}{500} = \frac{200}{500} = 40\% \quad (3.4)$$

O ROI pode ser usado para comparações com taxas que o mercado financeiro paga por um investimento do mesmo valor. Assim, só valeria a pena fazer um projeto, financeiramente falando, se ele resultasse em um retorno mais alto do que poderia ser obtido com o investimento em algum título do mercado.

Um cuidado ao analisar o ROI de um projeto é o fato de ser um valor percentual. Nesse caso, um projeto de R\$ 10.000,00 com um ROI de 150% pode parecer mais interessante que um projeto de R\$ 1.000.000,00 com um ROI de 1%, porém o segundo gera muito mais dinheiro que o primeiro.

### 3.9.3. Valor Presente Líquido

O Valor Presente Líquido é o valor atualizado para o presente de todos os gastos e ganhos de um projeto. Essa atualização é feita com uma taxa de desconto fixa. Isso significa que se você ganhar R\$ 100,00 daqui a um ano, e a inflação for de 5% ao ano, hoje isso tem o valor equivalente ao valor presente que, quando somado de 5%, resultasse em R\$100,00. A conta é  $1/1,05$ , que é aproximadamente R\$95,24.

A fórmula do VPL é(Kenton, 2019):

$$VPL = \sum_{t=1}^T \frac{C_t}{(1 + i)^t} \quad (3.5)$$

onde  $C_t$  é o caixa líquido, receitas menos despesas, em um período  $t$  e  $i$  a taxa de desconto em um investimento alternativo, sendo  $T$  o número de períodos.

Esse cálculo é exemplificado na Tabela 3.4, onde usamos o método de calcular o saldo de cada período e atualizá-lo para o presente, com uma taxa fixa de 5%.

Período	Custo	Receita	Caixa	Valor Presente
0	R\$ 50.000,00	R\$ 0,00	-R\$ 50.000,00	-R\$ 50.000,00
1	R\$ 40.000,00	R\$ 52.500,00	R\$ 12.500,00	R\$ 11.904,76
2	R\$ 40.000,00	R\$ 52.000,00	R\$ 12.000,00	R\$ 10.884,35
3	R\$ 50.000,00	R\$ 61.500,00	R\$ 11.500,00	R\$ 9.934,13
4	R\$ 20.000,00	R\$ 31.000,00	R\$ 11.000,00	R\$ 9.049,73
5	R\$ 20.000,00	R\$ 30.500,00	R\$ 10.500,00	R\$ 8.227,02
6	R\$ 20.000,00	R\$ 30.000,00	R\$ 10.000,00	R\$ 7.462,15
7	R\$ 20.000,00	R\$ 29.500,00	R\$ 9.500,00	R\$ 6.751,47
<b>Total</b>				<b>R\$14.213,63</b>

Tabela 3.4.: Tabela de cálculo do Valor Presente Líquido, considerando saldos diferentes em cada período. Foi usada uma taxa de 5% a cada período. A fórmula para cada Valor Presente é Caixa<sub>i</sub>/(1 + Taxa)<sup>Período</sup>

### 3.9.4. Taxa Interna de Retorno

A Taxa Interna de Retorno é um cálculo um pouco mais complicado, exigindo um processamento de tentativa e erro. Ela indica de quanto deveria ser a taxa de desconto para o VPL ser zero, isto é, o resultado do projeto não ser nem positivo, nem negativo. O processo de cálculala é numérico, mas o Excel e calculadoras financeiras fornecem funções para isso.

A fórmula que relaciona o VPL com o TIR é(Hayes, 2019):

$$0 = VPL = \sum_{t=1}^T \frac{C_t}{(1 + TIR)^t} - C_0 \quad (3.6)$$

onde  $C_0$  é o investimento inicial total.

Quanto mais alto for o TIR, melhor será o resultado do projeto, sempre lembrando que como o cálculo é percentual, isso depende do valor do projeto. Uma maneira de entender isso é que você precisaria de uma aplicação melhor que a taxa para o projeto não valer a pena financeiramente.

### 3.10. Valor em Software

Conhecendo algumas definições na Economia e no Marketing, podemos compreender melhor outras definições que encontramos na Engenharia de Software.

Segundo Erdogmus, Favaro e Halling (2006) , **valor** é a “diferença entre benefícios e custos de um bem, ajustados ao risco, em um certo momento de tempo”.

Ele é dirigido por valores individuais ou coletivos. As partes interessadas esperam obter algum **benefício**, seja ele tangível ou intangível, econômico ou social, monetário ou utilitário, ou ainda estético ou ético (Biffl et al., 2006).

Valor significa esse benefício final, que existe geralmente nos olhos do observado e admite múltiplas caracterizações (Biffl et al., 2006).

O risco, que ainda não tínhamos encontrado em outras definições, é importante, e é parte das técnicas de gestão de projeto modernas considerá-lo. O risco pode ser incluído inclusive indiretamente no custo total previsto de um projeto multiplicando sua probabilidade pelo seu impacto. Isso faz pouco sentido para um risco único, mas já faz sentido para o conjunto total de riscos de um projeto.

Assim, o valor de algo pode ser visto como uma fórmula matemática, de uma forma básica como:

$$\text{benefício} - \text{custos} \quad (3.7)$$

Ou, considerando os riscos:

$$(\text{benefício} - \text{custos}) \times \text{correção por risco} \quad (3.8)$$

A partir dessa interpretação, busca valor é buscar os maiores benefícios a um custo que valha a pena, com risco baixo. Esses benefícios devem ser identificados no início do projeto, e estar alinhados com o planejamento estratégico da empresa.

É importante frisar que o valor deve ser em relação a organização, não ao software propriamente dito. Assim, algumas funcionalidades que poderiam tornar o software muito mais poderoso podem, na prática, não ser úteis, não trazer benefícios, custar caro demais ou ter risco muito alto. E isso depende de cada projeto, ou cada aplicação do software.

Gane e Sarson (1979) introduziram um o acrônimo **IRACIS**, que identifica três formas de agregar valor (Gane e Sarson, 1979; Ruble, 1997):

- Aumentar Faturamento (*Improve Revenue*);
- Evitar Custos (*Avoid Costs*), e
- Melhorar Serviços (*Improve Services*).

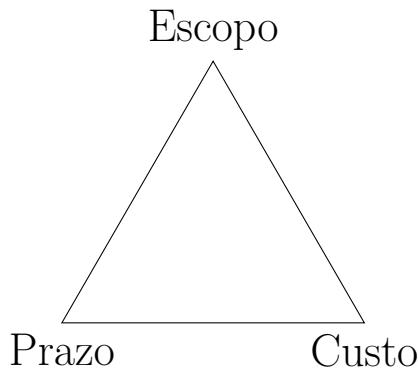


Figura 3.16.: O triângulo do valor para a maioria dos projetos.

### 3.10.1. Triângulo do Valor

É comum se falar, em projetos de software, que existem três parâmetros, que são qualidade (ou escopo), tempo e custo, formando um triângulo, mas o cliente só pode definir dois. Apesar de ser tratado como um brincadeira, é realmente impossível arbitrar esses três valores de forma independente. Ou seja, a afirmação “eu quero um software com a funcionalidade X em tempo Y e custo Z” não pode ser feita sem conhecimento de causa (BeSeen, 2015; Kerzner, 2017).

A princípio, isso pode ser entendido a partir do fato que Boehm et al. (2000) mostraram que há uma relação entre a funcionalidade e qualidade desejada, e o esforço necessário para desenvolver o sistema, o que é óbvio. O que não era tão óbvio é que, além disso, dado um esforço, o tempo necessário para desenvolver o sistema também é determinado. Podem ser feitas certas modificações na previsão estatística, porém se forem feitas no sentido de diminuir o prazo necessário para a entrega do produto, elas implicam em risco maior. O assunto já foi tratado há muito anos por Brooks (1978) em seu livro adequadamente chamado *The Mythical Man-Month*.

Kerzner (2017) trata escopo, tempo e custo como restrições primárias que competem entre si em um projeto tradicional, e ainda apresenta restrições secundárias: reputação e imagem, risco, qualidade e valor. Já em projetos complexos, ainda segundo Kerzner, as restrições primárias passam a ser reputação e imagem, valor e qualidade, enquanto escopo, risco, custo e tempo passam a ser restrições secundárias. Podem aparecer também outros fatores, como segurança e estética, que aparece em parques de diversão, por exemplo.

### 3.10.2. Valor em Métodos Ágeis

É comum que métodos ágeis, como Scrum (Satpathy et al., 2016), proponham priorizar, de forma contínua, em ciclos curtos, os requisitos baseados no valor de negócios que é entregue aos clientes e usuários. O foco do Scrum, por exemplo, é tornar entregar valor, na forma de software executável, rapidamente, por meio de entregues incrementais a

cada ciclo. Não há, porém, uma definição específica do significado da palavra valor, o que leva a necessidade de um entendimento maior do termo. Mesmo assim, cada história do usuário representa um valor a ser entregue ao cliente.

### 3.11. Conclusão

Não existe uma ideia unificada do que é valor, principalmente quando o conceito é comparado entre áreas distintas.

Como os clientes do analista de sistemas, partes interessadas nos projetos, estão em várias áreas diferentes, mesmo em um só projeto, então podem interpretar valor de formas diferentes também.

Mesmo assim, a principal função do desenvolvimento de software é produzir um software que tenha valor ao usuário, seja esse valor calculado por uma fórmula ou um conceito mais qualitativo.

Para isso é importante definir o que é valor no início de qualquer projeto, de maneira consensual entre as partes interessadas. Isso é uma prática ágil que deve sempre ser adotada.

A técnica MoSCoW também é muito prática e pode ser associada a qualquer outra. A partir dela pode ser mais fácil fazer uma ordenação de uma grande lista de requisitos, por exemplo.

Já a técnica de Kano, ou seu uso como proposto pelo Volare, permite trabalhar sobre a importância de cada item. Há, na verdade, uma relação entre o resultado da técnica de Kano e os grupos do método MoSCoW.

O uso dos Elementos de Valor (Almquist, Cleghorn e Sherer, 2018; Almquist, Senior e Bloch, 2016) pode ser uma solução apropriada para o processo de discussão com as partes interessadas em busca de definições mais precisas e completas do verdadeiro valor.

### 3.12. Exercícios

**Exercício 3.1:** Procure outras definições de valor na Economia ou investigue mais sobre as definições dadas. Discuta a diferença entre essas definições e como elas mostram a evolução do entendimento do que é valor.

**Exercício 3.2:** Busque outras definições de valor, principalmente de outras áreas que não foram citadas neste capítulo.

**Exercício 3.3:** Vamos supor que Alice queira abrir um negócio. Para isso ela precisa

de um software que permita registrar suas vendas. Sem esse software, por motivos legais, ela não pode vender. No mercado ela pode encontrar um software básico, que não atende todas as suas necessidades mas atende o requisito legal, por R\$ 100.000,00 por ano. Nesse caso, ela perderia a capacidade de mudar seus preços dinamicamente, o que faria com que ela deixasse de ganhar R\$ 240.000,00 por ano. Porém, a taxa de juros do mercado, se ela investir o dinheiro que usaria para o software novo é de 5% ao ano. Quanto ela pode pagar de aluguel, por ano, pelo software que atende completamente suas necessidades?

**Solução:** A princípio ela pode pagar um valor que 5% seja menor que R\$240.000, isto é R\$4.800.000,00. Se ela investir menos que isso vai ganhar mais no total, se investir mais que isso não vai ganhar tanto quanto ganharia se colocasse o dinheiro para render.

**Exercício 3.4:** Vá para o site <http://jogodeanalisedesistemas.xexeo.net/> e visite a Livraria Resolve. Identifique qual pode ser o conceito de valor para o projeto que essa organização deseja.

**Exercício 3.5:** Este trabalho foi inspirado no “Wake up in the morning game”<sup>7</sup>, adaptado para alunos de faculdade e um curso remoto.

- Listar atividades individualmente. (5 minutos)
  - Cada membro do grupo deve usar stick-notes para listar suas atividades de manhã, desde quando acordam até quando chegam à faculdade.
- Com o grupo, agrupar as atividades individuais em tópicos comuns. (5 minutos)
  - Usar stick-notes de outra cor ou tamanho.
  - Agrupar em função do objetivo.
- Ordenar as atividades no tempo, cronologicamente, da esquerda para direita. (5 minutos)
- Priorizar as os grupos de atividade segundo a técnica MoSCoW - de cima para baixo. (5 minutos)
- O que você faria se acordasse atrasado/tarde para uma reunião muito importante? (5 minutos)
  - Cada membro do grupo colocar estrelas nos stick-notes que não podem deixar de fazer.
  - Não tem limite de estrelas.
- Anotem onde o item estrelado muda a prioridade. (5 minutos)
  - Por exemplo, pode ser necessário pegar um táxi, o que antes seria considerado um “Won’t”.
  - Pode não existir no seu caso.

---

<sup>7</sup><https://www.agilesparks.com/blog/wake-up-in-the-morning-game/>



# 4

## 5W2H

I keep six honest serving-men  
(They taught me all I knew);  
Their names are What and Why  
and When  
And How and Where and Who.

*(Rudyard Kipling)*

## Conteúdo

---

4.1.	As Perguntas Certas . . . . .	80
4.2.	Usos do 5W2H . . . . .	81
4.3.	Onde Está o Valor? . . . . .	82
4.4.	Várias Perguntas para cada Palavra . . . . .	83
4.5.	Exercícios . . . . .	86

Este capítulo apresenta uma técnica informal, porém extremamente útil, de descrição de fatos, coleta de informações, análise crítica análise de situação, ou registro de decisão conhecida como **5W2H**.

Apesar de informal, ela é usada em vários modelos formais, tanto de forma explícita como implícita. Esta regra será usada em todo este livro, também implícita ou explicitamente, para ajudar a explicar os diversos modelos usados em análise de sistemas.

**5W2H** é a sigla a partir das palavras americanas que começam perguntas abertas

## 4.1. As Perguntas Certas

Todo trabalho que vai ser feito precisa ser definido de alguma forma. Para isso é importante que se façam as perguntas certas, e que essas perguntas levem a uma definição precisa, ou seja, não ambígua, do que vai ser feito e quais as condições em que será aceito. Também, quando se descreve algo, é necessário que as informações dadas sobre o que está sendo descrito sejam suficientemente completas para serem úteis. Isso é verdade para a criação de notícias, histórias, acordos de ações a serem tomadas no final das reuniões e, também, para a análise de sistemas, onde estamos na prática descrevendo um sistema de forma a permitir sua implmentação.

Um quantidade muito grande de textos, desde daqueles que tentam ensinar crianças a contar histórias até aqueles que ajudam a profissionais a determinar uma arquitetura de informação empresarial, indica que é necessário responder as seguintes perguntas, conhecidas como 5W2H em inglês:

- Por que? (*Why*)
- O que? (*What*)
- Quem? (*Who*)
- Quando? (*When*)
- Onde? (*Where*)
- Como? (*How*)
- Por quanto? (*How Much*)

Existem variações. Uma lista mais básica só possui 6, exclui o “Por quanto?” (5WH). Listas maiores incluem termos como “ganhos” ou *Wins*, “Quantos”, e variações de “Quem” que são usadas em inglês, como “Whose”, em português “De quem?”, e “Whom”, que é uma versão passiva. Na prática, essa lista de 7 é a mais conhecida na área de negócios e a versão 5WH, em áreas como jornalismo e educação .

A fonte original destas perguntas é o estudo da Ética, e mais tarde da Retórica, na Grécia antiga, e foram chamadas originalmente de **sete circunstâncias** por Aristóteles. Em latim as perguntas eram: *quis, quid, quando, ubi, cur, quem ad modum e quibus adminiculis*. A tradução seria: quem, o que, quando, onde, por que, como e por que meios (Fedotov, 2019; Sloan, 2010).

Por que essas perguntas são boas? Primeiro, quando usadas realmente como perguntas em uma entrevista, são perguntas abertas, que não incluem a resposta dentro delas e obrigam quem responde a responder sem usar apenas uma resposta do tipo “sim” ou “não”. Também são perguntas que, a princípio, não induzem uma resposta. Por exemplo, perguntamos “Quem merece ganhar o prêmio?” em vez de “Alice merece ganhar o prêmio?”, deixando para o entrevistado decidir sem ser influenciado pelo nome de Alice na pergunta. Além disso, as perguntas, reunidas, mostram um grande quadro informational sobre um tema.

## 4.2. Usos do 5W2H

Um das áreas de utilização, por exemplo, é o jornalismo. Se uma notícia vai ser dada, é importante saber responder, no texto da notícia, todas essas perguntas, ou a notícia estará incompleta.

Um exemplo é o seguinte primeiro parágrafo de uma notícia:

BRASÍLIA — Um dia depois de vetar a proibição para que as companhias aéreas cobrem pelo despacho de bagagens, o presidente Jair Bolsonaro justificou a decisão, afirmindo, nesta terça-feira, que “empresas menores alegavam que seria um empecilho” à operação no país e disse que não pretende enviar outra medida ao Congresso Nacional para restringir a cobrança apenas para as chamadas “low cost”, de baixo custo. (Extra, 2019)

As perguntas 5W2H para essa notícia estão respondidas na Tabela 4.1.

Pergunta	Resposta
Onde	Brasília
Quando	Um dia depois de vetar nesta terça-feira
Quem	Jair Bolsonaro
O que	vetar a proibição para que as companhias aéreas cobrem pelo despacho de bagagens
Por que	empresas menores alegavam que seria um empecilho à operação no país

Tabela 4.1.: Perguntas e respostas a partir da notícia.

Outro uso é para registrar decisões de uma região. Para cada item da reunião deve ser registrado:

- O que deve ser feito?
- Quem é o responsável?
- Quando deve ser feito/entregue?
- Onde deve ser feito/entregue?
- Por que deve ser feito?
- Como vai ser feito?
- Quanto vai ser gasto?

Na análise de sistemas, ou na análise de requisitos, as perguntas mais respondidas são ligadas a **quem** deseja **o que** e **por que** deseja. Perguntas adicionais, feitas em algumas práticas ou que tem que ser respondidas nas fases iniciais de um projeto de software são, por exemplo: para **quando** deseja, **por quanto** deseja, **como** deseja que seja feito e **onde** deseja que funcione. Várias outras perguntas podem ser feitas e que também vão influenciar o projeto, como **quem** paga, **quem** vai ser afetado, até **quando** vai funcionar, etc.

### 4.2.1. O Framework de Zachman

O *Framework de Zachman* é uma metodologia de descrição de uma arquitetura de sistemas de informação baseado na técnica 5WH, i.e., não incluindo custos (Sowa e Zachman, 1992; Zachman, 1987). Várias das técnicas que usaremos nesse livro geram artefatos que pertencem ao *Framework de Zachman*. Ele é um dos principais exemplos do uso direto da técnica 5WH em Engenharia de Software.

Sua principal caracterização é o quadro da Figura 4.1.

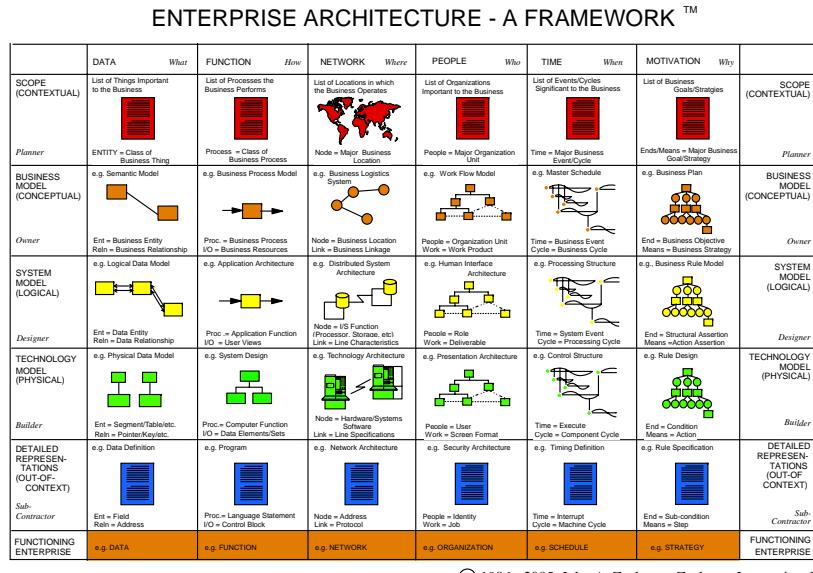


Figura 4.1.: O Framework de Zachman fornece uma visão da arquitetura empresarial a partir de 6 dimensões definidas pelo 5WH. Reprinted with permission by John A. Zachman, Zachman International.

### 4.3. Onde Está o Valor?

Tendo em vista que a proposta é **desenvolver software tendo em vista entregar valor ao cliente**, onde está o valor no método 5W2H?

Normalmente a resposta é razoavelmente simples: o valor está normalmente caracterizado, de alguma forma na resposta a pergunta “por que?” feita a um requisito, pois a justificativa de um requisito indica o motivo pelo qual ele é importante.

É possível que, para descobrir o verdadeiro valor, seja necessário fazer essa pergunta mais de uma vez, isto é, perguntar por que a resposta para a pergunta “por que” foi dada, e então perguntar de novo até que se alcance uma resposta que realmente caracterize o valor. Essa técnica é conhecida como **5 Por QuesSerrat (2017)**.

Algumas vezes também é possível que a resposta necessite de perguntas do tipo *O que*, ampliando o conceito tanto do “por que” quanto do “o que”. Por exemplo, se em algum momento a resposta for “preciso dessa função por causa de uma lei”, a próxima pergunta poderia ser “O que (ou qual) é essa lei?”.

Perguntar por que sucessivamente ou perguntar o que sucessivamente, ou misturar as duas sequências, é uma prática chamada *squeeze and stretch*Higgins (1994).

## 4.4. Várias Perguntas para cada Palavra

Algumas perguntas podem ser feitas mais de uma vez. Por exemplo, “Quem?” podem se referir ao agente ou ao paciente de uma ação, ou a um observador. A pergunta “Quando?” pode se referir ao quando começou ou quando acabou o evento, ou quando ele foi planejado.

Cabe ao profissional entender como usar essas possibilidades ao seu favor. As subseções a seguir mostram algumas opções que podem, e na maioria das vezes devem, ser investigadas em um projeto, sem esgotar as possibilidades.

Além disso, como as práticas de desenvolvimento de software tratada nesse livro podem ser usadas desde o início do projeto, para servir de especificação do mesmo, várias perguntas podem ser feitas não só sobre o produto, mas também sobre o projeto.

### 4.4.1. Perguntas sobre o que

Descobrir o que é a tarefa principal da análise. Basicamente todo este livro fala sobre isso.

Normalmente as perguntas do tipo “o quê” caracterizam mais o produto da análise que o projeto propriamente dito.

- O que será feito?
- O que não será feito?
- O que é necessário?
- O que é suficiente?
- O que é insuficiente?

### 4.4.2. Perguntas sobre quem

As perguntas sobre “quem” são muito importantes em várias fases do projeto. No início porque precisamos identificar as partes interessadas (Capítulo ??). Depois porque cada funcionalidade do sistema vai atender a algum usuário.

#### 4. 5W2H

Uma das técnicas muito usadas em gerência de projetos é, para cada atividade, classificar dos as partes interessadas de acordo com suas responsabilidades, criando uma matriz conhecida como Matriz RACI (PMI, 2017). Essa matriz contém pacotes de trabalho ou atividades nas linhas e partes interessadas nas colunas, sendo que cada célula deve indicar se a parte interessada **realiza**, **aprova**, **é consultada** ou **é informada**.

- Quem deve realizar?
- Quem deve aprovar?
- Quem deve ser consultado?
- Quem deve ser informado?
- Quem deve estar envolvido?
- Quem pode ser afetado?
- Quem pode se beneficiar?
- Quem pode ser prejudicado?
- Quem deve verificar?
- Quem deve validar?
- Quem deve homologar?
- Quem deve pagar?
- Quem deve usar?
- Quem deve operar?
- Quem deve manter?

#### 4.4.3. Perguntas sobre quando

Geralmente as perguntas do tipo “quando” tem relação com prazos. Elas podem se refenciar tanto ao negócio quanto ao projeto, ou seja, existem tempos do negócio, como o prazo para entregar um relatório, e tempos do projeto, como o prazo para entregar uma função no software.

- Quando deve ficar pronto?
- Quando deve iniciar a execução?
- Quando vai acontecer?
- Quando será tarde demais?
- Quando os recursos estarão disponíveis?
- Quando devem ser as entregas?
- Quando deve ser aprovado?

#### 4.4.4. Perguntas sobre onde

Novamente esse tipo de pergunta tem relação com diferentes assuntos. Um é o local do mundo real, por exemplo, um software pode ser usado em uma fábrica, mas não no escritório de uma organização. Outro é em relação a infraestrutura de TI, o software pode rodar na nuvem, em um servidor da empresa ou em celulares. Finalmente, em

relação ao projeto, um software pode ser feito em um laboratório ou por uma equipe espalhada pelo mundo.

- Onde será feito?
- Onde será testado?
- Onde será homologado?
- Onde será instalado?
- Onde será utilizado?
- Onde será mantido?

#### 4.4.5. Perguntas sobre como

Geralmente a fase de Análise, objetivo deste livro, não discute muito o “como”, sendo mais preocupada com o “o quê”, sendo o “como” mais tratado na fase de projeto. Mesmo assim, várias perguntas desse tipo já tem que ser respondidas pela análise, tanto em relação ao produto como ao projeto de construí-lo.

- Como será feito?
- Como será utilizado?
- Como será comercializado?
- Como será corrigido no futuro?
- Como sua operação será gerenciada?

#### 4.4.6. Perguntas sobre por que

Apesar de parecer que os porquês não influenciam a execução do projeto, a verdade é que considerá-los traz uma mudança importante na visão que o analista vai ter da solução. Os porquês são na verdade a justificativa, definem valor e são a orientação do caminho a ser seguido a cada decisão.

- Por que aconteceu?
- Por que será feito?
- Por que foi pedido?
- Por que foi aprovado?
- Por que será aprovado?
- Por que existem resistências?
- Por que alguém é a favor?
- Por que alguém é contra?
- Por que a funcionalidade é desejada?
- Por que o requisito foi pedido?

#### 4. 5W2H

##### 4.4.7. Perguntas sobre quanto

Não existe projeto sem orçamento. Mesmo que você vá trabalhar para si, ou de graça, há um custo em horas e outros recursos necessários. Por isso essas perguntas são normalmente respondidas quando todas as outras já foram respondidas, ou, ao contrário, são respondidas antes porque vão servir de limites ao projeto, limitando as respostas aceitáveis para as outras perguntas.

- Quanto custará fazer?
- Quanto custaria não fazer?
- Quanto será obtido em benefício quando feito?
- Quanto será necessário do recurso X<sup>1</sup>?
- Quanto tempo será necessário?

#### 4.5. Exercícios

**Exercício 4.1:** Vá para o site <http://jogodeanalisedesistemas.xexeo.net/> e visite a Livraria Resolve. A partir da visita, analise as respostas dadas as perguntas 5W2H e veja se elas são adequadas. Escreva uma lista de perguntas adicionais, sempre no formato 5W2H a serem feitas para cada personagem.

---

<sup>1</sup>pessoas, funções no projeto, equipamentos, etc.

# 5

## Modelos e Abstrações

The purpose of abstraction is not to be vague, but to create a new semantic level in which one can be absolutely precise

(Edsger Dijkstra)

### Conteúdo

5.1.	Modelos . . . . .	88
5.2.	Tipos de Abstrações . . . . .	91
5.3.	Trabalhando com as abstrações . . . . .	96
5.4.	Exercícios . . . . .	96

#### Por que modelos e abstrações?

Todas as práticas deste livro realizam alguma forma de abstração do problema ou solução que fornece modelos aos desenvolvedores e outras partes interessadas.

Este capítulo faz uma introdução aos temas correlatos **modelo** e **abstração**, que são as bases de todos os métodos de Análise de Sistema.

Todas as técnicas estudadas nesse livro implicam na criação de um modelo, seja ele do domínio da aplicação, do negócio, do problema, ou do sistema que será implementado, inclusive modelos que fazem a relação entre modelos distintos.

## 5. Modelos e Abstrações

Apesar de usarmos estes termos de forma coloquial, é importante responder às perguntas: o que é um modelo? O que é uma abstração?

Um **modelo** é uma representação de algum objeto, conceito, conjunto de conceitos ou realidade. Modelos são criados para que nós possamos estudar, normalmente segundo algum aspecto escolhido, o objeto modelado. Na grande maioria das vezes, um modelo é uma versão simplificada, ou seja, abstrata do objeto modelado. Essa versão simplificada permite uma comunicação com foco em alguns conceitos, evitando o que é irrelevante (Yourdon, 2006).

**Abstração** é o processo mental de separar um ou mais elementos de uma totalidade complexa de forma a facilitar a sua compreensão por meio de um modelo, eliminando (ou subtraindo) o resto. Segundo Rosen (2018), a “abstração é um processo mental distinto em que novas ideias ou conceitos são formadas pela consideração de vários objetos ou ideias e omitindo características que os distingue.”

A Tabela 5.1 mostra uma sequência de imagens, a partir de uma foto, que ilustram o conceito de abstração. Enquanto a primeira foto<sup>1</sup> é muito detalhada, as seguintes continuam de alguma maneira representando o conceito de mulher, porém de uma forma cada vez mais abstrata, até que se chega em um ícone que não tem nenhuma relação com a imagem de uma mulher, o espelho de Vênus, reconhecido internacionalmente como símbolo do sexo feminino.

Tabela 5.1.: Sequência de imagens que mostram abstrações crescentes de uma imagem inicial que representa uma mulher. Foto por Michael Jatremski.



## 5.1. Modelos

No nosso dia a dia nos deparamos constantemente com modelos. Um mapa é um modelo do território que descreve. Uma maquete de um prédio é um modelo que nos permite ver o prédio a ser construído como um objeto tridimensional. Uma receita de bolo é um modelo do processo para construir o bolo. Mesmo uma foto de um modelo pode ser vista como outro modelo.

Um modelo deve ser simples o bastante para ser fácil de manipular e, simultaneamente, complexo o suficiente de forma a servir para a solução do problema em questão, de acordo com o ponto de vista desejado. Assim, um mapa para navegação tem informações

<sup>1</sup>Foto original por Michael Jatremski, <https://openphoto.net/gallery/image/view/5539>

### 5.1. Modelos

diferentes de uma mapa para estudo do clima, e um modelo de comportamento do software também tem informações diferentes do modelo da informação que o software armazena.

Quanto mais simples o modelo, maior a abstração feita para produzi-lo, ou seja, mais características são suprimidas do objeto original. Chaitin (2006) chega a dizer que se uma teoria, que é também um modelo, é do mesmo tamanho que o dado que ela explica, então não tem valor nenhum. Ainda segundo o autor, uma teoria só é útil quando é uma compressão dos dados, e compreender é comprimir.

No nosso dia a dia utilizamos continuamente a capacidade humana de abstrair para poder trabalhar com toda a informação que o mundo nos fornece. Voltando ao caso do mapa: ele é um modelo de uma região. Dependendo da informação que queremos, colocamos alguns símbolos e tiramos outros do mapa. Um mapa também não pode ser perfeito, tem que “abstrair” as informações que não são necessárias para a utilização que foi construído. Se não houvesse nenhuma abstração, o mapa teria que ter o mesmo tamanho da cidade.

Podemos usar mapas com diversos graus de detalhe, ou seja, mais ou menos abstratos. Um globo terrestre, por exemplo, é um mapa muito abstrato, geralmente com o objetivo de ensinar noções básicas de geografia. Já uma carta náutica é um mapa muito detalhado, que permite a navios ou barcos menores navegarem de forma segura em uma região. Ainda mais detalhada será a planta de uma casa ou prédio. Os níveis de detalhe são infinitos e são usados de acordo com a necessidade do problema a ser resolvido. A Figura 5.1 exemplifica diferentes mapas, com diferentes níveis de abstração adequados ao seu uso.

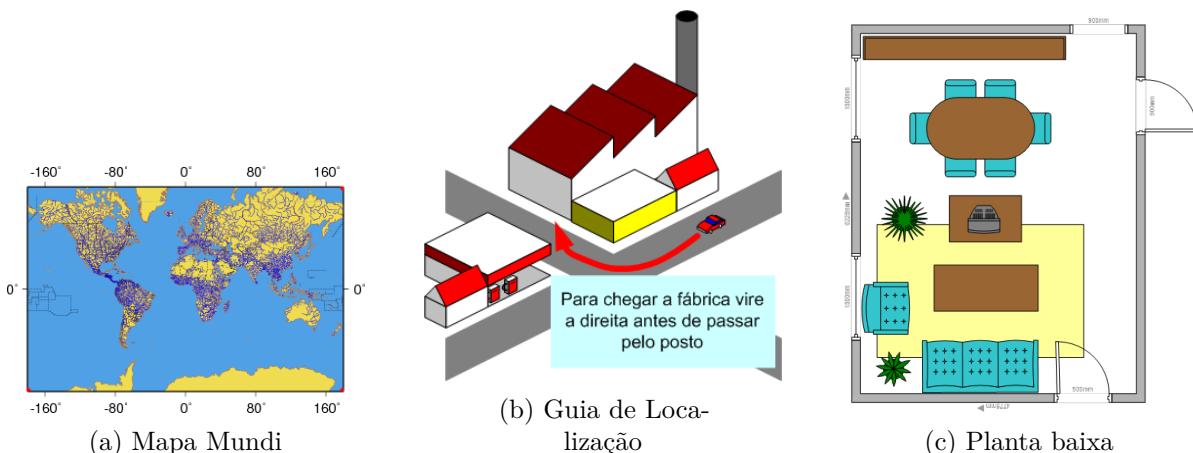


Figura 5.1.: Diferentes tipos de mapas, ou seja, modelos, cada um com um nível diferente de abstração e dedicado a uma utilização distinta.

Um tipo especial de modelo é o protótipo. Um protótipo é um modelo exemplar, no sentido que fornecer um exemplo, onde as simplificações de certo aspecto são muito pequenas ou não existem. Por exemplo, um protótipo de um software pode ter todas as suas telas na versão final, sendo esse o aspecto estudado, mas nenhuma funcionalidade.

## 5. Modelos e Abstrações

Em algumas áreas, onde há a necessidade de fabricar algo em série, protótipos são um modelo de fabricação, completo em funcionalidade, mas que não foram feitos pelo processo final esperado no momento em que serão produzidos em série.

Finalmente, relativo a aplicação de modelos em nosso problema Yourdon (2006) afirmou que o analista de sistema usa modelos para:

- dar foco as características importantes do sistema e diminuindo a influência de características menos importantes;
- discutir mudanças e correções nos requisitos do usuário a um custo baixo e risco mínimo, e
- verificar que o analista de sistema entende corretamente o ambiente do usuário e o documentou de forma que os outros desenvolvedores possam construir o sistema.

Um típico modelo usado em análise de sistemas, na atualidade, é o Diagrama de Casos de Uso, como o apresentado na Figura 5.2.

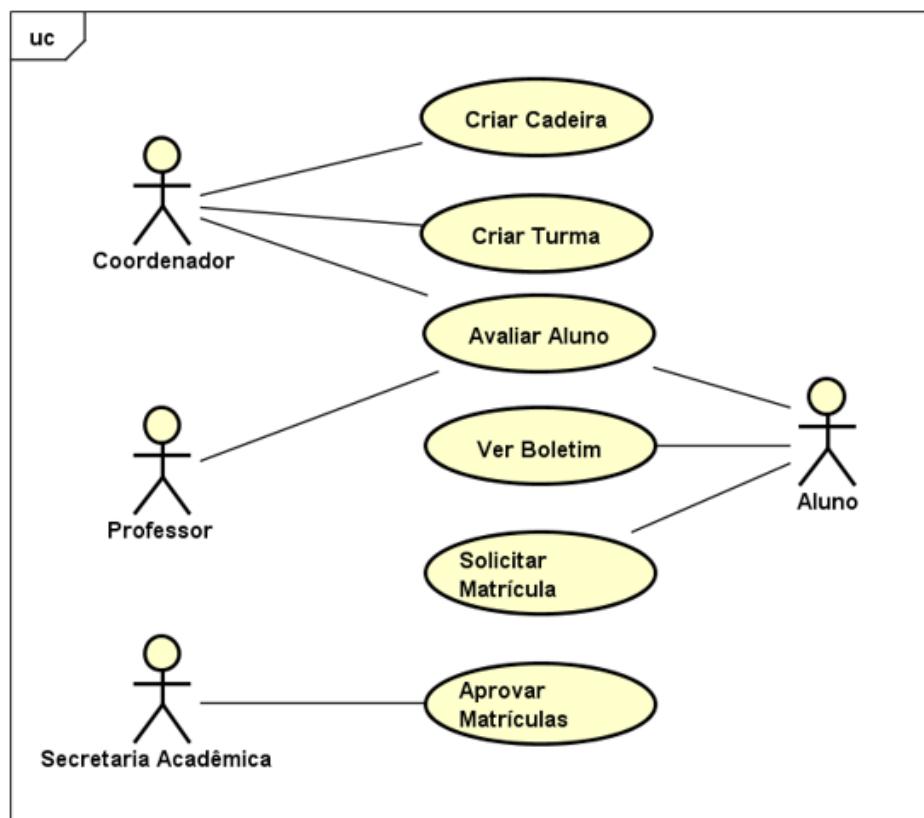


Figura 5.2.: Exemplo de Diagrama de Casos de Uso.

## 5.2. Tipos de Abstrações

Existem várias formas de abstração praticadas no dia a dia ou em situações especiais. No desenvolvimento de sistemas, utilizamos alguns processos de abstração típicos:

- **ocultação da informação** (ou abstração da caixa-preta, ou encapsulamento);
- **classificação**, que relaciona instâncias e classes;
- **generalização**, ou herança, que relaciona classes;
- **composição**, ou todo-parte;
- **identificação**;
- **simplificação pelo Caso Normal**;
- **foco/inibição**;
- **refinamento sucessivo**, e
- **separação de interesses**.

### 5.2.1. Ocultação da Informação

Pela abstração de **Ocultação da Informação**, deixamos de nos preocupar com o interior de uma coisa, só prestando atenção a seu exterior observável.

Podemos encontrar esse exemplo facilmente em muitas coisas do mundo real. Quantas pessoas, por exemplo, sabem como funciona um telefone celular? Poucas, certamente. Porém quase todas sabem usá-lo, porque “abstraem” o seu funcionamento interno (isso é, não pensam nisso) e colocam em foco apenas o comportamento externo.

Por isso chamamos também essa abstração de **abstração de caixa-preta**. O conceito inverso (ou seja, abrir a caixa) é chamado de **caixa-branca**.

Na área de Computação é muito comum ocultar informação para facilitar a compreensão. Por exemplo, em programação, se você usa uma biblioteca de funções, não está interessado em saber como uma função é realizada, mas sim apenas nos parâmetros de entrada e em como será o resultado da função. A função funciona como uma caixa-preta.

### 5.2.2. Classificação

A **classificação** é um mecanismo básico do raciocínio humano. Talvez seja um dos que mais nos habilita a tratar de toda informação que recebemos diariamente.

No processo de classificação eliminamos parte da individualidade do objeto ou sistema analisado e o consideramos como um exemplar de uma **classe**. Quando fazemos isso, aceitamos que esse objeto, agora uma **instância** da classe, divide com todas as outras instâncias da classe um conjunto de características.

## 5. Modelos e Abstrações

Segundo Parrochia (2020), **classificar** é “a operação de distribuir objetos em classes ou grupos, que são, em geral, menos numerosos que eles.”. Essa operação simplifica o mundo, tendo tem vantagens principais Parrochia (2020):

- substitui um multiplicidade caótica e confusa por um ordem racional e regular;
- permite trabalhar com um número reduzido de classes de equivalências em vez de trabalhar com os elementos, e
- detectam uma simetria nas classes que diminui a complexidade do problema e simplifica o mundo.

Na classificação o que estamos fazendo é imaginar uma ideia única que descreve, de forma abstrata, todos os objetos de uma conjunto. Ao eliminar a necessidade de tratar cada objeto de forma única, simplificamos o problema em questão.

A classificação é uma relação entre **instâncias e classes**. É possível associar uma classe com um conjunto de elementos que possuem algumas de propriedades em comum, como todos os alunos de uma universidade ou todos os animais mamíferos.

O processo reverso da classificação é a **instanciação**. O conjunto de todas as instâncias de uma classe é a extensão dessa classe. Uma classe pode ser descrita por sua extensão, ou por sua intenção, que é uma descrição que permita identificar um conjunto de entidades(Parrochia, 2020).

Exemplos típicos de classificação aparecem na Tabela 5.2.

Tabela 5.2.: Exemplos de Classificação

Instâncias	Classe
Flamengo, Fluminense, São Paulo	Times de Futebol
Brasil, Estados Unidos	País
Pelé, Zidane, Messi	Jogador de Futebol

As classificações mais famosas certamente são a Classificação Científica, que classifica todos os seres vivos, e a Classificação Decimal Universal (CDU), que classifica documentos e é utilizada nas bibliotecas.

É importante notar que na vida real um objeto pode pertencer a várias classes. Uma pessoa pode ser um aluno, um professor, um policial, etc... Normalmente, em modelos teóricos como os que vamos usar, tentamos com que um objeto pertença, diretamente, a só uma classe, de modo a facilitar a manipulação do modelo.

Nada impede que uma classe seja uma instância de outra classe, que pode ser chamada de uma **meta-classe**, de uma outra hierarquia. Em Smalltalk-80, por exemplo, toda classe é uma instância da classe *Class*. Também podemos analisar que **Homo** é uma classe que é uma instância da classe *Gênero*, que reúne todos os gêneros. Na prática a classe “Gênero” e as outras similares, como “Espécie” são meta-classes no modelo da Classificação Científica. Não podemos confundir, porém, essa relação com a próxima, a de generalização.

As classificações também são uma forma especial de Ocultação da Informação, pois trabalhando com a classe ocultamos as informações como a identidade.

### 5.2.3. Generalização

Com a **generalização** nós somos capazes de entender como uma classe pode ser descrita por outra classe, mais geral. É importante ver a **diferença entre a classificação e a generalização**: a primeira trata da relação entre objeto e classes, enquanto a segunda trata da relação entre classes.

A generalização é uma relação muito comum entre classes, permitindo que qualquer objeto de uma classe possa ser visto, de uma forma mais geral, como um objeto de outra classe. Por exemplo, “Alice” é uma instância da classe “mulher”, enquanto “Bruno” é uma instância da classe “homem”, porém ambas as classes, “homem” e “mulher” podem ser generalizadas na classe “humano”. Dessa forma, “Alice” e “Bruno” deve ser tratados similarmente quanto membros da classe, mais geral, “humano”, mas podem ser tratados de forma diferente quanto membros das classes, mais específicas, “mulher” e “homem”. Utilizando judiciosamente a generalização podemos simplificar a forma de tratar objetos de classes similares em uma hierarquia.

O processo reverso da generalização é a **especialização**. Ela é uma relação entre classes, ao contrário da classificação que é uma relação entre classes e instâncias.

Exemplos típicos de generalização aparecem na Tabela 5.3.

Tabela 5.3.: Exemplos de Generalização

Classes	Classe mais geral
Funcionário, Aluno, Professor	Pessoa
Automóvel, Avião, Navio	Meio de Transporte
Computador, Rádio, Televisão	Aparelhos Eletrônicos

Novamente a Classificação Científica dos seres vivos nos oferece um grande exemplo da relação de generalização. Cada instância da Espécie *Homo sapiens* é uma instância da Espécie *Homo*, logo a classe *Homo* é uma generalização da classe *Homo Sapiens*.

Na linguística a relação de generalização é conhecida como **hiperonímia**, e a especialização é conhecida como **hiponímia**.

### 5.2.4. Composição ou Agregação

Na **composição** entendemos um objeto complexo, formado de um conjunto de outros objetos, como um só objeto. É como vemos uma bicicleta ou um carro. Ao eliminar a necessidade de descrever as partes, simplificamos a compreensão do objeto analisado.

## 5. Modelos e Abstrações

O processo reverso da composição é a **decomposição**. A Figura 5.3 mostra um brinquedo, o todo, e as partes que o compõe. As partes em separado não são o brinquedo.

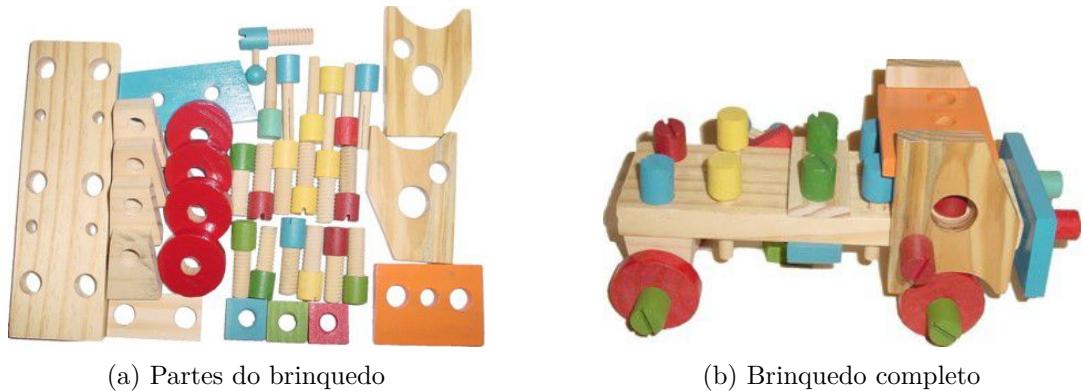


Figura 5.3.: Um brinquedo é composto de suas partes. Foto do autor

Exemplos típicos de composição aparecem na Tabela 5.4.

Normalmente, em modelagem de dados, usamos o conceito de composição para dizer que uma classe (como endereço) é uma característica de outra classe, descrevendo um entre seus atributos.

Tabela 5.4.: Exemplos de Composição

Partes	Objeto
Pneus, motor, etc.	Carro
Capa, centenas de folhas, etc.	Livro
Cabelo, pele, ossos, etc.	Homem

Na linguística a parte é conhecida como um **merônimo** e o todo como um **holônimo**. As relações são de **meronímia** ou **holonímia**.

### 5.2.5. Identificação

Com a **identificação** nós somos capazes de entender como caracterizar unicamente um objeto. Um nome identifica uma pessoa, por exemplo. Ao identificar unicamente um objeto podemos separá-lo de outro objeto semelhante e atribuir a entidades específicas atributos e características que só pertencem a ela, e não pertencem a outros elementos daquela classe.

Há uma diferença entre instanciar e identificar. Uma instância deve possuir uma identificação e uma identificação se aplica a uma instância. A identificação permite a que duas instâncias sejam reconhecidas como distintas ou como representações de um mesmo objeto (normalmente devendo ser reunidas em uma).

### 5.2.6. Simplificação pelo Caso Normal

Toda aplicação, ao funcionar, deve tratar de casos específicos que ocorrem durante o funcionamento normal, sejam elas alternativas que são aceitas no negócio, como alternativas não aceitas ou mesmo erros do sistema. Porém, é bem mais fácil discutir o funcionamento normal antes e depois os casos específicos.

A abstração de **simplificação pelo caso normal** indica que devemos começar a trabalhar pelo modo comum ou normal de funcionamento, ou ainda melhor, o modo onde tudo ocorre da forma mais simples e depois ir inserindo mecanismos para tratar das variações possíveis, por meio da especificação das **condições** que levam a essas variações.

### 5.2.7. Foco e Inibição

Uma das características importantes do ser humano é ser capaz de prestar atenção, isto é, por o foco em um detalhe, inibindo parcialmente os outros detalhes ao redor, e assim processar a informação, detalhe a detalhe.

Podemos ver essa forma de abstração acontecer no dia a dia, quando estamos olhando para um local e as áreas ao redor ficam levemente vigiadas pela visão periférica, mas não estamos realmente prestando atenção nas mesmas.

A abstração de simplificação pelo caso normal é uma forma de colocar o foco em uma questão e inibir as outras, i.e., as variações, que depois serão o foco da análise uma a uma, mantendo as outras inibidas.

Isso também acontece em um modelo de dados. Cada parte de um modelo foca em alguma informação que pretendemos registrar, e possui regiões ao redor que nos informam outras informações adicionais, mas não precisamos olhar ao detalhe da outra parte para entender aquela. Por isso separamos o sistema em entidades, como “aluno”, “curso” e “professor”, e depois analisamos cada uma em separado.

Tecnicamente falando, foco e inibição são muitas vezes representados pela modularização e divisão do sistema em partes estanques, com as características de alta coesão, todo um módulo trata apenas de um assunto, e baixo acoplamento, um módulo não interfere no outro. Outra forma de usar o foco e a inibição é no refinamento sucessivo.

### 5.2.8. Refinamento Sucessivo

A técnica de **refinamento sucessivo** indica que cada problema deve ser tratado de uma forma mais geral para depois ser analisado de uma forma mais específicas, normalmente seguindo o conceito de “explodir” um problema anterior (mais geral) em sub-problemas mais específicos, sendo cada um desses sub-problemas “explodidos” também até chegarmos a um problema de solução simples.

## 5. Modelos e Abstrações

O refinamento sucessivo é uma forma mais específica da abstração de Foco e Inibição e faz parte de várias estratégias de abstração baseadas no conceito de **dividir para conquistar**. Equivale a uma estratégia *top-down* de solução de problemas, cuja a inversa é a estratégia *bottom-up*

### 5.2.9. Separação de Interesses

**Separação de interesses** é o processo de abstração onde tentamos descrever, ou produzir, um conceito separando-o em conceitos distintos com a menor quantidade possível de interseção, baseado em algum aspecto específico do problema sendo tratado. Esses conceitos normalmente são características ou comportamentos.

A separação de interesses é uma forma mais específica da abstração de Foco e Inibição e também faz parte de várias estratégias de abstração baseadas no conceito de dividir para conquistar.

## 5.3. Trabalhando com as abstrações

Imagine que precisamos descrever comprar um carro. É óbvio que todo carro possui quatro pneus, um motor, etc. Isso é uma classe bastante geral. Porém, desejamos ainda falar sobre um modelo específico: uma Ferrari Testarossa, por exemplo. Logo, acabamos de especializar nosso modelo, mas ainda não chegamos ao nível de objeto. Quando vemos o carro específico, aí temos o objeto. Ele é identificável como instância daquela classe porque apesar de dividir várias características em comum com outros objetos da classe, também tem algumas características únicas, como o número de série do chassi. Finalmente, desejamos trocar a cor do assento do carro. Nesse instante, já estamos vendendo uma parte do carro, decompondo-o em suas partes.

## 5.4. Exercícios

**Exercício 5.1:** Vá para o site <http://jogodeanalisedesistemas.xexeo.net/> e visite a Livraria Resolve. Faça um modelo na forma de um diagrama de toda a informação levantada. Pode ser, por exemplo, como um Mapa Mental.

# 6

## Dados e Informação

The goal is to turn data into information, and information into insight.

(Carly Fiorina)

### Conteúdo

---

6.1.	Usando Significados Diferentes . . . . .	98
6.2.	Conclusão . . . . .	99

Este capítulo trata de quatro palavras que no dia a dia parecem ser sinônimas, mas têm significados diferentes quando usadas na Análise de Dados: dado, informação, conhecimento e sabedoria.

Vamos recorrer a dicionários para ter uma definição inicial. Segundo o American Heritage, **informação**, no contexto de computadores, “é o dado quando processado, guardado ou transmitido”(American Heritage, 2019).

Já no dicionário Aurélio, informação, entre outros significados, pode ser “Conhecimento amplo e bem fundamentado, resultante da análise e combinação de vários informes”, “Coleção de fatos ou de outros dados fornecidos à máquina, a fim de se objetivar um processamento” ou ainda “Segundo a teoria da informação, medida da redução da incerteza, sobre um determinado estado de coisas, por intermédio de uma mensagem”.

Apesar de não estarmos diretamente envolvidos com a teoria da informação, não podemos deixar de notar a importância da definição que diz que a **informação reduz a incerteza por meio de uma mensagem**.

Na teoria da informação a quantidade de informação de uma mensagem pode ser medida como o menor número de bits para representar ela completamente.

## 6.1. Usando Significados Diferentes

Dados, informação, conhecimento e sabedoria<sup>1</sup> são quatro palavras que na língua corrente possuem uma interseção de significados, porém são consideradas diferentes em áreas específicas, como Computação e Engenharia de Sistemas e Sistemas de Informação.

A distinção entre essas palavras e a existência de uma hierarquia de entendimento entre elas é uma discussão corrente, construída em cima de uma pirâmide que registra a complexidade do que esses termos significam, conhecida como hierarquia DIKW (Rowley, 2007).

Diversos autores propõe definições diferentes. Zins (2007) chega a apresentar 44 definições diferentes para dados, informação e conhecimento, fornecidas por especialistas no assunto reunidos em um painel, usando a metodologia *Critical Delphi* (Zins, 2012).

Seguindo a classificação de Zins (2007), preferimos uma conceituação baseada em um modelo onde dados e informação são fenômenos externos, enquanto conhecimento, e consequentemente sabedoria, são fenômenos internos. Isso significa que dados e informação são do domínio universal, externos à mente, enquanto conhecimento e sabedoria são do domínio subjetivo, internos à mente (Zins, 2007).

**Dados** são apenas os símbolos que usamos para representar a informação, o registro de diferentes aspectos de um fato ou fenômeno, como Os números que guardamos em um banco de dados. Dados não são interpretados, eles existem, são adquiridos de alguma forma, via coleta, pesquisa ou criação, guardados de outra forma e, possivelmente, apresentados em uma terceira. O computador é uma máquina programável que manipula dados.

Por outro lado, informação é o dado com significado, normalmente processado de forma a ser útil. Uma informação deve permitir responder perguntas como “quando”, “quanto”, “quem”, etc.

$$\text{Informação} = \text{Dado} + \text{Significado} \quad (6.1)$$

É necessário fazer um mapeamento entre dados e informação. Esse mapeamento pode ser simples ou complexo, dependendo de várias variáveis envolvidas, que vão desde decisões arbitrárias tomadas pelo desenvolvedor até padrões internacionais. Por exemplo, em muitos sistemas é preciso ter a informação do sexo de uma pessoa (masculino ou feminino). Para isso, são guardados um número (1 ou 0) ou uma letra (M ou F), que é o dado que faz o registro da informação.

Existem outras definições para dados e informação, mas esta diferenciação entre símbolo e significado será suficiente neste livro. Em especial, na Teoria da Informação tem uma abordagem onde o significado não é importante, apenas a quantidade de mensagens possíveis (Claude E Shannon, 1963).

---

<sup>1</sup>Em inglês, *wisdom*

Definir conhecimento e sabedoria é muito mais complexo, sendo conceitos em que há mais incerteza ou mesmo desacordo sobre o que realmente significam formalmente, além do significado ambíguo dessas palavras na língua portuguesa ou inglesa. Porém, vamos tomar algumas decisões sobre esses significados, sempre no contexto deste livro.

Sumarizando vários autores, Rowley (2007) define conhecimento como “uma mistura de informação, entendimento, capacidades, experiências, habilidades e valores.” Além disso, o mesmo autor lembra a diferença entre conhecimento explícito, que está codificado em documentos, bancos de dados e outros registros, e tácito, que é está na mente das pessoas, ou nos processos que elas realizam. Essa diferença foi bastante explorada por Nonaka e Takeuchi (1995), que apontam também a dificuldade de processar computacionalmente conhecimento tácito.

Finalmente, Rowley (2007) afirma que vários autores tratando de dados, informação e conhecimento não falam de sabedoria, e aponta uma publicação anterior sua, que define sabedoria como “a capacidade de por em ação o comportamento mais apropriado, levando em conta o que é conhecido (conhecimento) e o que traz o bem maior (considerações sociais e éticas).”(Rowley, 2006).

Rowley (2007) também nos lembra que enquanto dados são muito fáceis de processar e contém pouco significado, e ficam na base da hierarquia, sabedoria é muito difícil de processar e fica no topo da mesma. Isso resulta em uma pirâmide como na Figura 6.1.

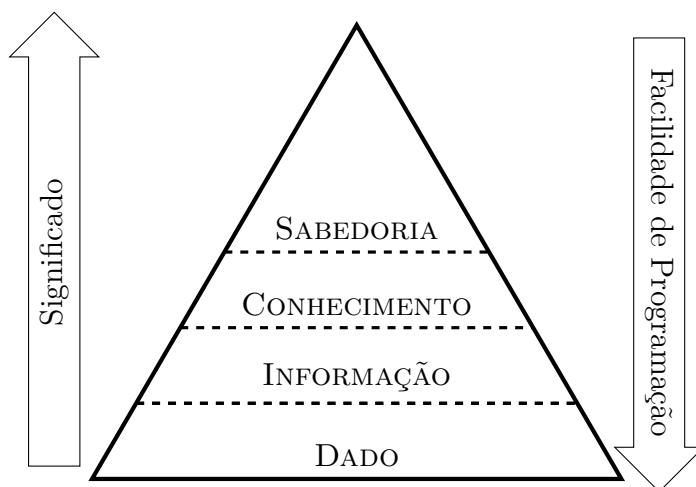


Figura 6.1.: Pirâmide da hierarquia do DIKW. Fonte:(Rowley, 2007)

## 6.2. Conclusão

Por exemplo, um sistema pode ter conter milhares de registros representando vendas, com números guardados em um formato binário que nada significa para o usuário, mas o usuário vai poder ver em um relatório o valor total vendido no mês, informação que é relevante para ele.

## *6. Dados e Informação*

O conhecimento normalmente está mapeado nos algoritmos do sistema, nas regras de negócio implementadas. Alguns sistemas, que por exemplo usam aprendizado de máquina, podem inferir novos conhecimentos. Um exemplo disso seria o sistema de vendas descobrir que um tipo de item é mais vendido no início do mês, ou sempre junto de outro item, o que é chamado de análise de cesta de compras.

Já a sabedoria parece, por enquanto, estar ainda nas mãos dos seres humanos.

# 7

## Sistema

Sufficiently simple natural structures are predictable but uncontrollable, whereas sufficiently complex symbolic descriptions are controllable but unpredictable.

(Howard Patte)

### Conteúdo

7.1.	Tipos de Sistemas . . . . .	103
7.2.	Sistemas Automatizados . . . . .	105
7.3.	Princípios Gerais de Sistemas . . . . .	105

#### Por que sistemas?

Este livro é sobre análise de sistemas de informação, e para entender o que é um sistema de informação é preciso antes entender o que é um sistema.

O dicionário Houaiss oferece duas definições da palavra sistema, entre várias, que são adequadas a nossa visão:

- “conjunto de elementos, concretos ou abstratos, intelectualmente organizado” (Houaiss, Villar e Mello Franco, 2009), e
- “conjunto de pessoas, procedimentos e equipamento projetado, construído, operado e mantido com a finalidade de coletar, registrar, processar, armazenar, recuperar

## 7. Sistema

e exibir informação, podendo assim servir-se de diferentes tecnologias”(Houaiss, Villar e Mello Franco, 2009).

A primeira definição é bem geral e se aplica a qualquer sistema. De acordo com essa definição, um sistema existe quando um conjunto de um tipo variados de coisas é entendido como organizado de alguma forma, sejam essas coisas concretas ou apenas conceitos.

A partir dessa definição podemos pensar sobre o Sistema Solar, entendendo como um conjunto de uma estrela, planetas e outros corpos organizados pela lei da gravidade, por exemplo. Ou um relógio, como um sistema de peças organizadas por conexões mecânicas.

A segunda já é uma definição específica que se adapta melhor a um tema que é o principal foco deste livro: sistemas de informações.

Bunge é um importante filósofo argentino que define formalmente o conceito de sistema. Apesar de ser uma definição sofisticada e também traduzida em notação matemática, ela é simples de entender e muito parecida com a primeira definição do Dicionário Houaiss, de forma que atende a visão mais ampla do que é um sistema.

Segundo ele, um sistema é composto de: um conjunto de **componentes**, um **ambiente**, que é um conjunto de itens nos quais ele é conectado, e uma **estrutura**, que é a relação entre os componentes e entre eles e a estrutura.

Um de seus exemplos de sistemas é uma escola, seus componentes são os funcionários e dos alunos, seu ambiente é a sua sociedade, e a estrutura são as relação de ensinar, aprende, gerenciar e outras.

Bunge (1979) também chama atenção que elementos de sistemas precisam agir entre si, o que ele chama de conexão, que é uma relação mais forte do que a simples associação. Existem dois tipos de conexões, a ação de um elemento sobre o outro, no caso um é o agente e outro o paciente, e a interação, onde existem ações recíprocas. Assim, uma família é um sistema, mas não um conjunto de estados de uma coisa, porque não há uma conexão, apenas uma relação. Um sistema (concreto) só existe para Bunge (1979) se possui ao menos duas coisas que se conectam dentro de um ambiente.

A ideia básica de Bunge (1979) para a importância de sistemas é a famosa frase “o todo é maior que as somas de suas partes”. O sistema tem novas propriedades que emergem das ações e interações entre as partes e que não existem nas partes quando isoladas.

Seguindo essa linha, Bertalanffy (1969) defende que os elementos de um “complexo” podem ser distinguidos de três maneiras:

1. pelo seu número;
2. pela sua espécie, e
3. pelas suas relações.

As duas primeiras maneiras de considerar os elementos são dependentes apenas deles, de forma que suas características são mantidas dentro e fora do complexo, já a terceira

é dependente do que acontece dentro do complexo, logo para entendê-las não basta conhecer as partes.

Esse raciocínio é bem aplicado a sistemas de informação, já que as partes devem ser entendidas como um todo, dentro do sistema, e não isoladamente.

## 7.1. Tipos de Sistemas

Podemos encontrar três divisões básicas sobre sistemas:

- quanto a sua natureza, entre sistemas **naturais** e **artificiais**, ou seja, feitos pelos homens(Yourdon, 2006);
- quanto a sua origem, entre **abstratos**, como sistemas de informação, e **concretos**, como um automóvel, e
- quanto ao seu tipo, entre sistemas **abertos**, que recebem influência do meio externo, como matéria, energia ou informação, e **fechados**, que não recebem(Bertalanffy, 1969).

A primeira divisão é fácil de entender. Sistemas naturais estão presentes na Natureza, como sistemas estelares, o sistema digestivo e o Sistema Solar, representado na Figura 7.1. Já os sistemas artificiais são criados pelo ser humano, como o sistema aeroviário, o sistema elétrico e sistemas de informática. Podemos chamar a atenção que, de certa forma, os sistemas naturais são também imaginados pelos seres humanos, que vêm em um conjunto de coisas inter-relacionadas as características de um sistema, porém humanos não os constroem. Sistemas artificiais são construídos por seres humanos. Assim, humanos decidem que partes compõem o Sistema Solar, e até decidem de que classe alguns objetos são, como Plutão, que deixou de ser um planeta, mas não tem interferência na existência desse sistema.

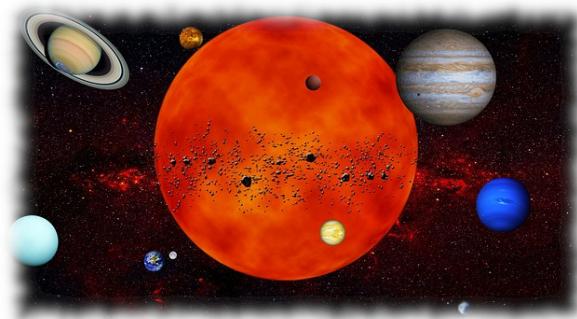


Figura 7.1.: O Sistema Solar é um sistema natural. Imagem por Don Cloud no Pixabay 

Já para entender a terceira divisão, entre sistemas abertos e fechados, é preciso levar em conta a forma como são analisados. Um sistema fechado não troca matéria, energia,

## 7. Sistema



Figura 7.2.: Um *smartphone* é um sistema artificial. Imagem CC BY-NC

ou informação, com o ambiente ao seu redor ou outro sistema. Já um sistema aberto faz essas trocas. Sistemas fechados são simplificações teóricas.

## 7.2. Sistemas Automatizados

A raça humana não só criou seus próprios sistemas, como máquinas, fábricas e sistemas de informação, como também os automatizou, isto é, criou formas para que esses sistemas funcionem com uma intervenção muito menor do ser humano. É só comparar uma fábrica automática de biscoitos com o trabalho artesanal de um confeiteiro. A fábrica vai seguir uma receita, misturar ingredientes, bater, assar e empacotar os biscoitos, tendo vantagens sobre o ser humano;

Nós criamos sistemas automatizados para:

- aumentar a produção;
- diminuir o custo;
- acelerar os tempos;
- aumentar a segurança de processos;
- atender a questões políticas;
- diminuir a carga de trabalho insalubre sobre seres humanos;
- diminuir a variação na produção, etc.

## 7.3. Princípios Gerais de Sistemas

Yourdon (2006) propõe alguns princípios gerais de sistemas<sup>1</sup>:

- quanto mais especializado um sistema é, menos ele é capaz de se adaptar a circunstâncias diferentes;
- quanto maior é um sistema, mais de seus recursos tem que ser dedicados a sua operação e manutenção diária;
- sistemas são sempre partes de sistemas maiores e sempre podem ser particionados em sistemas menores;
- é comum que sistemas cresçam, e
- as interações entre componentes de um sistema são muitas vezes complexas e sutis.

---

<sup>1</sup>E em especial sistemas de informação



# 8

## Organizações

Every company has two organizational structures: The formal one is written on the charts; the other is the everyday relationship of the men and women in the organization.

(Harold Geneen)

### Conteúdo

---

8.1.	Funções Típicas de uma Organização . . . . .	109
8.2.	Estrutura das Organizações . . . . .	109
8.3.	Estratégia Organizacional . . . . .	111
8.4.	A Necessidade de Sistemas de Informação . . . . .	111
8.5.	Processo de Informatização nas Organizações . . . . .	112
8.6.	A necessidade centralização dos dados . . . . .	114
8.7.	Organograma . . . . .	114
8.8.	Conclusão . . . . .	120
8.9.	Exercícios . . . . .	120

## 8. Organizações

### Por que organizações?

Este livro é sobre análise de sistemas de informação, e prioritariamente sobre aqueles que trazem valor para organizações, principais demandantes desse tipo de sistema.

Neste texto, usamos o termo **organização** para representar, de forma geral, todas as pessoas jurídicas e suas subdivisões. Empresas, departamentos e seções, associações, órgãos governamentais, clubes, igrejas, etc. Uma forma melhor de dizer é que queremos discutir sobre todas as pessoas **menos** as pessoas naturais ou pessoas físicas.

Está claro que **uma organização é um sistema**. Ela é compostas de partes, como pessoas, que se interrelacionam e tem relações com o ambiente. Elas são sistemas abertos e artificiais.

Por exemplo, uma escola privada é um sistema composto de:

- proprietários;
- diretores;
- administradores;
- coordenadores acadêmicos;
- professores;
- funcionários administrativos;
- funcionários de serviços gerais;
- alunos;
- pais de alunos;
- currículos;
- material escolar, e
- prédios e suas salas.

Como interação entre as partes temos, entre outras: professores dão aulas para alunos dentro de uma sala de aula seguindo um currículo e usando material escolar.

Segundo Chiavenato (2014), a organização “é um conjunto de cargos funcionais e hierárquicos a cujas prescrições e normas de comportamento todos os seus membros devem se sujeitar”, com o princípio que seus membros se comportam racionalmente. O autor ainda chama atenção para o fato das organizações serem a forma dominante de instituição no mundo moderno. Notamos que essa definição só inclui pessoas.

Se mesmo uma pessoa física já precisa de sistemas de informações, na forma de agenda, caderno de telefone, uso de planilhas eletrônicas para calcular os gastos do mês, mais ainda se pode esperar de uma organização. Isso acontece por vários motivos, entre eles a necessidade de manter uma memória, a de guardar informações para seu funcionamento, obrigações impostas pelo governo e outras.

Uma organização tem um propósito de ser. Para as mais complexas ou mais bem gerenciadas, este propósito é definido em um planejamento estratégico que conta com a definição de uma missão, uma visão, metas e objetivos. Muitas vezes são definidos *KPI*

### 8.1. Funções Típicas de uma Organização

- *Key Performance Indicators*, que permitem medir o desempenho da organização em relação ao que foi planejado.

Quanto maior a organização, maior a quantidade de dados que ela gera. Estes dados vêm tanto da interação com o mundo externo, quanto da comunicação e gestão interna. Esses dados representam informações que explicam de várias formas o que é a organização e qual o seu desempenho. Para manipular esses dados são usados os **Sistemas de Informação**.

Assim, a organização precisa que os dados estejam disponíveis para se conhecer. Ao mesmo tempo, ela precisa que esses dados sejam de boa qualidade.

## 8.1. Funções Típicas de uma Organização

Toda organização tem que cumprir algumas funções básicas para seu funcionamento. Chiavenato (2014) faz uma primeira divisão em cinco grandes funções:

- produção, operações ou serviços;
- comercial ou marketing;
- financeira;
- recursos humanos, e
- administrativas.

Várias outras funções podem ser citadas, como a engenharia, a contábil, a de TI, a inovação, etc. Essas funções são normalmente organizadas na estrutura da empresa. É comum, por exemplo, que a fundão de TI fique subordinada a um diretor financeiro<sup>1</sup> ou administrativo. Em organizações mais voltadas para a informação, já existem cargos de diretoria para a área, como o CIO (*Chief Information Officer*).

Essas funções interagem. Por exemplo, marketing faz um plano de propaganda, que faz com que vendas aconteçam, os produtos vendidos são feitos pela produção, de acordo com um projeto da engenharia, a partir de insumos comprados pela compra e guardados pelo estoque, baseado na pesquisa e inovação. Tudo é entregue pela logística. A cobrança é feita pelo financeiro, que informa a contabilidade. O jurídico faz os contratos de todas as vendas.

## 8.2. Estrutura das Organizações

As organizações, em uma visão neoclássica, podem se estruturar de três formas (Chiavenato, 2014):

- **linear**, a estrutura mais simples e antiga, inspirada nos exércitos e igrejas do passado, hierárquica e com a autoridade única do superior sobre seus subordinados,

---

<sup>1</sup>Mesmo que isso não faça muito sentido.

## 8. Organizações

onde as linhas de comunicação são formais e sempre pela hierarquia da organização, com aspecto piramidal e decisões centralizadas;

- **funcional**, onde a autoridade é relativa ao conhecimento e especialização, cada funcionário se reportando a vários chefes, de acordo com a especialidade, com linhas de comunicação diretas, e decisões descentralizadas, e
- **linha-staff**. resultado da combinação dos dois tipos anteriores, buscando as vantagens dos dois, havendo dois tipos de órgão, de linha, que tratam dos objetivos da organização e seguem uma autoridade linear, e *staff*, que assessoram, planejam, controlam, etc. com autoridade funcional.

Já mais modernamente se reconheceu a estrutura **matricial**, onde se combina a departamentalização funcional com a por produto ou projeto na mesma organização.

Além disso, Chiavenato (2014) também mostra que as organizações tradicionais têm, pelo menos, três níveis:

1. **operações**, de nível técnico e onde existem os executores e supervisores;
2. **planos**, de nível gerencial, onde estão gerentes e chefes, também chamado de **tático**, e
3. **decisões**, de nível institucional, onde estão os executivos a nível de direção, também chamado de **estratégico**.

Mesmo hoje em dia, muitas organizações ainda podem ser caracterizadas dentro dessas formas. As teorias, porém, evoluíram. A era da informação causou mudanças drásticas nas estruturas de poder das organizações, e também causou o achatamento da hierarquia, eliminando a necessidade de vários níveis de gerência. Novas formas de gerenciar apareceram e levaram novamente a estruturas diferentes.

As tendências atuais são Chiavenato (2014):

- cadeias de comando mais curtas, com menos níveis hierárquicos;
- menos unidade de comando, com o crescimento dos relacionamentos horizontais sobre os verticais (estruturas tradicionais de comando);
- maiores amplitudes de controle e autonomia, com delegação de responsabilidade e menos supervisão direta;
- maior participação e empoderamento, com transferência de responsabilidades e delegação;
- ênfase em equipes de trabalho e projetos;
- organização em unidades de negócio;
- menos controles de comportamento e foco nos objetivos e resultados;
- forte infraestrutura de TI;
- foco no negócio essencial, com terceirização das atividades não essenciais;
- consolidação da economia do conhecimento;
- construção de competências;
- envolvimento de terceiros, e
- governança corporativa.

## 8.3. Estratégia Organizacional

Dentro das mudanças nas organizações citamos o foco no negócio essencial, que implica no desenvolvimento de uma estratégia organizacional, isto é, um “padrão ou plano que integra os objetivos globais de uma organização e as políticas e ações em um todo coerente.”(Chiavenato, 2014).

Uma das principais características do planejamento estratégico é a definição da missão e da visão da empresa. A **missão** é a finalidade da organização, ou seja, responde três perguntas(Chiavenato, 2014):

- quem somos;
- o que fazemos, e
- por que fazemos.

Já a **visão** é declaração de como a empresa se vê no futuro, como um projeto de futuro, em um determinado prazo, sendo “o destino que se pretende transformar em realidade”(Chiavenato, 2014).

Segundo a OMG (2015), na especificação do *Business Motivation Model*, a missão descreve os meios para alcançar a visão, que é o fim. A visão então permite gerar **objetivos** e **metas**, enquanto a missão vai levar a um curso de ação baseado em estratégias e táticas, e diretrizes na forma de regras e políticas empresariais.

## 8.4. A Necessidade de Sistemas de Informação

Se mesmo uma pessoa física já precisa de sistemas de informações, na forma de agenda, caderno de telefone, uso de planilhas eletrônicas para calcular os gastos do mês, mais ainda se pode esperar de uma organização.

Algumas organizações tem nas informações sua operação principal, como uma empresa que vende entradas de cinema, e já tem, no seu nível operacional, a TI como ferramente essencial. Porém todas as organizações têm a necessidade de controlar e gerenciar todas as informações que dispõem sobre si mesmas e sobre o mercado, de modo a permitir que suas decisões produzam ações que gerem benefícios e evitem prejuízos, de curto, médio e longo prazo, nos níveis operacional, tático e estratégico. Além disso, outros motivos existem como a necessidade de manter uma memória, a de guardar informações para seu funcionamento, e obrigações impostas pelo governo. Hoje em dia, por exemplo, as empresas devem fornecer todas as suas notas fiscais de forma eletrônica para os órgãos da receita.

As informações necessárias para a organização aparecem em todo seu ambiente. Cada ação realizada, cada decisão tomada, gera, ou pelo menos deveria gerar, um registro que pode ser utilizado mais tarde para a tomada de decisões. Por exemplo, cada venda feita por uma cadeia de lojas de varejo pode ser incluída em um gráfico ou relatório

## 8. Organizações

que permita entender o desempenho das vendas por local, por produto, por vendedor e até mesmo pela hora do dia. Em um uso mais avançado, esses dados podem ser correlacionados, com o processo de seleção da área de Recursos Humanos, de maneira a tentar desenvolver um perfil do vendedor a ser contratado, tanto de forma geral como para atender necessidades específicas. Com relação aos dados de mercado, podem ser usados para prever o desempenho de uma nova loja em uma localidade específica, de acordo com dados demográficos ou informações sobre outros tipos de comércio ou mesmo de consumo de água ou eletricidade.

Tradicionalmente, quanto maior a organização, maior a quantidade de dados que ela gera. Estes dados vêm tanto da interação com o mundo externo, quanto da comunicação e gestão interna. Esses dados representam informações que explicam de várias formas o que é a organização e qual o seu desempenho. No século XXI mesmo pequenas organizações, até mesmo compostas de um só homem, já possuem, por desejo ou obrigação, algum grau de informatização. Porém apareceram novas empresas, totalmente informatizadas, que operam dentro de ambientes virtuais, como a internet, e manipulam grande quantidades de informação mesmo sendo pequenas.

Para ser possível operar, controlar e gerenciar essas organizações é necessário implantar sistemas de informação.

### 8.5. Processo de Informatização nas Organizações

Frente a importância da informação para o sucesso da organização, é fácil entender que, ao longo de sua vida, ela deve passar por um processo contínuo de informatização, a fim de atender as demandas cada vez maiores tanto da própria organização, quanto de seus fornecedores, parceiros, clientes e também do Estado.

Esse processo é necessário e geralmente benéfico, mas não sem problemas.

Entre os principais problemas do processo de informatização das organizações está o fado dele ser normalmente de crescimento vegetativo, emergente, *bottom-up* e por demanda imediata. Os sistemas são criados um a um, em função de necessidades específicas trazidas pelas partes interessadas, e construídos de modo a atender principalmente os requisitos dessas partes, muitas vezes desconsiderando como as atividades e informações desses sistemas podem ser úteis para o resto da organização. Isso gera um descasamento entre os sistemas e a necessidade de interfaces complexas. Além disso, são usadas tecnologias diferentes, mesmo porque as mais antigas vão saindo do mercado, não restando alternativa que migrar.

Por outro lado, tentativas de criar grandes sistemas integrados que atendam toda a organização parecem estar fadados ao fracasso. O tamanho de um sistema é um fortíssimo fator de risco (Gibbs, 1994; R. S. Pressman e Maxim, 2014), já que a relação do tamanho com a taxa de fracasso é mais que linear.

## 8.5. Processo de Informatização nas Organizações

Como sempre, as melhores soluções, ou pelos menos as soluções mais viáveis, parecem estar em um meio termo, ou pelo menos em decisões tomadas de forma consciente dos riscos e benefícios de cada opção.

Não se pode também deixar de levar em consideração que organizações diferentes têm gestões diferentes desse processo. Enquanto uma pode escolher os sistemas de informação a serem implantados de forma *ad-hoc*, outra, mais corretamente, pode possuir um planejamento estratégico que indica em que direção a área de Tecnologia da Informação deve seguir e que sistemas serão estratégicos para o seu progresso, criando programas que comportam vários projetos e analisando frequentemente seu portfólio de software.

Esse quadro bastante variado, evoluiu muito tecnologicamente e no entendimento do negócio informatizado nos últimos anos. Alguns grandes marcos, como o aparecimento do computador comercial, do computador pessoal, das redes locais e dos sistemas cliente-servidor, da Internet e, mais recentemente, das tecnologias em nuvem, causaram grandes mudanças na forma de pensar a própria organização, criando até mesmo a possibilidade de organizações virtuais.

Ao logo desse tempo houve também um aprendizado que trouxe soluções bastante adequadas aos problemas comuns de gestão das organizações, como o uso de sistemas **ERP** e a compra de software **COTS**.

**COTS** significa **Commercial Of The Shelf**, e indica software pronto que fornece funções específicas com nenhuma ou pouca adaptação para quem o compra ou implementa. Principalmente pequenos negócios e profissionais liberais podem ter quase todas suas necessidades atendidas por meio de esse tipo de software. Algumas áreas de aplicação, mesmo em empresas grandes, podem se beneficiar desse tipo de produto. Deve ficar claro, porém, que nesse caso a organização deve se adaptar as práticas implementadas no produto.

Já os sistemas **ERP**, **Entreprise Resource Planning**, são na prática sistemas de gestão empresarial customizáveis destinados a integrar fortemente as áreas de negócio. Sua principal característica é funcionar em funções do negócio cuja tarefa é bem definida, e cuja experiência de desenvolvimento de software é muito forte, e seu maior sucesso vem da implementação, já no software, das melhores práticas do negócio. Assim, muitas empresas adotam não só os ERP, mas também as suas práticas nessas áreas, abandonando a sua forma de trabalhar em troca de uma melhor prática do mercado.

Soluções ERP hoje são capazes de servir grande parte das necessidades genéricas de todas as empresas, desde funções de Recursos Humanos até detalhes de Contabilidade.

Essas soluções, porém, são genéricas. Mesmo quando customizáveis, dificilmente atendem necessidades importantes dos processos das cadeias de valor das empresas, pois estes são específicos e poucas empresas desejam remodelá-los de acordo com uma prática comum no mercado, até mesmo porque suas diferenças podem fornecer justamente o diferencial competitivo que possuem. Assim, o software desenvolvido sob encomenda, interna ou externamente, ainda possui, e certamente sempre possuirá, um presença marcante em todos as organizações.

## 8. Organizações

Todos esses sistemas geram dados. Quase que escondidos nesses dados estão as informações necessárias para a tomada de decisão.

### 8.6. A necessidade centralização dos dados

O outro processo de grande sucesso foi a centralização dos dados dos sistemas operacionais e transacionais (OLTP) em bases de dados integradas na empresa, principalmente em SGDBs Relacionais de grande porte.

O quadro do primeiro quarto do século XXI é que a maioria das organizações entende que seus dados operacionais devem estar sob controle centralizado. O ideal é que todos esses sistemas compartilhem uma mesma base, porém necessidades como desempenho, custo de implementação, estratégia empresarial, permanência da tecnologia usada no mercado, e ainda outras, podem levar a existência de mais de uma base.

Esta situação também não é sem seus problemas. A centralização dos dados sob um controle único pode tanto auxiliar como trazer dificuldades para o desenvolvimento rápido de sistemas, principalmente quando há exigência de muita burocracia para alterar a base ou estruturar informações de forma a atender novas demandas de negócio.

### 8.7. Organograma

A forma mais simples de representar uma empresa é provavelmente o organograma.

Um **organograma** é uma descrição da organização de uma empresa, amplamente divulgada, descrevendo as suas áreas e as hierarquias entre elas, de comando ou comunicação. O Organograma é ferramenta essencial na compreensão de uma empresa e suas linhas de poder.

Organogramas são diagramas que descrevem a estrutura formal de uma organização incluindo suas relações hierárquicas, normalmente por meio de linhas e retângulos. São simples de ler, porém alguns organogramas são mais complicados que outros, de forma a representar informações adicionais.

Normalmente um organograma tem o formato de um grafo hierárquico, onde no alto está uma caixa com a posição mais importante da organização, e nos níveis abaixo aparecem caixas com seus subordinados. Cada caixa pode representar um cargo, uma função ou mesmo um departamento. A Figura 8.1 mostra um organograma simples.

Aris, uma linguagem de modelagem de processos apresentada no Capítulo ??, também permite representar organogramas, e isso é feito como na Figura 8.2.

Em geral o analista não precisa levantar o organograma, pois a empresa já o possui, mas é comum que haja algumas mudanças não registradas que ele deve corrigir. Na verdade, não é trabalho do analista de sistemas construir o organograma da empresa, porém

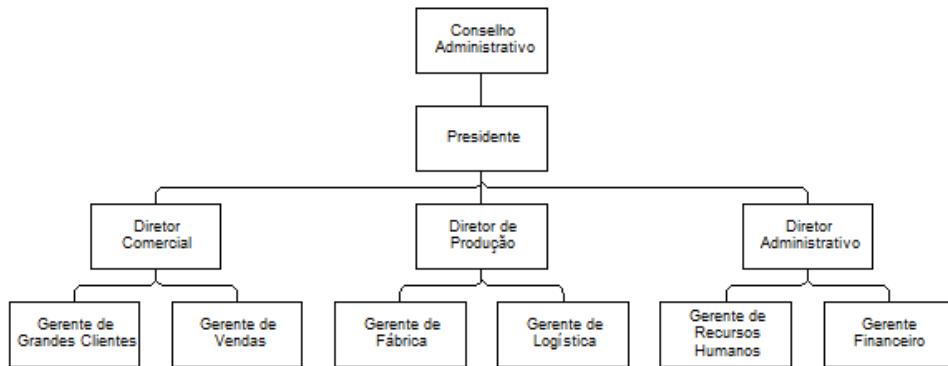


Figura 8.1.: Um organograma simples, de uma empresa hipotética.

ele precisa conhecê-lo para melhor desenvolver o seu trabalho. Para isso é importante obter esse documento junto ao cliente, e verificar não só a hierarquia de cargos, mas também quem ocupa cada cargo, e como entrar em contato com cada um dos membros da organização que possa ter interesse no sistema. O organograma é uma boa ferramenta para o levantamento inicial das partes interessadas.

A importância de conhecer o organograma da empresa se reflete tanto na modelagem propriamente dita, pois ele fornece a descrição da empresa que será convertida para objetos do modelo, como no processo de modelagem, pois a partir do organograma é obtido o conhecimento de cargos e responsabilidades, definindo pessoas a serem entrevistadas.

Ao levantar o organograma, pode ser interessante também solicitar documentos que registrem ou levantam com entrevista as descrições e responsabilidades dos cargos, se elas existirem. Apesar de recomendado, a existência desses documentos não só é infrequente, como muitas vezes mostra um quadro fora da realidade, então o analista deve também validar toda a informação.

Este texto não tratará do processo de levantamento do organograma, pois essa prática é mais afeita à administração. Fica, porém, o lembrete de sua importância como documento de referência ao analista. Em todo caso, em poucas entrevistas pode ser levantado um organograma se não perfeito, adequado ao trabalho de análise.

### 8.7.1. Relações em um organograma

Um organograma pode ser utilizado para representar diferentes formas de subordinação, como a subordinação direta (onde o subordinado deve cumprir as ordens de seu chefe), a assessoria (onde o assessor fornece conselhos e pareceres) e a subordinação funcional (onde o superior pode determinar funções e métodos a outras áreas).

Normalmente a subordinação direta é representada por uma linha cheia vertical, a assessoria por uma linha cheia horizontal e a subordinação funcional por uma linha pontilhada. As Figuras 8.3 e 8.4 exemplificam alguns desses casos.

## 8. Organizações

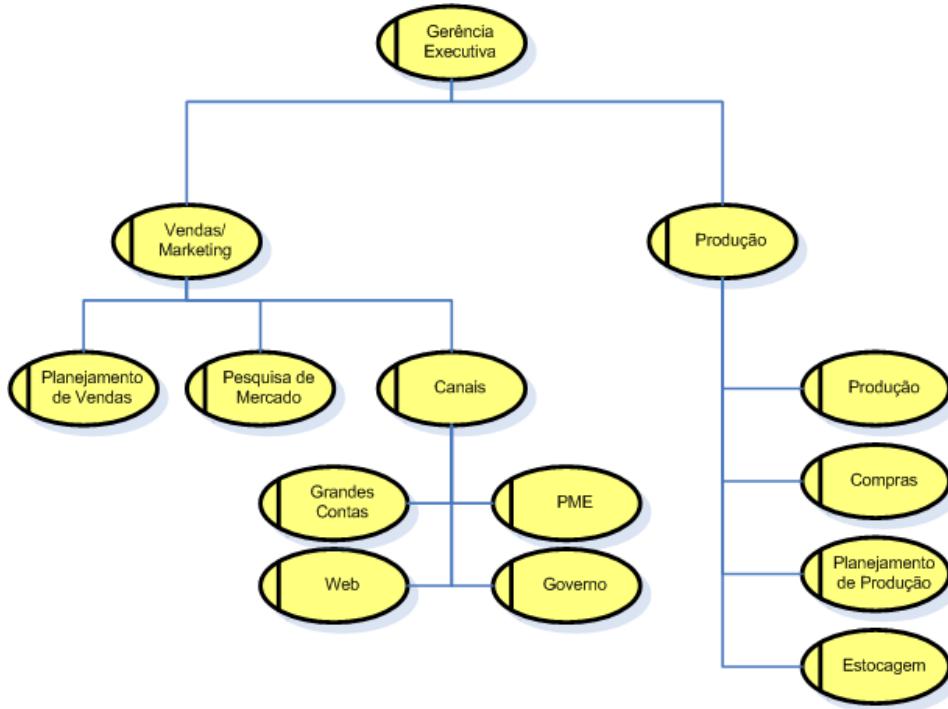


Figura 8.2.: Um exemplo de um organograma de um pedaço de uma empresa hipotética escrito em ARIS.

### 8.7.2. Outros formatos de organograma

Na grande maioria das vezes serão encontrados nas organizações organogramas clássicos ou pequenas variações. Existem, porém, algumas formas alternativas. Uma forma interessante, também com muitas variações, é a radial ou solar. A Figura 8.5 mostra um organograma radial com

Outra forma possível é inverter o organograma. O organograma da Figura 8.6, criado pelo site Guia de Direitos, mostra a estrutura do Poder Judiciário, não tratando de cargos, mas sim de partes desse poder.

Devido à grande variedade de formas usadas, organogramas podem ser desenhados tanto em sistemas específicos como em softwares genéricos de desenho.

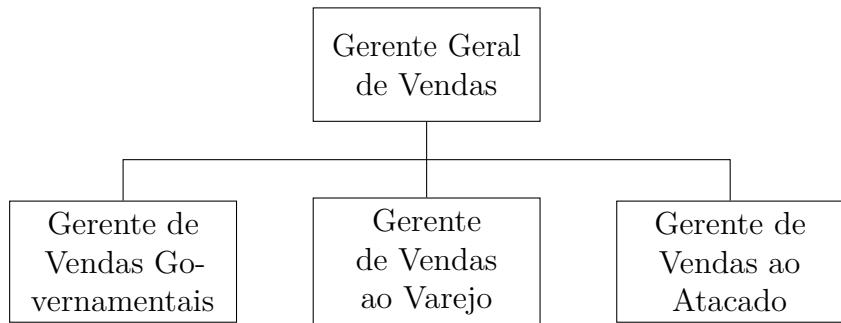


Figura 8.3.: A relação de subordinação, normalmente mantida na vertical, entre o gerente geral e seus gerentes subordinados, que aparecem em um mesmo nível.

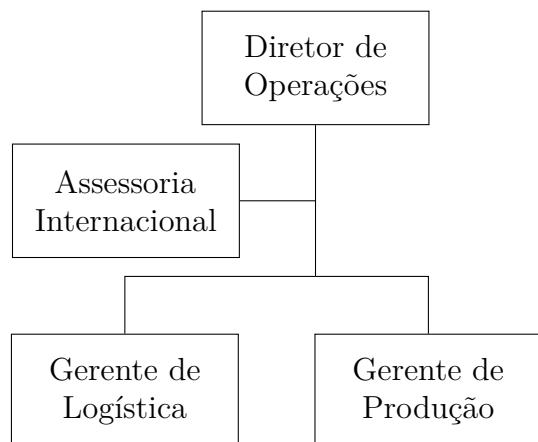


Figura 8.4.: A relação de assessoria é normalmente feita com uma barra horizontal, como a Assessoria Internacional aparece no diagrama acima.

## 8. Organizações

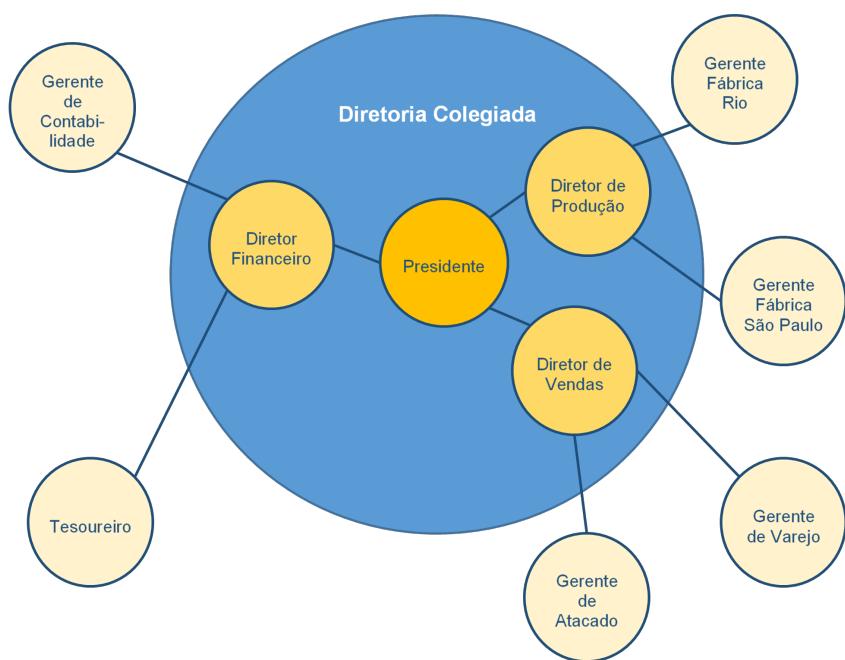


Figura 8.5.: Um exemplo de organograma radial, com uma espécie de “diagrama de Venn” indicando que os diretores e o presidente compõe a diretoria colegiada.



Figura 8.6.: Organograma do Poder Judiciário, em forma inversa, criado pelo site Guia de Direitos, deixado em Copyleft.

## 8.8. Conclusão

Em todo caso, a organização moderna não pode viver sem a Tecnologia da Informação. Ela é essencial, se não para atender sua função produtiva e suas outras funções, principalmente a administrativa.

Além disso, todo sistema de informação deve estar associado a estratégia da organização, em busca de um valor real para o negócio da organização. Isso é uma mudança de mentalidade importante que vem se fortalecendo cada vez mais.

## 8.9. Exercícios

**Exercício 8.1:** Vá para o site <http://jogodeanalisedesistemas.xexeo.net/> e visite a Livraria Resolve. Identifique, para cada personagem, qual função da organização ele ajuda a realizar.

**Exercício 8.2:** Vá para o site <http://jogodeanalisedesistemas.xexeo.net/> e visite a Livraria Resolve. Faça um organograma da livraria. Se necessário, planeje mais perguntas que podem ser feitas aos personagens, invente uma resposta razoável, e desenhe o organograma de forma a atender essas respostas.

# 9

## Sistemas de Informação

Information technology and business are becoming inextricably interwoven. I don't think anybody can talk meaningfully about one without the talking about the other.

(Bill Gates)

### Conteúdo

9.1.	Características dos Sistemas de Informação . . . . .	123
9.2.	Sistemas de Informação Típicos e a Organização . . . . .	124
9.3.	Tipos de Projetos de Sistemas de Informação . . . . .	124
9.4.	Porque são feitos projetos de SI . . . . .	125
9.5.	O Poder está com o usuário . . . . .	126
9.6.	Exercícios . . . . .	127

#### Por que sistemas de informação?

Este livro é sobre análise de sistemas de informação, logo é preciso entender o que são.

## 9. Sistemas de Informação

Sistemas de Informação são utilizados em organizações para planejamento, monitoração, comunicação e controle das suas atividades, por meio da manipulação e guarda de informações.

Segundo o Dicionário Aurélio, a palavra **sistema** significa, entre outras coisas, um “Conjunto particular de instrumentos e convenções adotados com o fim de dar uma informação”. Esta acepção claramente se refere ao que chamamos de **sistema de informação**. Os instrumentos são as ferramentas, os mecanismos, concretos ou abstratos, que utilizamos para fazer funcionar os sistemas, tais como: programas de computador, relatórios, formulários, etc. As convenções são as suas regras de utilização.

K. Laudon e J. Laudon (2011) fornecem uma definição formal para um **Sistema de Informação**: “Conjunto de componentes inter-relacionados, que coletam (ou recuperam), processam, armazenam e distribuem informações destinadas a apoiar a tomada de decisões, a coordenação e o controle de uma organização”

Sistemas de informação são sistemas artificiais, abertos e abstratos.

Um exemplo típico de sistema de informação é um sistema de controle de empréstimos de uma biblioteca. Entre suas várias finalidades, a principal é certamente controlar o empréstimo dos livros, informando quem está com qual livro em um determinado momento (quando). Além disso, o sistema permite outras atividades, como a gerência do estoque de livros, ou seja, que livros estão na biblioteca, a monitoração das livros mais e menos emprestados ou usados, etc.

Outro exemplo de sistema de informações é um *Dashboard* estratégico que informa o valor dos *KPI*, *Key Performance Indicators*, da organização, permitindo um diagnóstico de seu funcionamento e a verificação de sua aderência ao planejamento estratégico

A Figura 9.1 mostra uma tela de um sistema real (BDO) que gerencia uma frota de caminhões. Nesta tela são informados os dados sobre uma viagem do caminhão.

The screenshot shows a Windows application window titled "Cadastro de BDO". The interface is in Portuguese. At the top, there are input fields for "Nºm. BDO", "Data", "Turno", "BDO Subst.", "Frota", and "Motorista", along with dropdown menus for "Veículo", "Roteiro", and "Gerência". To the right of these are two checkboxes: "Não Calcular Horas Improdutivas" and "Não Calcular Horas Extras". Below these are buttons for "Pesobruto", "Local >>", and "Peso". The main area contains a large grid table with columns for "Saída Garagem KM" and "Hora", "Chegada Gerência KM" and "Hora", "Saída Gerência KM" and "Hora", and "Chegada Garagem KM" and "Hora". Below this grid is a "Período Defeito" section with "Início" and "Fim" date pickers and a "Motivo" dropdown. On the left, there's a "Tabela de Cálculo" section with checkboxes for "Utilizar Tabela Substituída" and "Tipo de Veículo para Cálculo". On the right, there are sections for "Usuário Cadastro" (with "Hora" and "Dt" fields), "Usuário Alteração" (with "Hora" and "Dt" fields), and "Usuário Alteração" (with "Hora" and "Dt" fields). At the bottom, there are several icons for file operations (New, Open, Save, Print, etc.) and a "BDO Cancelado" button.

Figura 9.1.: Uma tela de um sistema de informações real

Apesar de estarmos preocupados com o desenvolvimento de sistemas de informação automatizados, implementados na forma de programas de computador, isso não é uma

## 9.1. Características dos Sistemas de Informação

necessidade. Durante séculos as organizações usaram sistemas de informação apenas com o uso de pessoas, papel e tinta. Apenas bem mais tarde, aparecem máquinas como máquinas de escrever e de somar. Não seria exagerado dizer que a escrita e os números foram criados para suportar os primeiros sistemas de informação, que tratavam, por exemplo, de colheitas e comércio. Muitas das técnicas deste livro podem, e devem, ser aplicadas para entender sistemas de informação manuais.

Uma organização possui muitos sistemas de informação, integrados ou não. Mesmo uma pequena empresa caseira pode ser gerenciada com várias planilhas, enquanto uma grande multi-nacional vai certamente possuir de um grande sistema de *Enterprise Resource Planning* (ERP) , que é o nome dado a sistemas de gestão empresarial, e mais uma miríade de outros sistemas.

Este capítulo apresenta uma breve descrição de como funciona, para que servem e quem usa os sistemas de informação dentro de uma organização.

### 9.1. Características dos Sistemas de Informação

É importante entender que sistemas de informação são sistemas **interativos** e **reativos**(McMenamin e Palmer, 1984).

Interativo significa que o sistema troca informações com o ambiente, em especial com os agentes externos que fazem parte desse ambiente, pessoas e outros sistemas de computador. O sistema só faz sentido se é capaz dessa interação(McMenamin e Palmer, 1984).

Reativo significa que o sistema funciona reagindo a mudanças no ambiente, e em especial, a mudanças provocadas pelos agentes externos(McMenamin e Palmer, 1984).

Nossos sistemas também são **sistemas de respostas planejadas**. Isso significa que nossas respostas são determinadas e determinísticas, que podemos criar um programa que as produza. Também significa que todas as perguntas que podem ser feitas ao sistema podem, e são, identificadas previamente(McMenamin e Palmer, 1984).

Escolhendo essas regras de modelagem, escolhemos um caminho para decidir quando o sistema vai funcionar: em vez de deixar isso incerto, como em muitos métodos, nós determinamos que o sistema só funciona para responder um evento no ambiente, causado por um agente externo, e que possua uma resposta planejada.

As metodologias de desenvolvimento apresentadas neste livro são feitas sob medida para sistemas interativos e reativos, de respostas planejadas. Nesse caso, somos ao mesmo tempo restritivos, pois se o sistema não pode ser modelado dessa forma não serve para nossa metodologia, como ampliativos, pois a grande maioria dos sistemas pode ser modelada de forma natural com esses princípios.

## 9.2. Sistemas de Informação Típicos e a Organização

Atualmente já consideremos que vários sistemas de informações típicos de uma empresa são necessidades básicas que podem ser atendidas de uma só vez. Esses sistemas constroem o que é comumente chamado de ERPs – de Enterprise Resource Planning – ou Sistemas de Gestão Empresarial em português – mas que na prática não são sistemas de planejamento (ou de recursos), mas sim de controle e administração de uma empresa.

Entre as características encontradas em ERPs podemos citar a integração das atividades da empresa e o uso de um banco de dados único. O líder mundial do mercado é a SAP AG, com o produto SAP R/3. O custo de implantação de um ERP de grande porte pode chegar até 300 milhões de dólares. No Brasil, existem produtos menos ambiciosos e mais baratos.

Os sistemas de ERP atuais contêm módulos representando os mais típicos sistemas de informações necessários em uma empresa, tais como: Contabilidade Fiscal, Contabilidade Gerencial, Orçamento e Execução Orçamentária, Ativo Fixo, Caixa e bancos, Fluxo de Caixa, Aplicações e Empréstimos, Contas a Receber, Contas a Pagar, Controle de Viagens, Controle de Inadimplência, Administração dos preços de venda, Compras, Controle de fretes, Controle de contratos, Controle de investimentos, Cotações de vendas, Estoque, Exportação, Faturamento, Gerenciamento de armazéns, Importação, Obrigações fiscais, Pedidos, Previsão de vendas, Recebimento, Gestão de informação de RH, Pagadoria, Treinamento, RH scorecard, Planejamento de RH, Planejamento de produção, Planejamento da capacidade, Custos industriais, Controle de chão de fábrica, Controle da produção, Configurador de produtos, Planejamento de Manutenção, Acompanhamento de Manutenção e ainda muitos outros...

## 9.3. Tipos de Projetos de Sistemas de Informação

Existem três tipos de projeto de sistemas de informação: manual, manual para automático e re-automação. Os processos de re-automação ainda podem se dividir em recodificação, re-projeto e re-desenvolvimento, melhoria ou manutenção.

Todos esses tipos de projeto apresentam ao analista de sistemas o mesmo desafio: descobrir o que deve ser feito. Porém, cada tipo apresenta certas particularidades que facilitam ou dificultam esse trabalho de análise.

O trabalho do analista em sistemas manuais é mais relacionado à formalização, por meio de documentação e padrões, de processos já adotados, a criação de novos processos e a transformação de processos existentes tendo em vista otimizá-los ou possibilitar que atendam novas necessidades da organização. Esses processos podem ser bastante complexos e convolutos em alguns casos, o que exige do analista uma boa capacidade de compreensão e modelagem.

Porém, como não serão transformados em programas de computador, o analista pode trabalhar com ferramentais mais informais e mais próximas ao dia a dia do usuário.

Os projetos que apresentam maior dificuldade são os de passagem do processo manual para o automático. Isso acontece porque normalmente esses projetos exigem todo o trabalho feito no tipo anterior, e, de forma adicional, a criação de um modelo computacional e com certo grau de formalidade, que possa ser usado pelos desenvolvedores. Não há, a princípio, uma guia que indique a adequação da automação ou que novos resultados podem ser obtidos. O usuário, por não ter acesso a sistemas de informação que executem a mesma atividade, tem pouco conhecimento sobre o que é possível fazer, ou não tem ideia de qual o custo de produzir certos resultados.

Já os projetos de redesenvolvimento apresentam a vantagem de possuir uma base que pode ser utilizado como referência do que deve ser feito (repetido), do que não deve ser mantido (eliminações) e das novas atividades necessárias. O usuário, acostumado e experiente com um sistema existente, pode fornecer informações mais adequadas sobre o que espera do novo sistema, ou da manutenção ou melhoria sendo feita. Há um risco, porém. Quando o sistema antigo ainda funciona, muitos projetos de redesenvolvimento acabam falhando, ou por não atingir claramente a mesma qualidade ou funcionalidade do sistema anterior, por alguma percepção de risco que exista sobre o uso da nova versão, ou ainda por questões políticas da organização, como pressão exercida pelos responsáveis do sistema ainda rodando.

## 9.4. Porque são feitos projetos de SI

Muitos são os motivos que influenciam o início de um projeto de desenvolvimento de um Sistemas de Informação. Em geral, usando um raciocínio econômico, podemos dizer que um projeto é iniciado quando o benefício do retorno esperado supera o custo do projeto . O problema é que não é fácil converter esses valores em números normalmente.

### 9.4.1. Benefícios do Sistema

Vários motivos podem ser analisados como benefícios esperados de um projeto. O principal benefício que um sistema de informação pode oferecer é melhorar significativamente o negócio do usuário, aumentando seu lucro. Porém, essa não é a única motivação possível.

Uma motivação comum é modernização de um sistema. Com o tempo a tecnologia de um sistema vai se tornando superada. Isso faz com que o risco e o custo de manter o sistema funcionando naquela tecnologia aumentem, aumentando gradativamente o interesse de se transportar o sistema para uma nova plataforma. Simultaneamente, novas tecnologias apresentam novas oportunidades, como desempenho superior ou facilidade de

## 9. Sistemas de Informação

aprendizado, aumentando também com o tempo o interesse nessa atualização. Chega um momento então que passa a valer a pena o investimento em modernização.

Outro motivo importante é a mudança de premissas básicas do negócio, causada pela atuação da firma no mercado. Essas mudanças tanto podem de dentro da empresa quanto podem ser provocadas por mudanças na legislação ou por ação dos concorrentes. Por exemplo, há alguns anos atrás, no Brasil que convivia com inflações altíssimas, foram feitas várias modificações nos sistemas financeiros das empresas para aceitar a mudança de moeda do país e o convívio com moedas diferentes simultaneamente. Em outro exemplo, com a invenção e grande aceitação dos sistemas de premiação por viagens ou por milhas, as companhias aéreas tiveram que desenvolver sistemas específicos, interagindo com seus sistemas de passagens, para tratar do assunto. Muito comum também é a mudança de uma atividade da empresa, seja por um processo contínuo, como o de qualidade total, quanto por processos radicais como os de reengenharia e *downsizing*.

Os sistemas de informação também são importantes por oferecerem as empresas uma capacidade maior de competição. Com a informação correta e com os processos corretos de tratamento da informação uma empresa pode ter um diferencial de qualidade no mercado. Por outro lado, se todo um mercado já adotou um tipo de sistema, ou se pelo menos um concorrente já o fez, a empresa que não tem um sistema equivalente fica prejudicada na competição. Esse tipo de efeito foi visto quando as companhias aéreas passaram a vender passagens via Internet. No início era mais uma propaganda, depois passou a ser um diferencial positivo. Atualmente todas as companhias aéreas possuem formas de vender passagens diretamente via Internet, sendo uma funcionalidade obrigatória para as empresas.

Hoje em dia um grande motivador de novos projetos e a busca por melhor tratamento da informação que já existe em sistemas de tipo operacional, como a criação de Sistemas de Suporte Executivo. Nesse caso, o dado básico do sistema já está quase que totalmente mapeado e o que ocorreram são consultas e transformações. Isso é tão comum que novas áreas de T.I. apareceram ao redor desses sistemas, identificadas por *buzzwords* como *Data Warehouse* e *Business Intelligence*.

### 9.5. O Poder está com o usuário

Um dos acontecimentos mais marcantes da computação é a transferência do poder daqueles que operavam as máquinas, os famosos e muitas vezes odiados CPDs – os Centros de Processamento de Dados – para o usuário final.

Para aqueles que só chegaram ao mundo da informática agora, ou para aqueles que nasceram após a revolução causada pelos microcomputadores, é muitas vezes difícil de entender a complexidade e a mística que cercavam os grandes computadores. Resfriados a água, mantidos em salas seguras, gastando espaço e energia, esses computadores da década de 1970 tinham o poder computacional por vezes menor que um telefone celular do início do século XXI. Eram esses computadores, porém, que mantinham os dados

fluindo, as contas e salários sendo pagos, à custa de uma vigilância permanente de seu funcionamento e do uso de recursos. Até hoje, muitos serviços críticos funcionam em versões modernizadas desses computadores, geralmente a custos altos, por causa do alto risco de transferência para outras tecnologias .

Parte da transferência de poder aconteceu quando os microcomputadores chegaram em grande quantidade as empresas, permitindo que os usuários que não eram atendidos no prazo e na qualidade que esperavam, tomassem a rédea do processo de software, desenvolvendo seus próprios aplicativas, usando planilhas eletrônicas e sistemas simples de banco de dados (como dBase II e Access) e, muitas vezes, passando por cima da estrutura da própria organização e contratando soluções terceirizadas, já que não precisavam do computador central para executá-las.

Isso alterou drasticamente a estrutura de poder das organizações, que eram fortemente dependentes dos dados processados de forma central, em grandes máquinas, com software de ciclo de desenvolvimento muito longo. O processo de mudança não poupará carreiras, sendo que algumas empresas simplesmente fecharam seus CPDs, terceirizando suas atividades e despedindo ou transferindo todos seus funcionários.

Hoje em dia o próprio nome CPD é estigmatizado. Cabe agora ao setor de TI – tecnologia da informação – manter uma estrutura muito mais complexa que a anterior, unificando sistemas de várias gerações, em redes com grande quantidade de computadores executando sistemas operacionais diferentes e aplicações diferentes. Ainda existem conflitos entre o pessoal de TI e as outras partes da empresa, mas o foco cada vez mais é melhorar o negócio e atender melhor os usuários.

## 9.6. Exercícios

**Exercício 9.1:** Vá para o site <http://jogodeanalisedesistemas.xexeo.net/> e visite a Livraria Resolve. Descreva que sistemas de informação a livraria precisaria ter para funcionar.



# **Parte III.**

## **Data Warehouse**



# 10

## A Situação das Informações nas Organizações

### 10.1. A necessidade de informação das organizações

Todas as organizações têm a necessidade de controlar e gerenciar todas as informações que dispõem sobre si mesmas e sobre o mercado, de modo a permitir que suas decisões produzam ações que gerem benefícios e evitem prejuízos, de curto, médio e longo prazo, nos níveis operacional, tático e estratégico.

As informações necessárias para a organização aparecem em todo seu ambiente. Cada ação realizada, cada decisão tomada, gera, ou pelo menos deveria gerar, um registro que pode ser utilizado mais tarde para a tomada de decisões. Por exemplo, cada venda feita por uma cadeia de lojas de varejo pode ser incluída em um gráfico ou relatório que permita entender o desempenho das vendas por local, por produto, por vendedor e até mesmo pela hora do dia. Em um uso mais avançado, esses dados podem ser correlacionados, com o processo de seleção da área de Recursos Humanos, de maneira a tentar desenvolver um perfil do vendedor a ser contratado, tanto de forma geral como para atender necessidades específicas. Com relação aos dados de mercado, podem ser usados para prever o desempenho de uma nova loja em uma localidade específica, de acordo com dados demográficos ou informações sobre outros tipos de comércio ou mesmo de consumo de água ou eletricidade.

Para isso ser possível é necessário informatizar a organização, isto é, implantar dentro da organização sistemas de informação que gerenciem essas informações.

## 10.2. Processo de Informatização nas Organizações

Frente a importância da informação para o sucesso da organização, é fácil entender que, ao longo de sua vida, ela deve passar por um processo contínuo de informatização, a fim de atender as demandas cada vez maiores tanto da própria organização, quanto de seus fornecedores, parceiros, clientes e também do Estado.

Esse processo é necessário e geralmente benéfico, mas não sem problemas.

Entre os principais problemas do processo de informatização das organizações é que ele normalmente é de crescimento vegetativo, *bottom-up* e por demanda imediata. Os sistemas são criados um a um, em função de necessidades específicas trazidas pelas partes interessadas, e construídos de modo a atender principalmente os requisitos dessas partes, muitas vezes desconsiderando como as atividades e informações desses sistemas podem ser úteis para o resto da organização.

Por outro lado, tentativas de criar grandes sistemas integrados que atendam toda a organização parecem estar fadados ao fracasso. O tamanho de um sistema é um fortíssimo fator de risco(R. Pressman e Maxim, 2016), já que a relação do tamanho com a taxa de fracasso é mais que linear.

Como sempre, as melhores soluções, ou pelos menos as soluções mais viáveis, parecem estar em um meio termo, ou pelo menos em decisões tomadas de forma consciente dos riscos e benefícios de cada opção.

Não se pode também deixar de levar em consideração que organizações diferentes têm gestões diferentes desse processo. Enquanto uma pode escolher os sistemas de informação a serem implantados de forma *ad-hoc*, outra, mais corretamente, pode possuir um planejamento estratégico que indica em que direção a área de Tecnologia da Informação deve seguir e que sistemas serão estratégicos para o seu progresso.

Esse quadro bastante variado, evoluiu muito tecnologicamente e no entendimento do negócio informatizado nos últimos anos. Alguns grandes marcos, como o aparecimento do computador comercial, o aparecimento do computador pessoal, o aparecimento das redes locais e finalmente o aparecimento da Internet e das tecnologias em nuvem, causaram grandes mudanças na forma de pensar a organização, criando até mesmo a possibilidade de organizações virtuais.

Ao logo desse tempo houve um aprendizado que trouxe soluções bastante adequadas ao problema de gestão das organizações, como o uso de sistemas **ERP** e a compra de software **COTS**.

**COTS** significa **Commercial Of The Shelf**, e indica software pronto que fornece funções específicas com nenhuma ou pouca adaptação para quem o compra ou implementa. Principalmente pequenos negócios e profissionais liberais podem ter quase todas suas necessidades atendidas por meio de esse tipo de software. Algumas áreas de aplicação, mesmo em empresas grandes, podem se beneficiar desse tipo de produto. Deve ficar

### *10.3. A necessidade centralização dos dados*

claro, porém, que nesse caso a organização deve se adaptar as práticas implementadas no produto.

Já os sistemas **ERP, Entreprise Resource Planning**, são na prática sistemas de gestão empresarial customizáveis destinados a integrar fortemente as áreas de negócio. Sua principal característica é funcionar em áreas do negócio cuja tarefa é bem definida, e cuja experiência de desenvolvimento de software é muito forte, e seu maior sucesso vem da implementação, já no software, das melhores práticas do negócio.

Soluções ERP hoje são capazes de servir grande parte das necessidades genéricas de todas as empresas, desde funções de Recursos Humanos até detalhes de Contabilidade, sendo que algumas empresas inclusive mudam seus processos para se adaptar mais facilmente a um ERP específico.

Essas soluções, porém, são genéricas. Mesmo quando customizáveis, dificilmente atendem necessidades importantes dos processos das cadeias de valor das empresas, pois estes são específicos e poucas empresas desejam remodelá-los de acordo com uma prática comum no mercado, até mesmo porque suas diferenças podem fornecer justamente o diferencial competitivo que possuem. Assim, o software desenvolvido sob encomenda, interna ou externamente, ainda possui, e certamente sempre possuirá, um presença marcante em todos as organizações.

Todos esses sistemas geram dados. Quase que escondidos nesses dados estão as informações necessárias para a tomada de decisão.

## **10.3. A necessidade centralização dos dados**

O outro processo de grande sucesso foi a centralização dos dados dos sistemas operacionais e transacionais (OLTP) em bases de dados integradas na empresa, principalmente em SGDBs Relacionais de grande porte.

O quadro do primeiro quarto do século XXI é que a maioria das organizações entende que seus dados operacionais devem estar sob controle centralizado. O ideal é que todos esses sistemas compartilhem uma mesma base, porém necessidades como desempenho, custo de implementação, estratégia empresarial, permanência da tecnologia usada no mercado, e ainda outras, podem levar a existência de mais de uma base.

Esta situação também não é sem seus problemas. A centralização dos dados sob um controle único pode tanto auxiliar como trazer dificuldades para o desenvolvimento rápido de sistemas, principalmente quando há exigência de muita burocacia para alterar a base ou estruturar informações de forma a atender novas demandas de negócio.

O uso de softwares COTS ou ERP integrados, chave do sucesso para parte da operação da empresa, pode criar uma forte dependência da organização com um vendedor, de forma que seja difícil fazer decisões estratégicas de mudança de produto sem um grande

## 10. A Situação das Informações nas Organizações

custo para a organização, ou simplesmente que seja difícil extrair os dados desses sistemas para atender outras demandas.

Em todo caso, antes da proposta dos Data Warehouse a situação das informações dentro das organizações podia ser descrita, de forma geral, da seguinte maneira:

- Reconhecimento da necessidade de centralização dos dados operacionais, nas aplicações do tipo OLTP, normalmente na forma do uso de uma base de dados unificada, ou de várias bases controlados por um serviço central.
- Existência de vários sistemas de informação, centralizados ou não. Os sistemas não centralizados existem tanto por motivos históricos, no caso de software legado, quanto de aplicações que foram desenvolvidas fora do sistema padrão por diversos motivos: desenvolvimento distribuído nas partes da organização, necessidade de passar por cima de processos burocráticos para atender demandas imediatas de negócio, necessidade de uso de tecnologias específicas, etc.
- Existência de uma grande quantidade de sistemas criados pelos próprios usuários acima do nível operacional, muitas vezes na forma de planilhas eletrônicas, documentos ou mesmo bancos de dados simples, como o Microsoft Access.
- Pouca integração dos dados e dos sistemas destinado ao apoio da decisão, geralmente criados de forma *ad-hoc* e possivelmente a partir de retratos diferentes da informação disponível na organização.

Esse últimos dois fatores levaram a um quadro caótico nos dados destinados a tomada de decisão dentro da organização. Enquanto os dados operacionais eram mantidos dentro de algum controle, especialmente porque são essenciais para o funcionamento da organização, os dados usados para a tomada de decisão, geralmente sumários ou recortes dos dados transacionais, eram baseados em fotografias tiradas de forma diferente e em momentos diferentes da empresa. Esse é o principal diagnóstico que leva a necessidade da existência de Data Warehouses por Inmon (2005): a falta de organização e o excesso de redundância e incertezas nesses dados.

Um exemplo do que poderia acontecer dentro da organização por causa desta situação caótica são dois gerentes ou diretores chegarem a uma reunião com números diferentes sobre um mesmo objetivo da empresa, devido a formas como esses dados foram obtidos. Um diretor, por exemplo, com os dados de venda por semestre, até o último semestre completo, poderia dizer que as vendas estavam aumentando, enquanto um segundo diretor, com os dados de vendas por mês poderia dizer que estavam diminuindo. Dependendo da fonte da informação, a confusão podia ser maior. Um diretor, por exemplo, poderia estar contabilizando os pedidos, enquanto outro os pagamentos feitos. Ainda, um diretor poderia ter usado dados obtidos com algum viés de seleção, como desconsiderar um canal, ou considerar apenas as vendas para os grandes clientes.

A verdade é que se os dados são muitas vezes extraídos dos sistemas OLTP de forma arbitrária e depois trabalhados manualmente pelos membros da organização. A tendência é que se espalhem, sejam alterados em formato e conteúdo e criem uma miríade de fotografias que não só mostram realidades diferentes da empresa, mas que também foram

#### *10.4. Momento Histórico da Proposta dos Data Warehouse*

tiradas de forma diferente, com motivações diferentes e que não podem ser mapeadas ao processo original de extração.

Para dar um ideia do problema do espalhamento de dados em um organização, que ocorre tipicamente por planilhas eletrônicas, uma revisão de diferentes pesquisas sobre erros em planilhas de 1995 a 2008 calculou erros em 88% das planilhas no total, sendo que algumas dessas pesquisas encontrou erros em 100% das planilhas investigadas Panko (1998). Soto (2019) relata erros específicos em planilhas que causaram prejuízos a organizações, como a venda de 10.000 ingressos não existentes para os Jogos Olímpicos de Londres de 2012, obrigando o Comitê a substitui-los por outros ingressos de maior valor.

Todo esse quadro anteriormente exposto mostra que apesar dos enormes esforços e algo custo da gestão das informações nas empresas, mesmo em 2019, na prática essa informação ainda é muito carente de organização e controle.

Mesmo depois de todo o aprendizado que foi obtido, de todas as propostas feitas e adotadas, a verdade é que nas organizações do mundo real acaba levando a soluções não ideais. O resultado é que organização acaba por tomar decisões importantes sobre dados com pouca qualidade.

## **10.4. Momento Histórico da Proposta dos Data Warehouse**

Em outra escala, o problema do descontrole dos dados já ocorria na década de 1970, onde começaram a aparecer os primeiros sistemas de informação automatizados.

A partir de experiência em trabalhos semelhantes ao que hoje é chamado de Data Warehouse, em 1992 Inmon (1992) lançou seu primeiro livro sobre o assunto, chamando a atenção de todos para a necessidade de aumentar a qualidade dessa dado, na forma do que ele denominou de Data Warehouse. Nessa época, os microcomputadores, as planilhas e os sistemas criados diretamente pelo usuário já eram parte integradas da Tecnologia da Informação nas organizações. Porém, ainda não tinha sido atingida a fase do **Big Data**, e tantas outras tecnologias comuns hoje em dia não estavam disponíveis.

Mesmo assim, o retrato apresentado por Inmon (1992) e renovado em Inmon (2005) pouco difere do retrato que vemos agora. Se algo aconteceu foi o aparecimento de mais fontes de dados e de demandas cada vez maiores de tratar informação.

## **10.5. O Data Warehouse**

Então, da mesma forma que as bases relacionais propiciam a integração dos dados transacionais, Inmon (2005) e outros perceberam que era necessário **centralizar** os dados

## 10. A Situação das Informações nas Organizações

destinados a tomada de decisão, normalmente por meio de sistemas OLAP. Mais que isso, que esses dados não deveriam estar registrados da mesma forma dos sistemas OLTP, pois eles refletem uma outra visão, que é composta de dados históricos, persistentes organizados pelo assunto, e com outras características que vão definir um **Data Warehouse**.

De certa forma, o Data Warehouse, como conceito, está para os sistemas OLAP e de tomada de decisão assim como as Bases de Dados Integradas estão para os sistemas OLTP e operacionais. Trazer toda a informação para um ambiente controlado, sanitizado e centralizado é uma virtude a ser perseguida.

Para ser implementado e ser útil, porém, o Data Warehouse precisa ser adaptado ao fato que a base integrada organizacional é apenas uma utopia. Por isso, a implantação de um Data Warehouse também precisa estabelecer uma metodologia de coleta e transformação dos dados. Para isso ele tem que ter acesso as várias fontes de informação primitiva da empresa e a capacidade de transformá-la e armazená-la de forma a atender, de maneira eficiente, as demandas dos sistemas OLAP e de tomada de decisão. Isso vai levar a existência de uma estrutura complexa que inclui mecanismos que usam nomes como *ETL*, *ELT* e *Staging*, tratados neste livro.

O Data Warehouse também tem que ser separado dos sistemas OLTP por questões de desempenho. A forma de acesso as operações diárias gera uma carga no sistema, e exigências de tempo de atendimento, diferente das operações realizadas em um Data Warehouse (Inmon, 2005). A Figura 10.1 mostra a diferença do comportamento da carga desses dois sistemas.

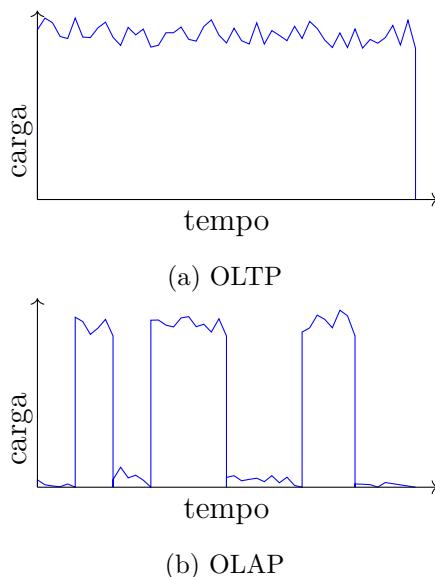


Figura 10.1.: Diferentes cargas exigidas de sistemas OLTP e OLAP. Fonte: (Inmon, 2005)

Finalmente, para corretamente distribuir os dados a quem precisa e pode consultá-los, levando em consideração fatores tanto de desempenho quanto como leis como a

Sarbanes–Oxley(United States Code, 2002), que fala do controle do acesso à informação dentro da organização, o Data Warehouse precisa alimentar bases menores conhecidas como **Data Marts**.

## 10.6. Data Warehouse e Bancos de Dados

O mercado atual usa o termo Banco de Dados, majoritariamente querendo dizer que é relacional, para indicar o repositório de dados da organização, em especial o repositório de dados OLTP, visando o rápido acesso a dados únicos, para acelerar essas transações. É possível e desejado fazer relatórios e gráficos a partir dos dados nos bancos de dados, porém eles normalmente refletem o estado atual da organização.

O termo Data Warehouse é reservado para indicar o repositórios de dados históricos da organização, visando o rápido acesso a visões agregadas, como relatórios, e servir operação OLAP.

Outra diferença básica é que o modelo de dados de um Banco de Dados é criado para permitir o uso mais geral possível dos dados, e segue regras, conhecidas como normalização, para evitar redundâncias. Já o Data Warehouse é criado para atender necessidades específicas, com um modelo muitas vezes não normalizado.

Porém, há um erro em tentar criar essa dicotomia. A verdade é que um Data warehouse precisa de um sistema de banco de dados para funcionar, ou seja, os Data Warehouse tem que existir dentro de um sistema de informação com as mesmas características dos sistemas de banco de dados. Devido a existência de SGBD relacionais de alto desempenho, que com o tempo foram adaptados para atender melhor as demandas de desempenho de um Data Warehouse, grande parte dos Data Warehouse funciona dentro de sistemas relacionais.

Existem porém outras bases não relacionais muito utilizadas para Data Warehouse. Normalmente elas se caracterizam por serem otimizadas para certo tipo de consultas, normalmente operações sumários em uma tabela e operações OLAP, e no uso do Modelo Dimensional, que será tratado mais tarde neste texto.

## 10.7. Data Warehouses ainda são necessárias?

Tendo em vista que Data Warehouses foram projetadas no final do século XX, é possível questionar sua aplicabilidade quase 30 anos depois de seu aparecimento.

Uma questão que se põe é causada pelo avanço da tecnologia. A disponibilidade de armazenar um volume muito maior em discos e o uso de múltiplas CPUs nos servidores permite que sejam construídos sistemas de informação OLTP que pouco se preocupam com a necessidade de limpar periodicamente os dados antigos do sistema. Esses dados podem então ser acessados diretamente?

## 10. A Situação das Informações nas Organizações

A resposta para essa questão está no fato que sistemas OLTP e OLAP, e também as novas aplicações de **Business Intelligence (BI)**, trazem requisitos diferentes, que podem ser atendidos exatamente pelos Data Warehouses.

Por outro lado, sistemas de *Big Data* e *Data Lake* podem parecer concorrentes ao sistemas de Data Warehouse, quando de fato são complementares e acabam criando uma sinergia.

### 10.8. Mais de um Data Warehouse?

Apesar da ideia básica do Data Warehouse é ser uma base central de informações, como nos casos dos sistemas de informação tradicionais, fatores do dia a dia fizeram que o sonho do Data Warehouse centralizado fosse substituído, no mercado, pela existência, dentro da organização, de mais de um Data Warehouse. Uma pesquisa descrita por D. Wells e Nahari (2019) mostrou que a maioria das empresas usa mais de um Data Warehouse, de acordo com os dados apresentados na Figura 10.2.

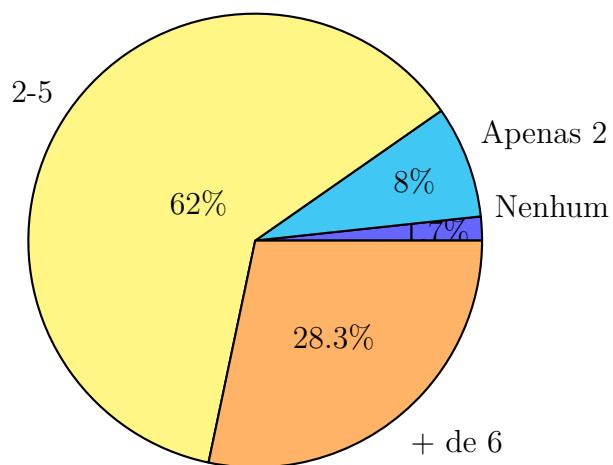


Figura 10.2.: Número de data warehouses por empresa(D. Wells e Nahari, 2019).

# 11

## Data Warehouse

### 11.1. O Debate Kimball x Inmon

Dois autores dominam as publicações sobre Data Warehouse no mundo.

O primeiro deles é **Bill Inmon**, considerado o pai do Data Warehouse, que começou a discutir o assunto e cunhou o termo ainda em 1970. Seu livro *Building the Data Warehouse*, em sua primeira edição de 1992, foi um marco importante na divulgação do tema(Kempe e Williams, 2012).

O segundo autor é **Ralph Kimball**. que com a primeira edição de *The Data Warehouse Toolkit*, de 1996, trouxe um conjunto de práticas e exemplos que facilitaram a implementação de data warehouses para muitos(Kempe e Williams, 2012).

Inmon e Kimball mostram caminhos diferentes de implementar data warehouses, sendo que Inmon favorece a abordagem *top-down*, baseada em um modelo normalizado, e Kimball a abordagem *bottom-up*, baseada em um modelo dimensional, de criação de data warehouses. A polêmica entre suas ideias é uma das mais conhecidas da Informática.

Em especial, Ralph Kimball (2013) apresentou ao público e defende o uso do do Modelo Dimensional<sup>1</sup>. Ele diz: “*Data in the queryable presentation area of the data warehouse must be dimensional*”(Ralph Kimball, 2013). Já Inmon (2005) diz “*Star schemas are not very good in the long run for a data warehouse.*” Esquemas Estrela são a forma de implementar modelos dimensionais em Bancos de Dados Relacionais.

---

<sup>1</sup>O Modelo Dimensional e os termos Fato e Dimensão foram inventados nos anos 1960 por um projeto conjunto da *Dartmouth University* e da *General Mills*(Ralph Kimball, 2013)

## 11.2. Definição de Inmon para Data Warehouse

Segundo Inmon (2005), um **Data Warehouse** é “Uma coleção de dados orientada a assunto, integrada, não volátil e variante no tempo que apoia as decisões de gerência.”

### 11.2.1. Orientação a Assunto

A ideia de Inmon é que enquanto os sistemas tradicionais são organizados ao redor das funções da empresa, ou seja, do que a empresa faz, os data warehouses são organizados em torno dos **assuntos** que a empresa trata, ou seja, dos tópicos sobre as quais ela tem que tomar decisão.

Por exemplo, o que faz um mercado de varejo? Em sua cadeia de valor, Figura 11.1, ele compra no atacado, estoca e vende produtos no varejo. Os seus sistemas de informação são orientados para essas atividades. Uma lista de sistemas de informação que podem estar disponíveis em um pequeno mercado é mostrada na Tabela 11.1.

Tabela 11.1.: Sistemas em um pequeno mercado.

#### Sistemas do Mercado

- Controle de Estoque
- Contas a Pagar/Receber
- Controle do Fiado
- Caixa Registradora
- Pedidos por Telefone
- Entregas

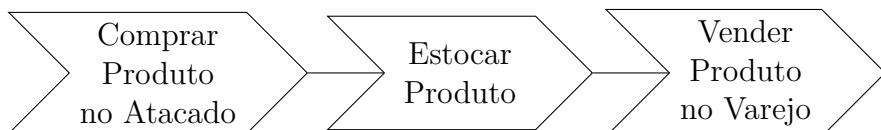


Figura 11.1.: A cadeia de valor de um mercado, descrição em ARIS.

Porém, como o mercado analisa suas informações por assuntos? Por exemplo, com perguntas como “vendas por produto” ou “vendas por loja”. O assunto, nesse caso, é **vendas**, e a informação principal é um agregado de várias instâncias do banco de dados que registram um tipo de venda.

Nas fontes originais de dados dentro da empresa, a informação referente a um assunto pode estar dividida de várias formas. Por exemplo, para o assunto cliente podemos ter seu nome e CPF em um registro de uma venda, porém seu endereço só estará em um registro de uma entrega. Já seu telefone pode estar em um registro de um pedido feito pelo telefone para entrega. Todas essas ações são atendidas por um ou mais sistemas de informação da empresa, e os dados podem estar em um ou mais repositórios, e mesmo

## 11.2. Definição de Inmon para Data Warehouse

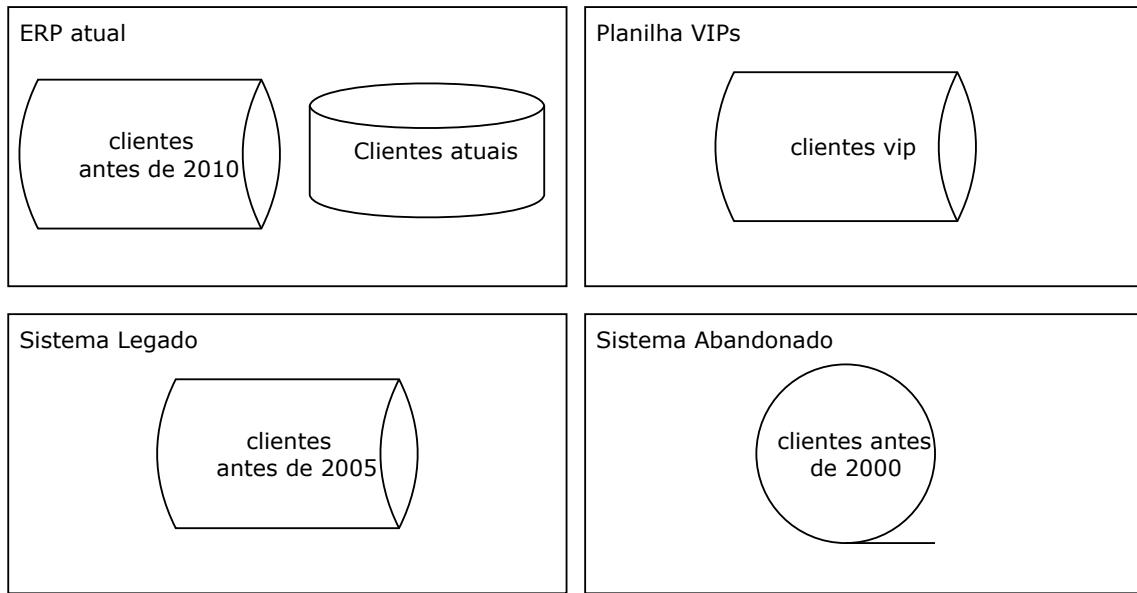


Figura 11.2.: Diversas fontes de onde pode ser obtido o assunto cliente

dentro de tabelas diferentes em uma base de dados única, dependendo da maturidade da empresa no tratamento de seus dados.

Além disso, tanto os sistemas são trocados e evoluem com o tempo, quanto os dados e seus formatos.

Por exemplo, para o assunto cliente, podemos ter dados dos clientes atuais em um SGDB, dados de clientes antigos guardados em fitas backup de sistemas legados, dados de clientes de um setor específico guardados em uma planilha, etc.

Podemos também ter dados repetidos de um mesmo cliente, e com algumas alterações. Um cliente que mudou de endereço, ou mesmo uma cliente que trocou de nome ao casar (Inmon, 2005). A Figura 11.2 mostra um exemplo de fontes possíveis sobre clientes.

Cada assunto define um conjunto de tabelas correlacionadas (Inmon, 2005). No modelo dimensional (Ralph Kimball, 2013), cada assunto vai definir um (ou mais) modelos compostos de uma tabela central, que indica o tópico consultado, a **tabela fato**, e tabelas auxiliares que mostram como a tabela fato pode ser consultada, as **tabelas dimensão**.

É importante lembrar que quando trazidas para o data warehouse, todas essas informações sobre o cliente tem que ser processadas de forma que um mesmo cliente seja sempre identificado por uma mesma chave.



# 12

## Componentes do Data Warehouse

Antes de discutir os componentes do Data Warehouse, ou do Data Warehousing, como a atividade geral, é importante notar que esse termo faz parte de uma sequência de termos que foram usados como “códigos” no mercado para vender produtos e serviços que atendem a necessidade da empresa de captar, organizar, armazenar, processar e analisar dados, de forma a cumprir suas funções e, principalmente, tomar decisões de forma correta.

Assim, Data Warehouse é apenas um mais nome na lista que inclui termos como Business Intelligence, Information Lifecycle Management, Big Data, Data Mining, Data Analysis, Data Science, Data Warehouse 2.0, Data Lake, Actionable Analytics, outras que foram esquecidas e mais que vão aparecer. Isso implica que há grande interseção entre essas áreas, e que uma área pode colocar a outra como “menor” e aparecendo dentro dela.

Em todo caso, podemos ver quatro grandes componentes nos sistemas de Data Warehouse(Ralph Kimball, 2013):

- os sistemas fonte, que possuem os dados a serem colocados no Data Warehouse;
- a Data Staging Area, onde trabalhamos os dados que vem dos sistemas fonte, e os preparamos para o Data Warehouse;
- o Data Warehouse propriamente dito, e os Data Marts associados, e
- os sistemas que usam os Data Warehouse e Data Marts.

Os três primeiros desses quatro componentes se comunicam por meio de um processo que trata da maior parte dos problemas do Data Warehouse: o ETL, a sigla para Extração, de dados dos sistemas fontes, Transformação e Carga, dos dados nos DMs e DWs. Mais recentemente, uma alternativa ao ETL é o ELT, onde a transformação é feita já dentro do DW.

## 12. Componentes do Data Warehouse

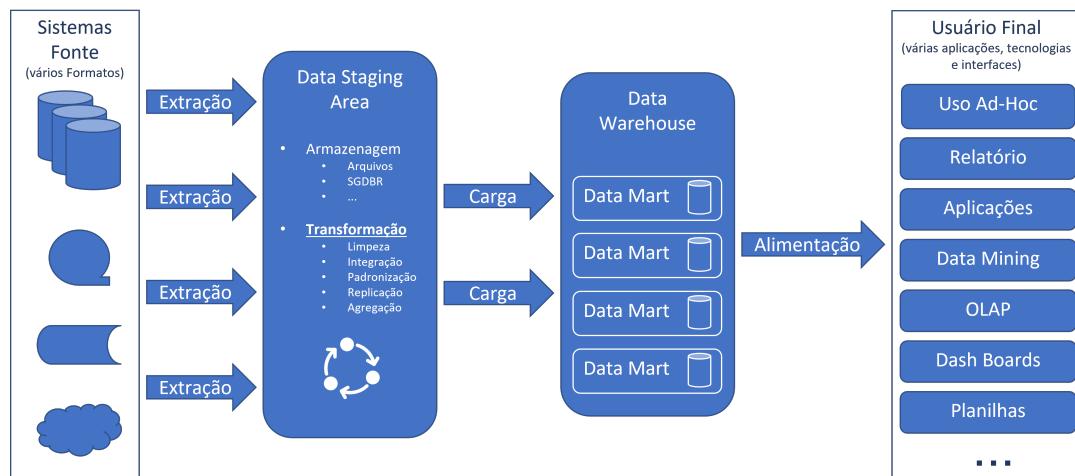


Figura 12.1.: Componentes do Data Warehouse. Inspirada em Ralph Kimball (2013)

### 12.1. Sistemas Fonte

Qualquer sistema usado na empresa, ou mesmo outros dados que estão disponíveis internamente, como logs de web, ou externamente, como dados públicos ou produzidos por alguma organização, podem ser usados para obter os dados a serem organizados no DW.

Esses sistemas são chamados **sistemas fonte**, por serem a fonte original dos dados.

Em geral, os sistemas fonte são bastante heterogêneos. Existem diferenças na construção lógica do entendimento dos dados, na implementação física dos repositórios de dados e nos dados propriamente dito.

Por exemplo, em uma organização qualquer, os dados de vendas, por exemplo os gerados por emissão de nota fiscal, podem estar guardados em um banco de dados Microsoft SQLServer rodando em um sistema operacional Windows, sendo usado para isso um sistema comprado no mercado, onde os campos possuem nomes na forma de código que não contém semântica (XXFIELD10, por exemplo), e os produtos são identificados por seu número EAN. Já no sistema de produção, sendo um fábrica, é usado um banco de MariaDB, executando em Linux. O primeiro banco usa uma codificação ASCII de 7-bits para seus caracteres, o segundo a codificação UTF-8. Um produto, no segundo banco, é representado por seu número de fabricação, que ainda não foi integrado com o EAN.

Essas diferenças terão que ser resolvidas no processo de ETL - Extração, Transformação e Carga, ou ELT, de forma que o DW seja integrado. Isto é, levando em conta que o produto vendido é o menos que o produto fabricado, é importante registrar isso no modelo dimensional de alguma forma.

## **12.2. Data Staging Area**

Esse componente do Data Warehouse é ao mesmo tempo um grande rascunho no qual o dado é trabalhado, como também é um espaço onde são armazenados dados que são importantes para a continuidade do processo de ETL.

## **12.3. Data Warehouse e Data Marts**

## **12.4. Sistemas que usam o Data Warehouse**

## **12.5. O Processo de ETL**



# 13

## OLAP

**OLAP** é a sigla para **Online Analytical Processing**. O termo é usado para diferenciar o tipo de sistema, ou de operações, que se usa quando o objetivo é analisar os dados de uma organização, normalmente para a tomada de decisão, do uso normal dos dados dos sistemas que suportam o funcionamento da organização, conhecidos como transacionais ou **Online Transaction Processing**, **OLTP**, ou ainda **processamento de transações em tempo real**.

Assim, sistemas OLTP e sistemas OLAP, dentro de uma organização, são normalmente usados em momentos diferentes, por usuários diferentes e com objetivos diferentes. Além disso, guardam tipos de dados diferentes.

Sistemas OLAP são tipicamente construídos sobre data warehouses, mas não necessariamente. Além disso, eles admitem uma série de operações, conhecidas como **operações OLAP**, que permitem navegar e analisar os dados, normalmente por meio de um programa de computador com interface de usuário bastante dinâmica.

As principais operações OLAP são o **slice and dice** e o **drill down and roll up**. Além disso são também conhecidas as operações **pivot** e **drill across**.

Para ilustrar algumas operações, neste capítulo serão usados como dados exemplos uma planilha com os micro-dados do ENEM 1998<sup>1</sup>(INEP, 2016) dentro do software Microsoft Excel™(Microsoft, 2019).

---

<sup>1</sup>Esta planilha tem poucos candidatos e poucas colunas

### 13. OLAP

The screenshot shows a Microsoft Excel spreadsheet titled "MICRODADOS\_ENEM\_1998.csv". The spreadsheet contains data from the 1998 Brazilian National Exam (ENEM) microdata. The columns represent various demographic and academic variables. The first few columns include NU\_INSCRICAO, NU\_ANO, NUIDADE, TP\_SEXO, CO\_MUNICIPIO\_RESIDE, NO\_MUNICIPIO\_RESIDE, CO\_UF\_RESIDENCIAL, SG\_UF\_I, TP\_PRES, CO\_PRC, and VL\_PE. The data spans from row 1 to approximately row 37. A green rectangular selection highlights a specific cell in the 15th row, column E. The Excel ribbon at the top shows the "Home" tab selected. The status bar at the bottom right indicates a zoom level of 70%.

NU_INSCRICAO	NU_ANO	NUIDADE	TP_SEXO	CO_MUNICIPIO_RESIDE	NO_MUNICIPIO_RESIDE	CO_UF_RESIDENCIAL	SG_UF_I	TP_PRES	CO_PRC	VL_PE
1	1	1998	19 F	2408003	MOSSORÓ	24	RN	1 B	43.3	3
2	2	1998	17 F	2401453	BARAÚNA	24	RN	1 B	23.3	3
3	3	1998	18 M	2408003	MOSSORÓ	24	RN	1 Z	46.7	3
4	4	1998	18 M	2408003	MOSSORÓ	24	RN	1 G	3	3
5	5	1998	21 M	2413102	SENAÍRIO ELOI DE SOUZA	24	RN	1 A	43.3	3
6	6	1998	17 F	2408102	NATAL	24	RN	1 G	56.7	3
7	7	1998	18 M	2408102	NATAL	24	RN	1 A	36.7	3
8	8	1998	17 M	2408102	NATAL	24	RN	0	0	3
9	9	1998	30 M	2408102	NATAL	24	RN	0	0	3
10	10	1998	19 M	2408102	NATAL	24	RN	0	0	3
11	11	1998	18 M	2412104	SAO JOAO DO SABUGUEIRO	24	RN	1 B	4	3
12	12	1998	18 F	2408102	NATAL	24	RN	0	0	3
13	13	1998	25 F	2408102	NATAL	24	RN	0	0	3
14	14	1998	19 F	2408102	NATAL	24	RN	1 B	3	3
15	15	1998	38 F	2408102	NATAL	24	RN	0	0	3
16	16	1998	21 M	2408102	NATAL	24	RN	0	0	3
17	17	1998	21 M	2408102	NATAL	24	RN	1 G	46.7	3
18	18	1998	23 F	2402006	CAICO	24	RN	1 G	3	3
19	19	1998	18 F	2408102	NATAL	24	RN	1 G	23.3	3
20	20	1998	17 F	2408102	NATAL	24	RN	1 B	66.7	3
21	21	1998	22 F	2408102	NATAL	24	RN	0	0	3
22	22	1998	17 M	2408102	NATAL	24	RN	1 Z	8	3
23	23	1998	23 F	2402006	CAICO	24	RN	1 A	2	3
24	24	1998	20 M	2408102	NATAL	24	RN	0	0	3
25	25	1998	17 F	2408102	NATAL	24	RN	0	0	3
26	26	1998	18 F	2408102	NATAL	24	RN	1 Z	4	3
27	27	1998	18 F	2408102	NATAL	24	RN	1 Z	33.3	3
28	28	1998	17 F	2408102	NATAL	24	RN	1 B	36.7	3
29	29	1998	22 F	2408102	NATAL	24	RN	1 A	36.7	3
30	30	1998	19 F	2403251	PARNAMIRIM	24	RN	0	0	3
31	31	1998	19 F	2408003	MOSSORÓ	24	RN	0	0	3
32	32	1998	18 F	2408003	MOSSORÓ	24	RN	1 G	3	3
33	33	1998	31 F	2408003	MOSSORÓ	24	RN	1 Z	23.3	3
34	34	1998	21 M	2408003	MOSSORÓ	24	RN	1 Z	2	3
35	35	1998	21 F	2408003	MOSSORÓ	24	RN	0	0	3
36	36	1998	21 F	2408003	MOSSORÓ	24	RN	1 G	26.7	3
37	37	1998	42 F	2408003	MOSSORÓ	24	RN	0	0	3

Figura 13.1.: Pequena fotografia de uma planilha com os micro-dados do ENEM 1998. Fonte (INEP, 2016).

## 13.1. O Cubo de Dados

Um metáfora usada para entender melhor as aplicações OLAP é a do Cubo de Dados. Usa-se um cubo porque é uma imagem 3D que pode ser entendida quando desenhada, e pode ser usada por algumas aplicações.

A ideia do cubo é estender para uma dimensão a mais o conceito de uma matriz de dados, para permitir visualizar melhor as operações OLAP.

Um cubo de dados mostra um conjunto de dados, equivalentes aos fatos de um Data Warehouse, de acordo com algumas dimensões. Por exemplo o cubo de dados OLAP da Figura 13.2 mostra a visão de dados sobre as vendas de uma rede de lojas de calçado. Em cada célula desse cubo está a informação desejada.

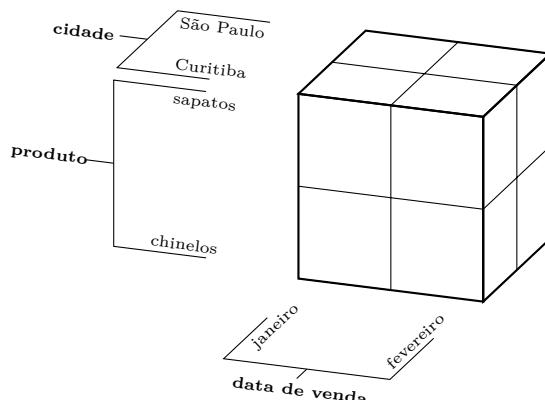


Figura 13.2.: Cubo OLAP simplificado para rede de lojas de calçado.

Uma representação simples do cubo de dados OLAP é a de uma tabela com várias páginas. Assim, as linhas da tabela indicam a primeira dimensão, as colunas a segunda, e a terceira dimensão é indicada pela página.

## 13.2. OLAP com Pivot Table no Excel™

As planilhas eletrônicas normalmente possuem uma funcionalidade que permite fazer facilmente uma análise de dados OLAP. Essa funcionalidade de análise é conhecida como **pivot table**.

Por exemplo, a Figura 13.1 mostra uma imagem com um pedaço de uma planilha Excel™ contendo uma única tabela com os micro-dados do ENEM de 1998(INEP, 2016).

A partir dessa tabela foi criada outra aba, contendo uma *pivot table*, que foi configurada para contar as presenças. Para isso foi usado o comando Insert → Table → Pivot Table, o que faz aparecer o diálogo da Figura 13.3. Normalmente basta apertar Ok.

### 13. OLAP

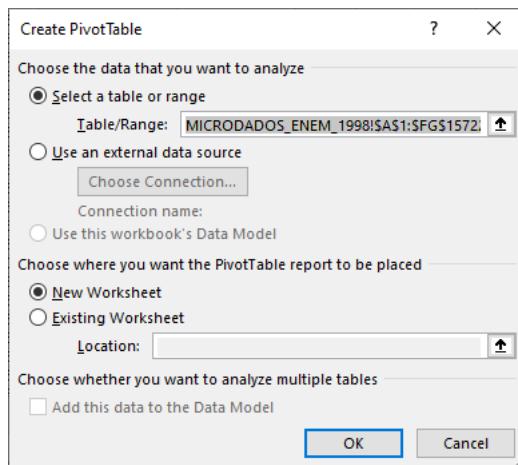


Figura 13.3.: Diálogo do Pivot Table do Microsoft Excel<sup>TM</sup>365.

A *pivot table* é controlada em duas áreas. Em uma, Pivot Table Field, detalhada na Figura 13.4 , é possível selecionar e configurar que campos serão mostrados de que forma, inclusive controlando fórmulas de cálculo.

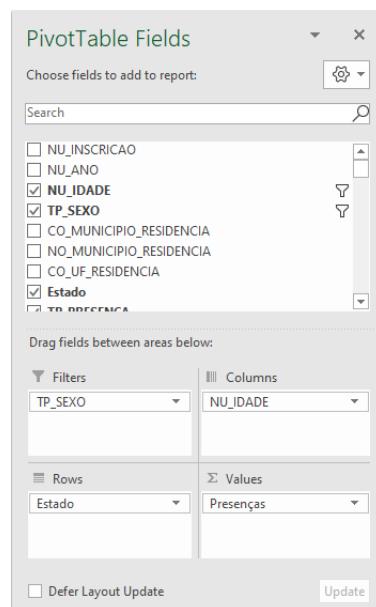


Figura 13.4.: Nesse quadro são controladas que colunas aparecem em que posição na pivot table

Na outra, que parece uma planilha comum, Figura 13.5 , acontecem visualizações dinâmicas do que é selecionado e ainda podem ser editados alguns nomes e usados filtros.

Analizando as Figuras 13.4 e 13.5 é possível ver que foram selecionadas algumas colunas para aparecer na planilha. É possível definir três dimensões para o Cubo, *Filters*, que indica uma espécie de página, *Columns*, que indica as colunas da tabela e *Rows*, que

The screenshot shows a PivotTable in Excel. The rows represent different states (AC, AL, AM, AP, BA, CE, DF, ES, GO, MA, MG, MS, MT, PA, PB, PE, PI, PR, RJ, RN, RO, RR) and the columns represent age groups (10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20). The values in the cells indicate the number of students present (1) or absent (0). The PivotTable is set up with 'TP\_SEXO' as the filter, 'Estado' as the row label, and 'NU\_IDADE' as the column label.

Figura 13.5.: Na planilha, a pivot table vai se alterando dinamicamente para atender a configuração dada no quadro de controle.

indicam as linhas da tabela. Mais tarde no capítulo serão usadas hierarquias dentro das dimensões.

No quadro *Filters* foi escolhido o campo “TP\_SEXO”, que indica o sexo do candidato e só tem duas opções “F” ou “M”. Para as linhas foi escolhido o campo “SG\_UF\_RSIDENCIA”, que foi renomeado “para Estado”. Para coluna foi escolhido o campo “NU\_IDADE”, que indica a idade do candidato. E para valor foi escolhido o campo “TP\_PRESENÇA”, também renomeado para “Presenças”.

Clicando em “Presenças” é possível escolher como vai ser feito o cálculo desse campo e foi escolhido a forma *Sum*, já que a presença é marcado com o número 1 e a ausência com o 0. Isso é mostrado na Figura 13.6.

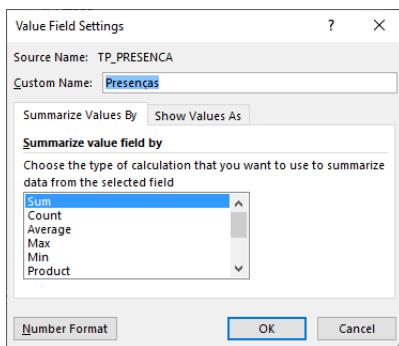


Figura 13.6.: Escolhendo “Sum” como função de agregação de dados para a presença dos candidatos.

### 13.3. Slice and Dice

A primeira operação OLAP, **slice and dice** é bem fácil de entender e é bastante conhecida em outras áreas pelo nome de **filtro**.

Nessa operação simplesmente o usuário escolhe, dentro de uma visão que mostra muitos dados, uma coleção menor de dados para ver. O termo **slice** indica que ele fez essa escolha em apenas uma dimensão da consulta, cortando o cubo OLAP em uma fatia, e o termo **dice** indica que fez em várias, cortando essa fatia mais vezes e obtendo um cubo menor.

Usando o cubo da Figura 13.2, uma operação de *slice* poderia requisitar apenas os dados de São Paulo, resultando na Figura 13.7.

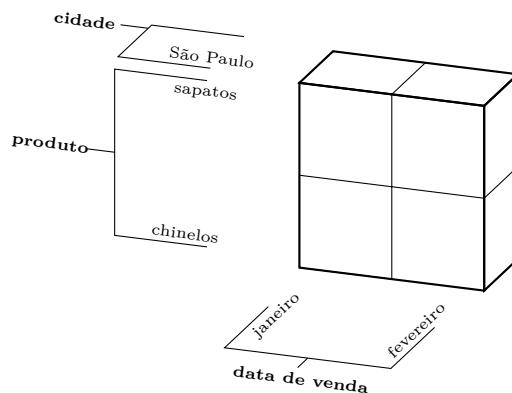


Figura 13.7.: Efeito do uso de um filtro selecionando apenas os dados da cidade de São Paulo, realizando um *slice*.

E usando mais uma vez esse cubo, é possível selecionar apenas as vendas de janeiro, o que acaba resultando em um *dice*, como na Figura 13.8.

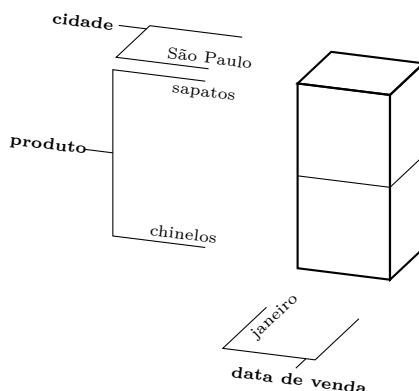


Figura 13.8.: Aplicando mais de um filtro, obtemos um *dice*.

### 13.3. Slice and Dice

É possível observar que, em relação ao cubo original, a Figura 13.7 faz um corte, portanto é um *slice*, ou seja, uma fatia do cubo. Já na figura 13.8 mais um corte é aplicado e ficamos com um pedaço menor que uma fatia, um *dice*.

Por exemplo, a Tabela 13.1 apresenta os dados de todos os estados do Brasil, tornando difícil a comparação dos dados dos estados do Sul do Brasil, RS, PR e SC, entre eles. Uma operação de *slice* então corta essa fatia, por meio de filtros, e mostra os dados desejados, como na Tabela 13.2.

Tabela 13.1.: Presenças por moradores dos estados no ENEM de 1998, Fonte: (INEP, 2016).

Estado	Presenças
AC	368
AL	152
AM	355
AP	46
BA	170
CE	622
DF	199
ES	2650
GO	443
MA	80
MG	14679
MS	1070
MT	1192
PA	248
PB	269
PE	5665
PI	68
PR	47650
RJ	22296
RN	2584
RO	99
RR	546
RS	800
SC	1001
SE	289
SP	7433
TO	44
(blank)	4557
Total	115575

## 13. OLAP

Tabela 13.2.: Presenças por moradores dos estados no ENEM de 1998, Fonte: (INEP, 2016).

Estado	Presenças
PR	47650
RS	800
SC	1001
Total	115575

### 13.3.1. Filtros (*Slice and Dice*) no Pivot Table do Excel™

Para usar os filtros do pivot table do Excel™ é preciso apertar nas pequenas caixas com setas, ou filtros, que são vistas ao lado das palavras “Estado”, “Idade” e “TP\_SEXO” na 13.5. No caso do aparecimento de um filtro, significa que há um filtro ativo. Nessa imagem já está sendo feito um filtro pelo sexo e um pela idade, logo caracterizando uma operação de *dice*.

Alterando os filtros, é possível obter outros resultados. Por exemplo, na Figura 13.9, são eliminadas as idades menores que 17 anos, resultando na tabela da Figura 13.10.

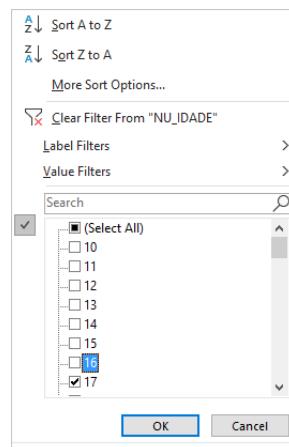


Figura 13.9.: Usando o filtro para cortar algumas idades

	A	B	C	D	E	F	G	H	I	J	
1	TP_SEXO	F									
2											
3	Presenças	Idade									
4	Estado		17	18	19	20	21	22	23	24	25
5	AC		38	28	7	9	7	2	1	1	1
6	AL		48	22	5	3	0	1			0
7	AM		82	60	12	18	7	4	6	3	2
8	AP		6	12	4	1	4				
9	BA		34	42	7	9	3	1			
10	CE		197	131	24	10	2	2	1	3	1
11	DF		47	45	8	4	2	1	1	0	1
12	ES		665	558	208	70	40	14	12	5	5
13	GO		146	74	20	10	6	5	5	2	
14	MA		19	17	5	4	2	1			1
15	MG		1704	2500	1219	730	456	302	208	124	136
16	MS		348	151	55	28	12	8	4	4	3
17	MT		286	173	107	52	34	17	9	12	3
18	PA		63	29	13	10	5	2	1		
19	PB		83	41	12	5	2	1			
20	PE		788	781	484	372	250	192	142	115	70
21	PI		13	11	2	1	2	1		2	
22	PR		5451	5291	3222	1930	1173	778	553	450	333
23	RJ		3069	3494	2136	1227	657	443	280	199	136
24	RN		272	388	248	177	143	80	64	52	28
25	RO		21	27	8	2	1			2	
26	RR		16	24	16	13	10	8	6	4	4

Figura 13.10.: Tabela de presenças no ENEM 1998 por estado para maiores de 17 anos. Fonte (INEP, 2016)

## 13.4. Drill-down e Roll-up

Quando estamos consultando uma base de dados é comum que uma pergunta possa ser feita em vários níveis de uma hierarquia que cobre o mesmo conceito.

Por exemplo, todas as perguntas a seguir querem obter informações sobre o valor total vendido, que é a informação importante, em um período de tempo, que é o conceito:

- Qual o valor total vendido por dia?
- Qual o valor total vendido por semana?
- Qual o valor total vendido por mês?
- Qual o valor total vendido por trimestre?
- Qual o valor total vendido por ano?

O mesmo pode ser visto em uma consulta sobre a quantidade de litros de combustível vendida por localidade:

- Qual a quantidade de litros de combustível vendida por posto de gasolina?
- Qual a quantidade de litros de combustível vendida por bairro?
- Qual a quantidade de litros de combustível vendida por cidade?
- Qual a quantidade de litros de combustível vendida por estado?
- Qual a quantidade de litros de combustível vendida por país?

Hierarquias como [dia - semana - mês - trimestre - ano] e [posto - bairro - cidade - estado - país] são muito comuns. Essas duas, a de intervalo de tempo e a de localidade ou endereço, inclusive, são as mais comuns e aparecem em quase todos data warehouses. Outras

### 13. OLAP

hierarquias comuns são [produto - subcategoria - categoria], ou [setor - departamento - diretoria].

Essas hierarquias são muitas vezes desenhadas, como na Figura 13.11. Em especial, o desenho mostra que mês e semana devem ser vistos em paralelo, isto é, não é possível navegar de semana para mês nesse modelo.

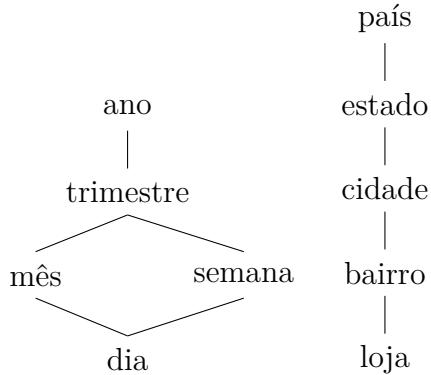


Figura 13.11.: Hierarquias de conceitos. No caso, na hierarquia de períodos de tempo, temos dois caminhos paralelos.

Muitas vezes quando fazemos uma pergunta como as listadas acima, como “Qual o valor total vendido por trimestre?” ficamos insatisfeitos com o resultado e queremos mais detalhe. Por exemplo, queremos saber o valor total vendido por mês, em um determinado trimestre. Essa operação, onde pedimos mais detalhes sobre uma dimensão da pergunta, é conhecida como **drill down**, pois é como se estivéssemos “escavando mais profundamente” para entender o que está acontecendo.

Normalmente o *drill down* é feito em uma interface gráfica, onde o usuário clica em um dado para ver o detalhe. Por exemplo, clicando em um dado sobre um mês ele poderia ver os dados sobre cada dia do mês.

A ideia básica é demonstrada na Figura 13.12, escolhendo uma célula do cubo de OLAP e fazendo o *drill down*, chegamos a um cubo mais detalhado.

Não é necessário todo um cubo para entender um *drill down*, basta olhar para uma tabela com dados agregados e procurar abrir esses dados de alguma forma prevista na interface do software.

A Tabela 13.1, por exemplo, mostra o número de presenças no ENEM de 1998, por estado de residência do candidato. Supondo que um usuário, vendo esse dado, queira saber de onde vieram os candidatos do Acre; em um sistema que suportasse o *drill down*, ele poderia clicar no Acre e receber uma visão da Tabela 13.3.

A operação inversa, que nos permite ter uma visão mais ampla, foi chamada inicialmente de **roll up**, e com o tempo ganhou também o nome de **drill up**<sup>2</sup>.

<sup>2</sup>Que, convenhamos, não faz muito sentido

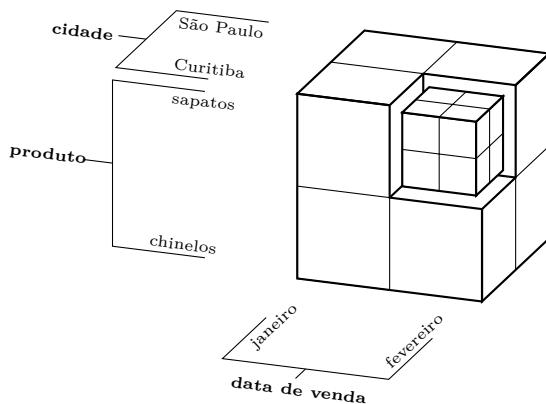


Figura 13.12.: Fazendo um *drill down* se obtém um novo cubo mais detalhado.

Tabela 13.3.: Presenças por moradores das cidades do Acre no ENEM de 1998.

Fonte: (INEP, 2016)

Estado	Presenças
ACRELANDIA	2
MANCIO LIMA	1
RIO BRANCO	365
Total AC	368

Na operação inversa é importante notar que a visão passa a um nível superior da hierarquia. Então, pedindo para fazer um roll-up quando vê a tabela de cidade do Acre (13.3), o usuário passaria a ver a tabela que inclui todos os estados, e o Acre como um deles (13.1).

### 13.4.1. Drill down e Roll Up

Nas linhas foi definida uma hierarquia [município - estado], como mostrado na Figura 13.13. A imagem ainda mostra, à direita, o painel de configuração da *pivot table*.

Clicando nos estados é possível abrir ou fechar o detalhe dos municípios, o equivalente à função de *drill down*. A Figura 13.14 mostra uma imagem da mesma pivot table com alguns estados fechados, ou seja, tendo sofrido um *roll up*.

### 13. OLAP

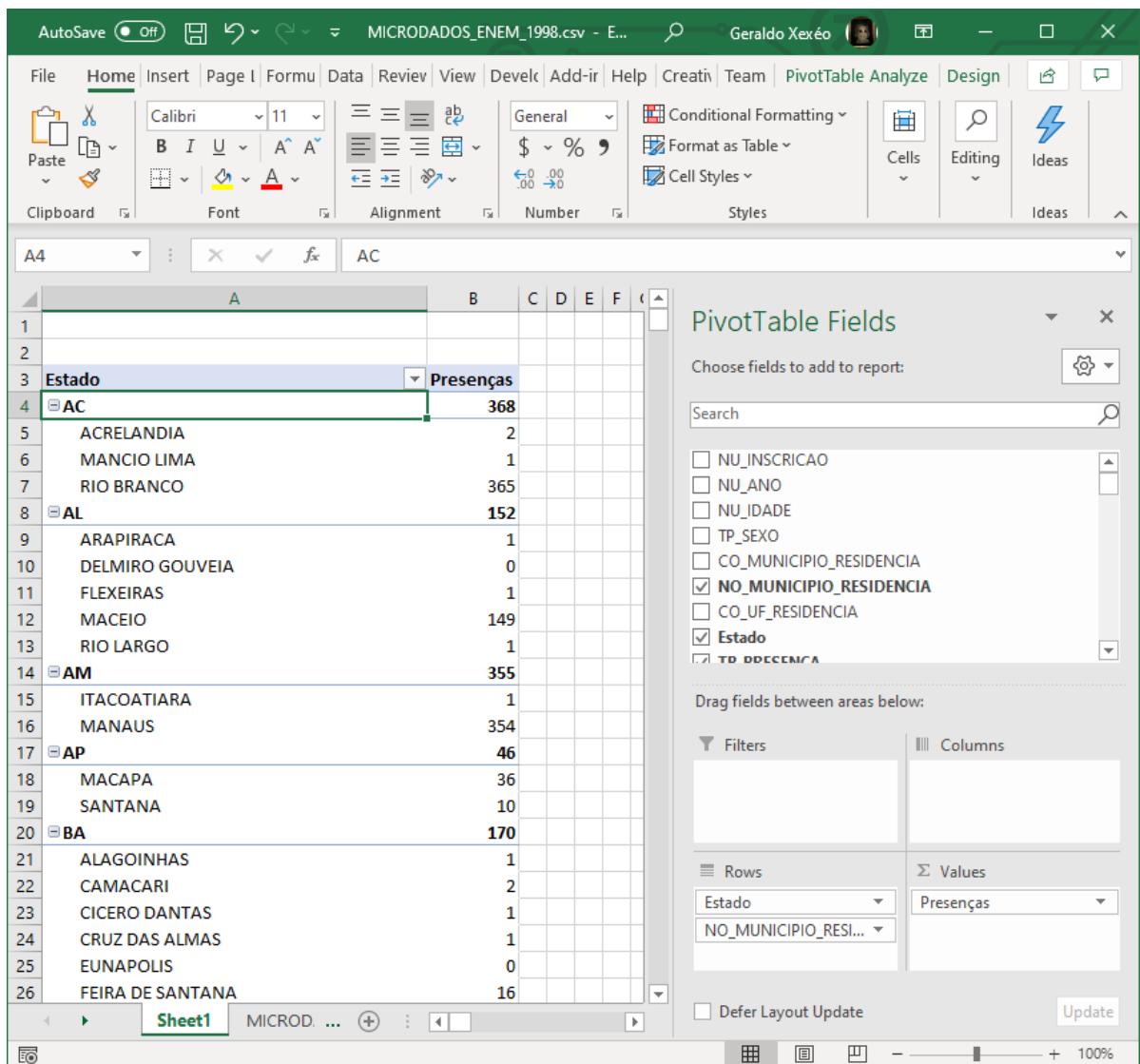


Figura 13.13.: Pequena fotografia de uma planilha com uma pivot table verificando as presenças por cidade e estado do ENEM 1998. Fonte (INEP, 2016).

### 13.4. Drill-down e Roll-up

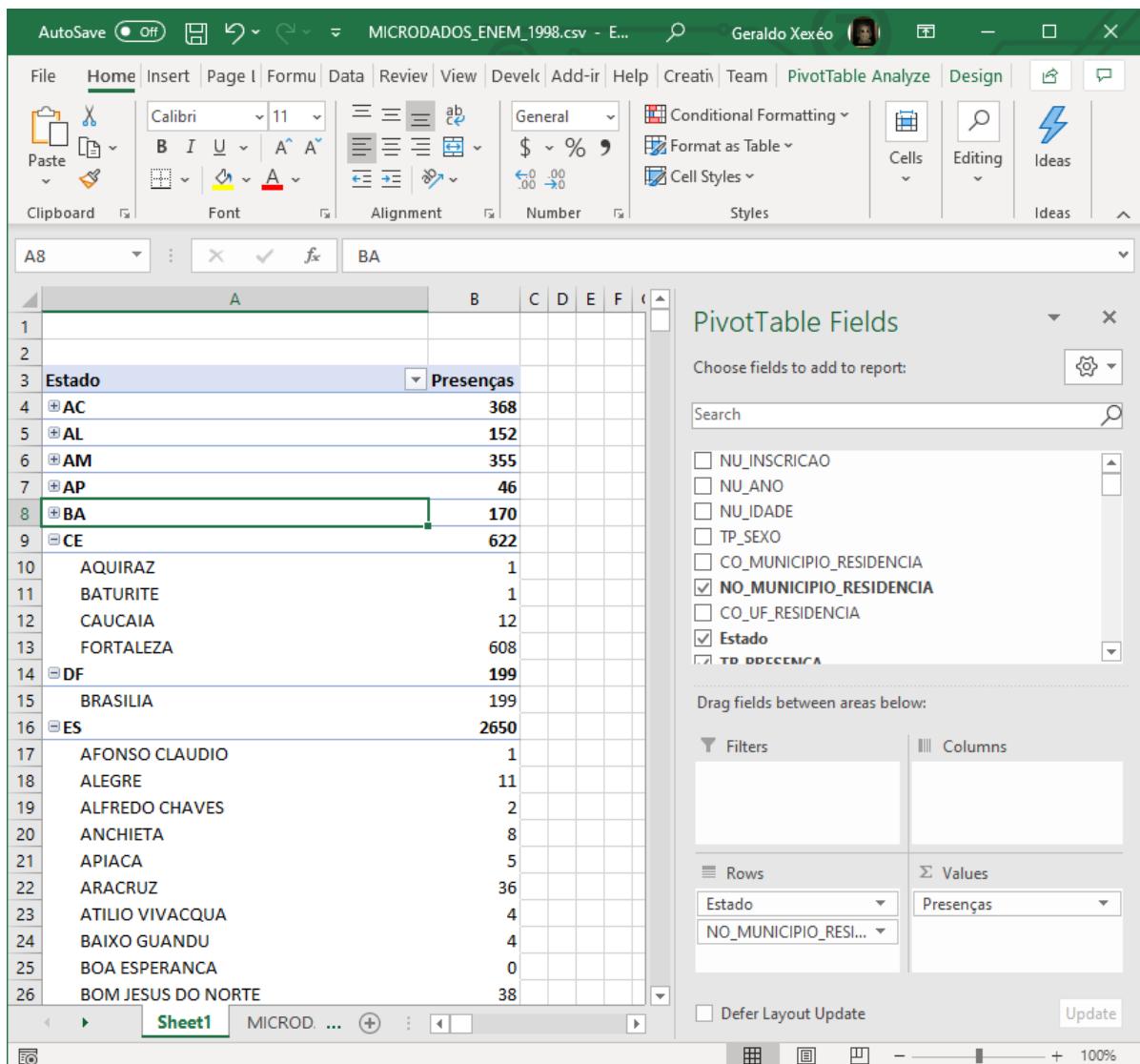


Figura 13.14.: Pivot table com algumas informações suprimidas por meio de um roll up nos estados de Acre, Alagoas, Amazonas, Amapá e Bahia. Fonte (INEP, 2016).

## 13.5. Pivot

A operação **pivot** é basicamente uma mudança de visualização, que reorienta o cubo. Seu efeito típico é trocar linhas por colunas, ou páginas por linhas.

Uma operação de *pivot* na Figura 13.2 poderia resultar na 13.15.

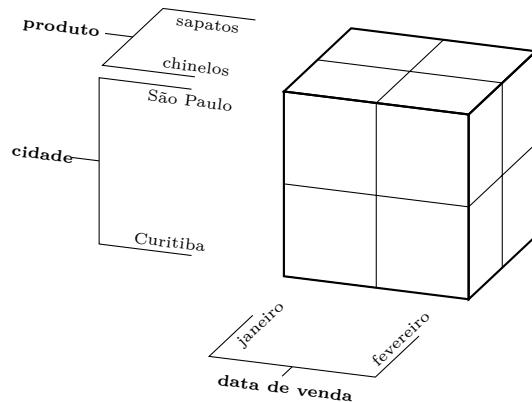


Figura 13.15.: Operação de *pivot* no cubo OLAP troca a posição das dimensões. É necessário comparar com a Figura 13.2.

### 13.5.1. Pivot no Excel™

Já diz o nome que as *pivot table* no Excel™ permitem facilmente fazer o *pivot*. Para isso basta um dado de linha para coluna e vice versa, com é demonstrado nas Figuras 13.16 e 13.17.

### 13.5. Pivot

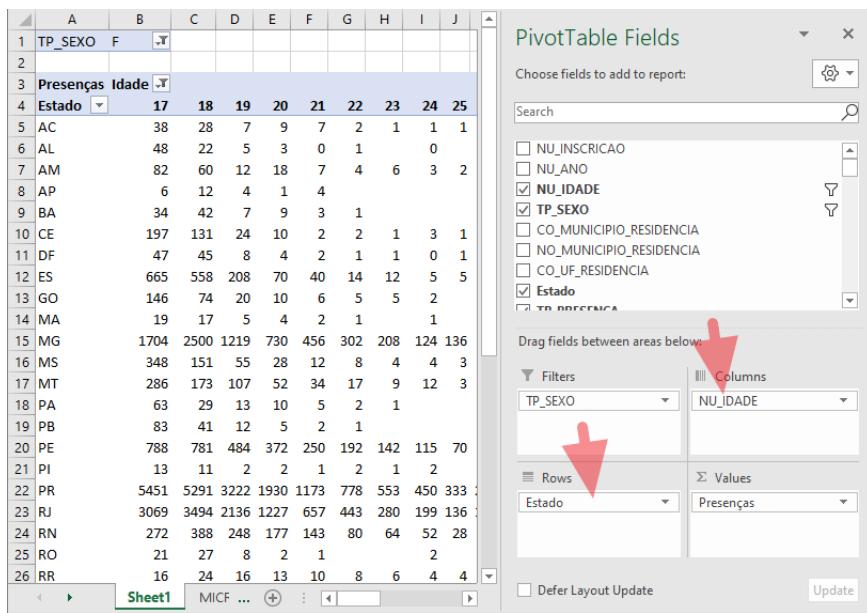


Figura 13.16.: Posição dos dados antes da operação *pivot*.

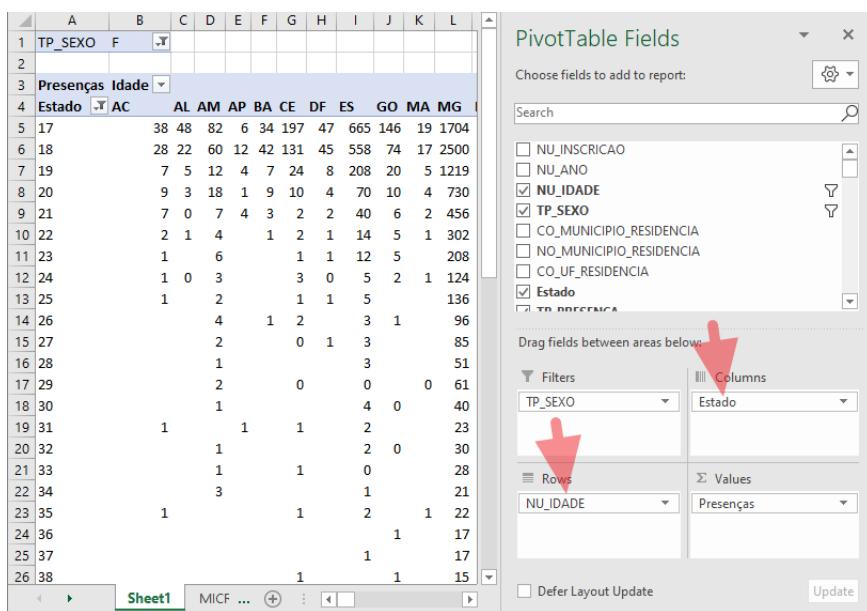


Figura 13.17.: Após o *pivot*.



# 14

## Modelagem Dimensional

### Conteúdo

---

14.1. O Modelo Estrela . . . . .	164
----------------------------------	-----

O objetivo da Modelagem Dimensional é fornecer ao usuários de um negócio uma visão intuitiva dos informações a que desejam ter acesso. O foco da Modelagem Dimensional é a simplicidade, ao contrário de outros modelos conceituais ou lógicos que buscam uma grande força de expressão ou flexibilidade no uso.

Outra função importante da Modelagem Dimensional é criar uma representação lógica que já direcione a organização das representações físicas possíveis para um alto desempenho de consulta.

Inicialmente o Modelo Dimensional era composto de uma tabela com uma chave composta com uma grande quantidade de dados, conhecida como Tabela Fato , que contém os dados numéricos que são manipulados nos relatórios e gráficos desejados, e um conjunto de tabelas que definem parâmetros que são usados em processos de agregação, projeção ou seleção, cada uma sendo uma Tabela Dimensão.

A ideia básica é que as tabelas dimensão fornecem as dimensões para a análise dos dados da tabela fato. Desta forma, cada tabela dimensão tem uma chave primária simples, que é referenciada pelas chaves externas da tabela fato. Isso faz com que a tabela fato seja cercada de outras tabelas, o que resulta em um modelo conhecido como Modelo Estrela.

O Modelo Dimensional, ou o Modelo Estrela, divide o mundo em dois grandes grupos de dados, Medidas e Contexto.

**Medidas** são dados capturados pelos sistemas de informação que executam os processos de negócio da organização. Normalmente são valores numéricos, sobre os quais podemos

## 14. Modelagem Dimensional

fazer operações como somatório ou média. Números como o CPF e o Telefone de uma pessoa não são considerados medidas, e normalmente não são considerados números. As medidas definem os **fatos**(Kimball et al., 2008).

**Contexto** são os dados que mostram em que situação, momento, ou outros parâmetros os dados foram adquiridos. Normalmente são dados textuais ou datas, indicando informações como nomes, tipos de produto, datas de compra e venda, etc. O contexto é representado pelas dimensões.

As dimensões normalmente respondem as famosas questões do 5W2H: o que, onde, quando, quem, por que, como e por quanto(Corr e Stagnitto, 2012).

(Kimball et al., 2008) propõe que cada processo pode ser representado por um modelo dimensional, contendo uma tabela fato com as medidas numéricas levantadas ao longo de uma execução do proecesso, e tabelas que definem o contexto de cada medida. Na prática, várias tabelas fato podem ser levantadas para o mesmo processo, inclusive porque há escolhas de modelagem que são feitas, de acordo com a necessidade dos usuários, e também há uma quantidade de usuários e visões diferentes que um processo pode ter dentro de um sistema de data warehousing.

### 14.1. O Modelo Estrela

O Modelo Estrela é baseado em 4 conceitos(Kimball et al., 2008):

1. fatos,
2. dimensões,
3. atributos, e
4. relacionamentos.

#### 14.1.1. A Tabela Fato

Uma tabela fato guarda indicadores de desempenho gerados pela organização durante seu funcionamentos. Tipicamente elas possuem milhares ou milhões de linhas, ocupando grande parte da área de disco usada pelo sistema de data warehousing, chegando a 90% de todos os dados(Kimball et al., 2008).

Esses valores devem ser numéricos e aditivos, para serem analisados em tabelas e gráficos. Muitos deles representam dinheiro e quantidades.

Os tipos de fatos que podemos encontrar são(Kimball et al., 2008):

- **aditivos**, que são fatos a serem resumidos, podendo ser acumulados em qualquer dimensão; exemplos típicos são quantidade vendida e valor da venda;
- **semi-aditivos**, que só podem ser acumulados em algumas dimensões, como saldo em conta e quantidade em estoque, e

#### *14.1. O Modelo Estrela*

- **não aditivos**, que não podem ser resumidos, e são armazenados na dimensão, não sendo, na verdade, fatos, como o preço médio.

Na chave primária da tabela fato estão todas as chaves primárias das tabelas dimensão, representando uma ou mais relacionamentos 1:N com todas essas tabelas. Um exemplo de mais de um relacionamento seria um registro de compra e venda, que teria dois relacionamentos com a dimensão pessoa, uma indicando um vendedor, outra indicando um comprador.

É possível que uma tabela fato não possua nenhum dado além das chaves estrangeiras(Kimball et al., 2008).



# 15

## O Método Beam\*

**Beam\***, **Business Event Analysis and Modelling** é um método ágil de projeto de Data Warehouse proposto por Corr e Stagnitto (2012) que visa geral o modelo estrela de forma colaborativa. Ele se baseia no conceito de 5W2H, que os autores chamam de **7W**. Esse capítulo é um resumo do material apresentado no livro<sup>1</sup>.

O método propõe um ciclo de vida ágil, baseado em iterações sucessivas em busca de obter valor para as partes interessadas. Cada passo do ciclo tem quatro passos principais: Levantar Requisitos, Construir o ETL, Construir o Protótipo de Business Intelligence e Revisão. Dentro desse ciclo de vida, BEAM\* se preocupa principalmente em definir um modelo dimensional, na forma de um Esquema Estrela, de maneira colaborativa, em sessões de trabalho conjuntas.

Para construir o modelo dimensional, o método busca responder as perguntas 5W2H, *Who, What, Where, When, How Many, Why and How*, a partir de exemplos colocados em uma tabela em um quadro branco. Na prática funciona como uma modelagem de dados por exemplo. Corr e Stagnitto (2012) chamam isso de *Framework 7W*, e o ilustram com a Figura

A **modelagem ágil de dados** tem como característica ser Colaborativa, combinando análise e projeto em um mesmo processo e envolvendo as partes interessadas, de forma evolucionária, incremental e iterativa.

Uma visão rápida do método de modelagem colaborativa pode ser resumida em quatro passos:

1. Junte seus usuários de uma área de negócios
2. Pergunte o que acontece nessa área, isto é, eventos na forma “quem faz o que?”, levantando uma lista priorizada desses eventos;
3. Peça exemplos de dados reais de “quem” e o “que”, do evento prioritário;

---

<sup>1</sup>Todo esse capítulo é um resumo baseado no livro, com alguns comentários deste autor, onde foi feito o maior esforço possível para deixar indicadas as citações literais.

## 15. O Método Beam\*

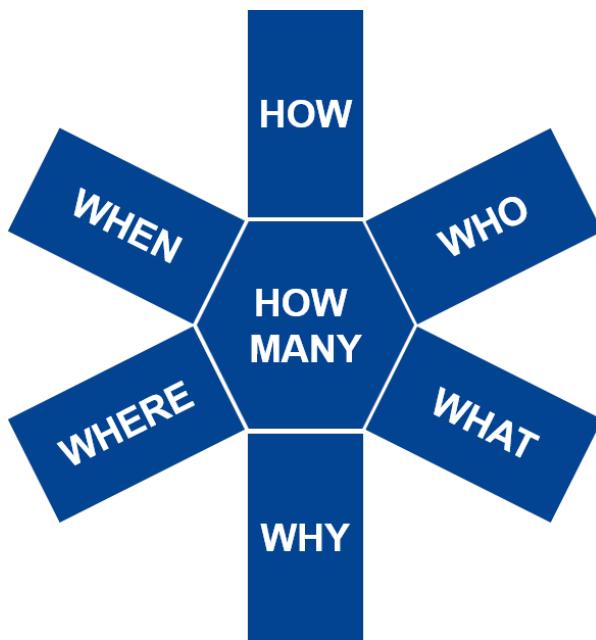


Figura 15.1.: Descrição gráfica do *Framework 7W*. Fonte:(Corr e Stagnitto, 2012)

4. Preencha o resto dos dados: quando, onde, quantos, por que e como.
5. Repita para cada “quem faz o que?”, por ordem de prioridade.

Isso permite criar, para cada um desses eventos, um modelo estrela que segue o **Framework 7W**, formado por uma Tabela Fato que descreve ”quantos?” cercada de Tabelas Dimensão que definem como, quem, o que, por que, onde e quando.

O método usa 5 ferramentas: a Tabela Beam\*, a Carta de Hierarquia, a Linha de Tempo, a Matriz de Eventos e o Modelo Estrela Aumentado.

Cada **Tabela Beam\*** modela um evento de negócio, usando exemplos de dados reais para documentar os detalhes para o framework 7W. Ela descreve fatos e dimensões e explica os padrões do projeto dimensional.

A **Carta de Hierarquia** é um diagrama simples que mostra as relações hierárquicas entre os atributos das dimensões. Ela ajuda a levantar quais são esses atributos e também a entender que operações OLAP de *drill down* e *roll up* que serão necessárias.

As **Linhas de Tempo** exploram as relações temporais entre os eventos de negócio, permitindo descobrir detalhes da pergunta “Quando?” Elas também indicam como funcionam os processos e como são sequenciados, além da duração dos fatos.

A **Matriz de Eventos** mostra os relacionamentos entre todos os eventos e as dimensões do modelo.

O **Modelo Estrela Aumentado** provê a visualização do modelo dimensional e permite gerar esquemas físicos. O modelo tradicional é estendido com **short codes**

**BEAM\***, que são siglas de duas letras que indicam propriedades dos dados, fornecendo informação sem ocupar espaços.

A estratégia de trabalho é chamada *Modelstorming* e se baseia na tríade abrir, explorar e fechar, comum em métodos criativos. Esse modelo pode ser descrito na figura 15.2. Nessa figura são descritas as atividades e as ferramentas nelas usadas.

O primeiro passo é descobrir “Quem faz o que”, esse passo fornece as informações iniciais que criam a **Tabela Beam\***. Os dados de cada evento são detalhados. O próximo passo é modelar as dimensões, ainda com a Tabela Beam\* e criando a Carta de Hierarquia. A seguir são modeladas as sequências de evento, o que produz a Matriz de Eventos e a Linha do Tempo. Finalmente se decide que eventos implementar.

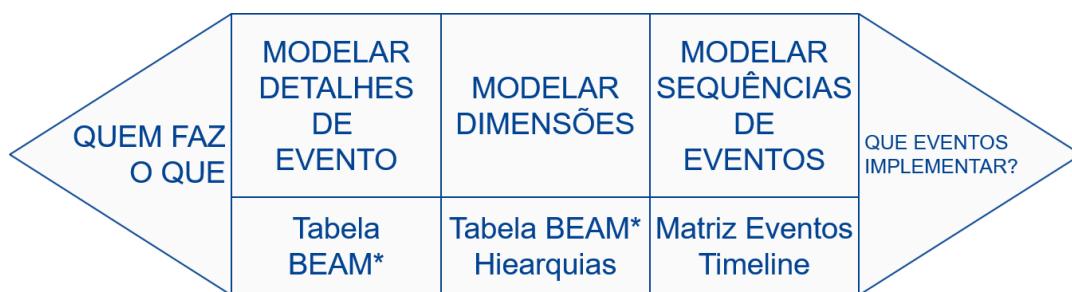


Figura 15.2.: Descrição gráfica do Modelstorming. Fonte:(Corr e Stagnitto, 2012)

Esse diagrama pode ser definido como um conjunto de processos de negócio escritos em ARIS-EPC, como na Figura 15.3. Isso define basicamente um processo de seis passos:

1. perguntar aos participantes quem faz o que nos negócios, o que gera uma lista de histórias, cada história descrevendo um evento;
2. priorizar as histórias
3. documentar cada evento em uma tabela BEAM\* seguindo uma **ordem de descoberta**;
4. descrever detalhadamente as dimensões relativas a cada evento;
5. criar o Modelo de Sequência de Eventos, e
6. escolher os modelos a implementar.

## 15. O Método Beam\*

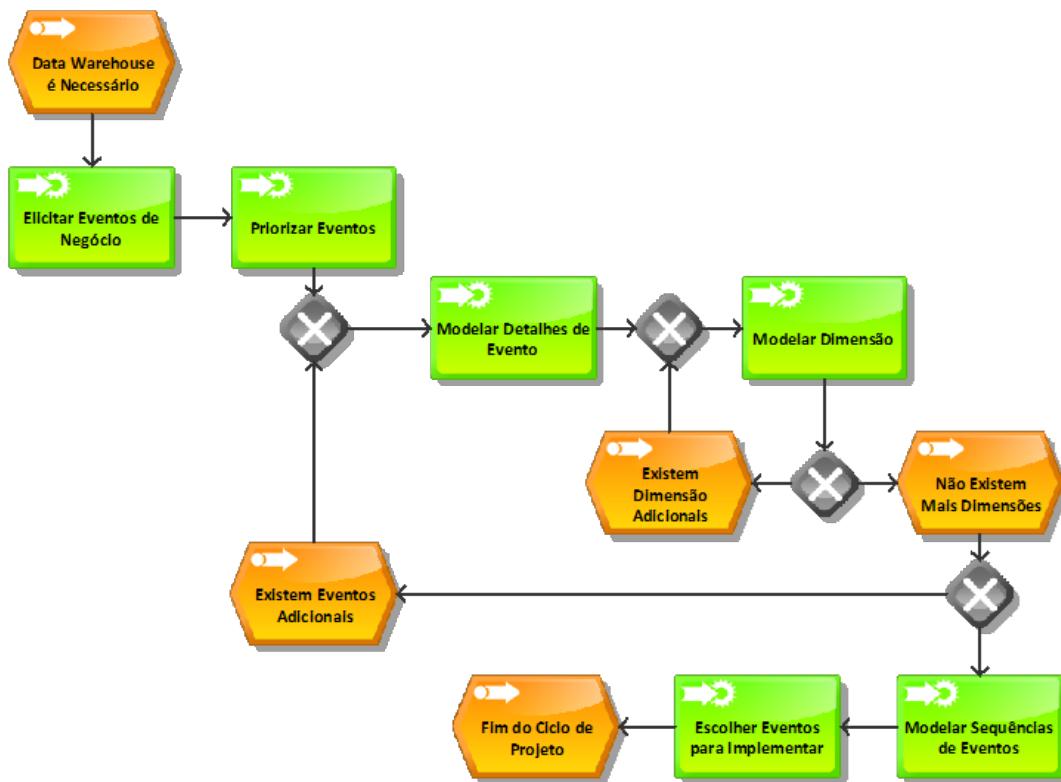


Figura 15.3.: Descrição do Modelstorming em ARIS-EPC

### 15.1. Eventos de Negócio OU Histórias de Dados

Segundo Corr e Stagnitto (2012), um **Evento de Negócio** é uma ação individual, executada por pessoas ou organizações durante a execução de seus processos de negócio. Quando a empresa funciona, deixa uma trilha de eventos de negócios dentro dos bancos de dados operacionais, que contém detalhes mensuráveis no nível atômico. Exemplos de eventos de negócio são: uma venda, uma entrega, uma montagem de equipamento, o atendimento de uma reclamação, etc.

Para identificar um evento de negócio, no Beam\* as partes interessadas devem contar suas *data stories*, que vão produzir requisitos de dados concisos.

As *data stories*, ou **histórias de dados**, são semelhantes conceitualmente às histórias de usuário. Elas são escritas ou contadas pelas partes interessadas envolvidas com o negócio.

Essas histórias devem se concentrar em requisitos de dados, e são escritas como um quadro, tabela ou planilha. Assim, a narrativa de um evento de negócio é usada para descobrir requisitos de dados para o BI.

Basicamente, Histórias de Dados e Eventos de Negócio podem ser usados de formas intercambiáveis, mas a ideia é que Histórias de dados descrevem Eventos de Negócios.

Eventos de negócio são sempre atividades, sendo representados por verbos. Existem três tipos de histórias de dados, ou eventos de negócio: discretas, evolutivas ou recorrentes.

Um **evento discreto** acontece em um instante ou um curto espaço de tempo que pode ser considerado pontual. Eventos desse tipo representam ações atômicas, por exemplo, a compra de um produto por um cliente em uma loja de varejo. Esses eventos são considerados completos no momento em que ocorrem, sendo considerados a partir daí acabados e imutáveis. Isso significa que quando chegam ao BI, provenientes dos sistemas operacionais da empresa, não serão mais alterados. Normalmente esses eventos usam um único verbo e um único *timestamp*. Além disso, seus detalhes não mudam no tempo.

Um **evento evolutivo** representa algo que acontece em períodos irregulares de tempo, podendo migrar dos sistemas operacionais para o BI antes de estarem terminados. Por exemplo, um venda feita pela internet e que já foi paga mas não foi entregue. Eventos desse tipo possuem vários verbos em sua descrição, como “pagar” e “entregar”. São, normalmente, um série de eventos discretos, como passos ou marcos de um processo longo.

Um **evento recorrente** aconcede em intervalos previsíveis e regulares. Por exemplo, um inventário noturno de um produto ou loja. Eventos desse tipo podem ser amostras ou sumários de eventos discretos, podendo ser cumulativos. São detalhes atômicos de medidas automáticas do sistema, como o saldo no fim do dia de uma conta bancária.

Cada um desses eventos exige um tipo de Tabela Fato, entre as propostas por Kimball et al. (2008), para representá-lo, como na Tabela 15.1.

Tabela 15.1.: Equivalência dos tipos de eventos do método BEAM\* e os Tipos de Tabela Fato de Kimball. Fonte: Corr e Stagnitto, 2012

Tipo de Evento	Tipo de Esquema Estrela
Discreto	Transação
Evolutivo	Fotografia Acumulativa
Recorrente	Fotografia Periódica

## 15.2. Descobrindo as Histórias

O primeiro passo da criação da Modelagem de Eventos de Negócio é descobrir quem faz o que, e está indicado na Figura 15.4. No Processo Modelstorming, da Figura 15.3, adaptada de Corr e Stagnitto (2012). Esse a passo corresponde as atividades Elicitar Eventos de Negócio e Priorizar Eventos.

Nesse passo, os participantes da reunião devem gerar sentenças na forma sujeito-verbo-objeto que indiquem quem faz o que no processo. Cada resposta dada nesse formato define um evento de negócio.

## 15. O Método Beam\*

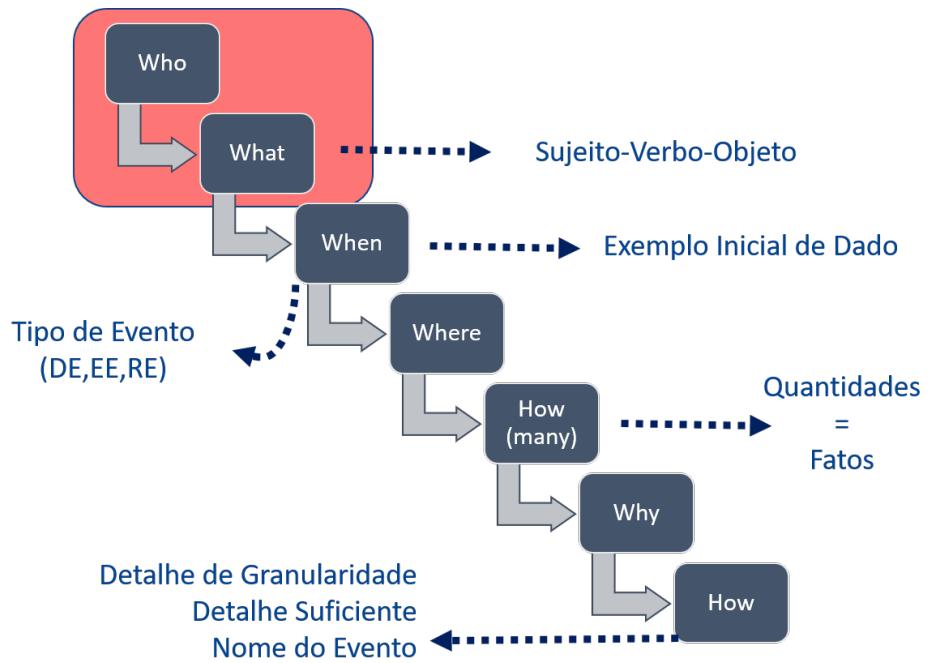


Figura 15.4.: Primeiro passo da ordem de descoberta. Fonte:(Corr e Stagnitto, 2012)

Um foco a ser dado nessa descrição, ou na seleção dos eventos que serão usados no próximo passo, é a necessidade desse evento ser interessante para a próxima *release* do projeto. Assim, essas sentenças podem ser levadas para um *backlog*, caso não sejam usadas nessa iteração.

É importante que, levantados os eventos, eles sejam tratados um de cada vez, e que sejam, pelo menos nas primeiras iterações, evento atômicos.

A descrição, feita pelos participantes, não deve se preocupar com o que eles propriamente ditos fazem, mas sim com que pessoa ou organização realmente executa a atividade que o verbo descreve.

Um exemplo simples é dado por Corr e Stagnitto (2012): o grupo pode estar convicto que é importante caracterizar o faturamento gerado por um produto, mas quais eventos ele pode procurar para isso? A resposta indica dois eventos possíveis: cliente compra produto ou vendedor vende produto. É possível ver a estrutura sujeito-verbo-objeto e a diferença de perspectiva do que é mais importante para a organização dada a escolha de uma das sentenças. É válido observar que outros sujeitos poderão ser inseridos depois, isto é, mesmo que a organização prefira privilegiar a narrativa que cliente compra sobre a que vendedor vende no início desse passo a passo, nos passos posteriores o outro agente poderá ser incluído, ou seja, a informação não será perdida.

### 15.3. Passo a Passo da Modelagem de Eventos de Negócio

A escolha do verbo é muito importante nesse passo, pois ele sempre fará parte das perguntas do *framework 7W*. A primeira pergunta é basicamente “Quem faz o que?”, e as próximas perguntas serão “Quando faz?”, “Como faz?”, etc.

Uma forma de descobrir os eventos é partir dos atores envolvidos na organização. Escolher um ator, como motorista, ou cliente, permite investigar o que ele faz. Atores podem fazer mais de uma coisa, por exemplo, um cliente pode: comprar, pedir cotação, devolver, etc.

#### 15.2.1. Prática para Levantamento e Seleção de Eventos

O levantamento de eventos pode ser feito de algumas maneiras. Um *brainstorm*, por exemplo, pode ser uma prática adequada para levantar os eventos.

Outra prática comum é o uso de *post-its* para escrever os eventos e colocá-los em um espaço ou quadro. Nesse caso, todos devem ter acesso ao quadro e ficar em pé para trabalhar no mesmo.

O quadro pode ser organizado ou não. Pode se organizar o quadro como uma linha de tempo, ou por ator, ou como o grupo achar mais eficiente. Também é possível não organizar o quadro inicialmente e depois agrupar os eventos obtidos em alguma organização.

Antes da seleção, porém, é importante localizar os eventos repetidos ou que possam significar a mesma coisa. Corrigidas as redundâncias, e até mesmo algum erro eventual, pode se passar para a fase de seleção.

Uma método comum de seleção é chamado de *dot-voting*, ou votar por pontos. Nesse caso, cada participante recebe um número de pontos que deve gastar marcando os eventos mais importantes ou mais prioritários para essa release. É comum que isso seja feito com 3 votos, na forma de marcar com uma caneta ou uma pequena etiqueta adesiva, ou mesmo um pedaço de *post-it*. Os eventos então são listados em ordem de votação para serem analisados detalhadamente.

## 15.3. Passo a Passo da Modelagem de Eventos de Negócio

Cada história de dados será contada por meio de uma Tabela BEAM\*. Nesta seção o método de construção desta tabela é mostrado passo a passo.

Esses passos correspondem a atividade Modelar Detalhes de Evento no Processo Modelstorming descrito pela Figura 15.3.

Cada história é contada na Tabela BEAM\* por meio da identificação de **detalhes** que a explicam e do uso de exemplos que mostram como podem acontecer.

## 15. O Método Beam\*

Esses detalhes mostram as várias informações que podem ser usadas na história para entendê-la melhor e trazer mais entendimento do negócio ou da organização alvo do sistema de BI ou Data Warehouse sendo criado.

Os detalhes correspondem a respostas a perguntas do 5W2H, ou 7W da notação de Corr e Stagnitto (2012). O termo **detalhe** é usado para mostrar que ainda não são as dimensões finais do Modelo Dimensional.

A atividade Modelar Detalhes de Eventos é na verdade um sub-processo que é descrito em ARIS-EPC na Figura 15.5 e que gera a Tabela BEAM\*.

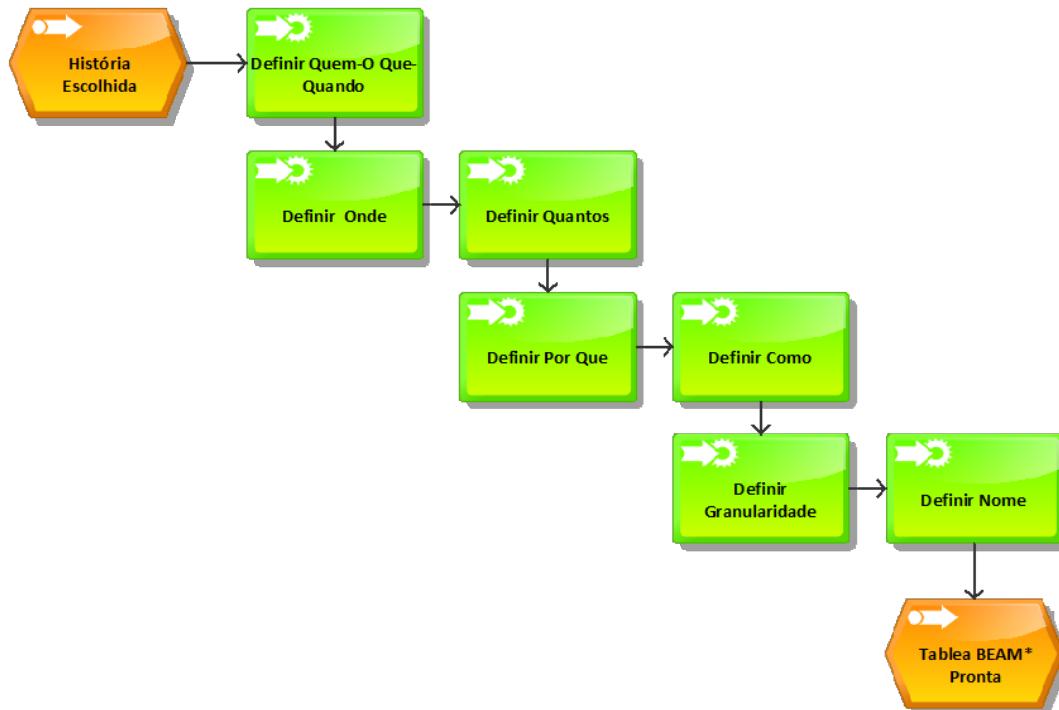


Figura 15.5.: O processo Modelar Detalhes de Evento, descrito em ARIS-EPC

Dentro desse processo, a primeira atividade a ser realizada, Definir Quem-O Que-Quando, também é um subprocesso, que é descrito em mais detalhe na Figura 15.6.

Nas próximas subseções serão descritos os passos de ambos os processos.

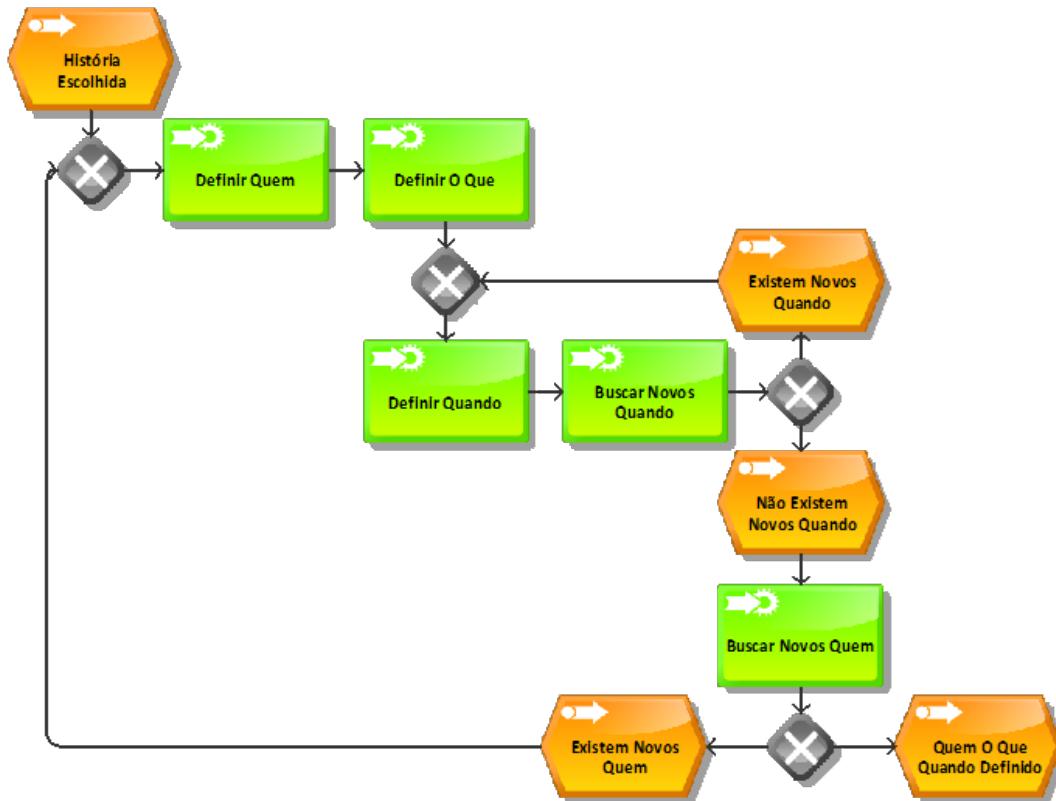


Figura 15.6.: Definir Quem-O Que-Quando, descrito em ARIS-EPC

### 15.3.1. A sessão de criação da Tabela BEAM\*

A sessão de criação da Tabela BEAM\* deve seguir o formato ativo e colaborativo que todo o método propõe.

A equipe deve ser a mesma que levantou as histórias, na verdade, o ideal é que essa sessão seja contínua com a anterior, pelo menos para a primeira história contada, para que os participantes possam entender melhor o resultado prático de suas ações.

A dinâmica exige o uso de um quadro branco, método preferido deste autor, ou ainda de grandes folhas brancas, que tem o defeito de não permitir apagar as informações. Obviamente, são necessárias canetas, possivelmente de mais de uma cor.

A ideia é que todos os participantes tenham acesso ao quadro branco, para incentivar a participação. É necessário que alguém com conhecimento do método participe, preferencialmente na forma de mediador e orientador.

Esta forma de trabalhar, participativa, ativa e colaborativa é comum a vários métodos modernos e tem bastante sucesso.

A tecnologia atual permite que sejam tiradas fotografias do quadro, em vários momentos, que podem ser compartilhadas entre os participantes e guardadas como referência.

## 15. O Método Beam\*

Durante ou após a sessão é interessante transcrever os resultados para planilhas eletrônicas.

### 15.3.2. Tabela BEAM\* Inicial

Tudo se inicia com a criação da primeira Tabela Beam\* em seu formato inicial. Essa tabela, feita no quadro branco, começa com o formato indicado na Figura 15.7, que indica o evento por meio de seu sujeito, verbo e objeto, como descritos no levantamento de histórias. Deve ser escolhida a história mais prioritária. No caso exemplo, a história escolhida é “Cliente encomenda produto”.

No quadro branco, deve se deixar bastante espaço em baixo e à direita, de forma que possam ser desenhadas novas colunas e usado o espaço da tabela para fornecer os exemplos que ajudarão a entender o evento.

CLIENTE	encomenda	PRODUTO
<who>	<what>	

Figura 15.7.: Configuração inicial da tabela BEAM\* para o evento Cliente Encomenda Produto. É necessário deixar bastante espaço para continuar a atividade.

Adaptado de:(Corr e Stagnitto, 2012)

A Figura 15.8 mostra a ordem de descoberta das informações necessárias para construir a Tabela BEAM\* para um evento. Estas informações são os **detalhes** da história e devem ser levantados de forma específica.

As subseções a seguir descrevem esse passo a passo.

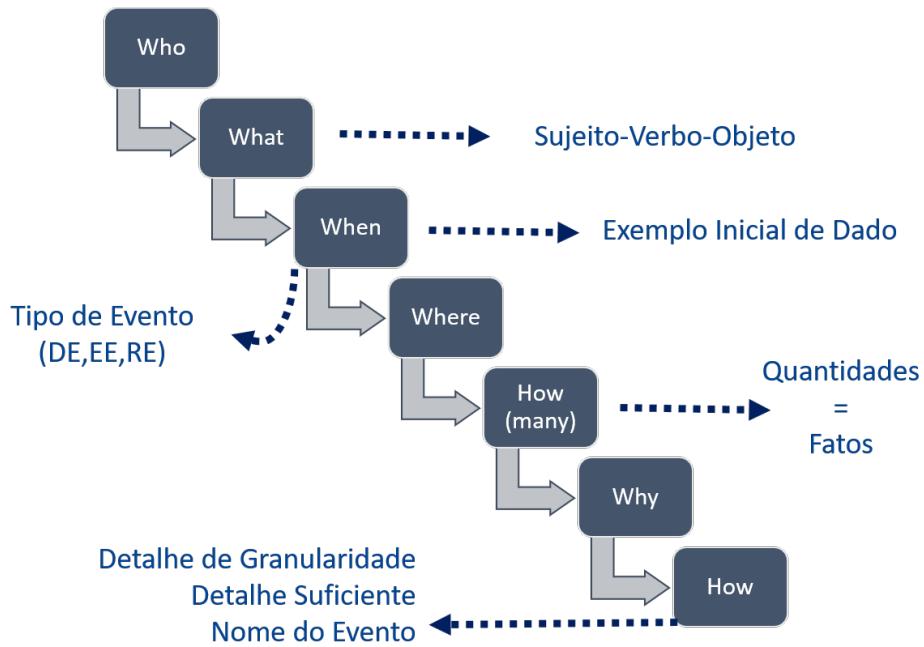


Figura 15.8.: Ordem de descoberta das informações do modelo. Fonte:(Corr e Stagnitto, 2012)

### 15.3.3. Modelando o quando

Todo evento, e toda história de dados, acontece em algum momento ou intervalo. Por isso tabelas dimensão falando sobre datas e horas são tão pervasivas nos modelos dimensionais.

O tempo em que acontece um evento é normalmente descrito por uma preposição e um substantivo, como em “na (em+a) data combinada”. As preposições que transmitem relações de tempo, muitas vezes em contração com um artigo, são: a, até, após, de, desde, em, entre, durante e todo.

Os detalhes definidos por “quando” são capazes, muitas vezes, de dar dicas adicionais sobre os detalhes, e por isso é a primeira após aquelas que definem o evento.

A Tabela BEAM\* com o primeiro detalhe de tempo fica como na Figura 15.9. A história passa a ser “Cliente encomenda produto na data do pedido.”

### 15.3.4. Primeiros Exemplos

Alcançado o ponto de preenchimento da Tabela Beam\* em que se obtém apenas os cabeçalhos das três primeiras colunas, é o momento de começar a preencher-a com exemplos de dados.

## 15. O Método Beam\*

CLIENTE	encomenda PRODUTO	na DATA DE PEDIDO
<who>	<what>	<quando>

Figura 15.9.: Configuração da tabela BEAM\* com o detalhe “quando”. Aparece a preposição utilizada (em+a) e um sustantivo. Adaptado de:(Corr e Stagnitto, 2012)

Para isso devem ser encontrados exemplos úteis para entender essa história, a partir de **temas**. Os temas sugeridos são:

**Típico** exemplos típicos dos dados que representam o evento, como os dados normais de uma venda;

**Diferente** mais exemplos semelhantes aos típicos, porém diferentes do primeiro. Também eventos buscando limites dos dados, como valores mínimos e máximos;

**Repetido** exemplos tão semelhantes quanto possível aos eventos típicos;

**Ausências** exemplos onde faltam algumas informações;

**Grupos** exemplos onde aparecem grupos, como grupos de clientes ou de produto. e

**Limites** exemplos que mostram limites.

No exemplo, que agora evolui com seu caso típico, a história passa a ser “Cliente Antônio Alves encomenda produto suéter azul na data de pedido 10 de maio de 2011.” Isso permite chegar a tabela BEAM\* da Figura 15.10. O ideal é que sejam exemplos reais conhecidos pelos participantes ou tirados diretamente das bases de dados da empresa.

Em busca de histórias diferentes, encontra-se primeiro uma variação simples, como mostra a Figura 15.11.

Seguindo, os participantes podem encontrar um exemplo com repetições, como na Figura 15.12. Nesse caso temos uma compra igual, onde a mesma pessoa comprou o mesmo item na mesma data, ou seja, ela comprou o mesmo item duas vezes e foi registrado assim.

### 15.3. Passo a Passo da Modelagem de Eventos de Negócio

CLIENTE	encomenda PRODUTO	DATA DE PEDIDO	
<who>	<what> MD	<quando> MD	
Antônio Alves	Suéter Azul	10-maio-2011	

Figura 15.10.: Configuração da tabela BEAM\* com o primeiro exemplo, mostrando uma história típica. Adaptado de:(Corr e Stagnitto, 2012)

Também é possível que para algumas histórias que ocorreram, algum dado esteja faltando. Não importa o motivo, o que queremos registrar é que, nos sistemas da organização é possível encontrar o registro de um evento assim. Na Figura 15.13 temos um caso onde não foi registrado o nome do cliente, mas onde também a data está registrada de forma incerta, ou é conhecida por uma regra, pois veio de um sistema que era usado “dez anos atrás.”

O processo segue dessa forma, com os colaboradores preenchendo mais e mais exemplos, até que achem que todas as possibilidades foram esgotadas. Na Figura 15.14 foram adicionados dois casos onde o cliente é um **grupo**: Colégio LMS e Empresa EPM. Um grupo é apenas algo coletivo em relação ao normal esperado no campo. Também aparecem registros que contém mais de um item, como “Calça+Camisa”

Para trabalhar com a dimensão “quando” é interessante pensar em referências relativas, como ontem, hoje e 5 anos atrás. Isso facilitará aos participantes entender melhor a latência do dado, ou por quanto tempo ele é interessante.

## Sujeitos desconhecidos

Em muitos casos o sujeito existe mas nunca é registrado. Isso é o caso, por exemplo, de uma rede de vendas a varejo que nunca guarda o nome do cliente.

Nesse caso o que acontece é que o sujeito existe no modelo lógico, mas não existirá no modelo físico implementado. Durante a reunião pode ser usado um nome artificial, como

## 15. O Método Beam\*

CLÍENTE	encomenda PRODUTO	na DATA DE PEDIDO	
<who>	<what> MD	<quando> MD	
Antônio Alves Bia Betrand	Suéter Azul Calça Vison	10-maio-2011 29-jun-2011	

Figura 15.11.: Configuração da tabela BEAM\* com o segundo exemplo, mostrando uma outra história típica, diferente da primeira. Adaptado de:(Corr e Stagnitto, 2012)

“Zé das Couves” e “Maria Chiquinha”, ou pode ser usado apenas um traço ou a palavra anônimo.

### 15.3.5. Refazendo o ciclo “quem-o que -como”

Após terminar o primeiro passo preenchendo os exemplos com as colunas “quem”, “o que” e “quando”, é **necessário** exaurir as possibilidades de existirem outras colunas semelhantes.

É possível encontrar mais de uma coluna “quando”, principalmente quando em vez de um instante está sendo tratado um intervalo de tempo. Um exemplo típico é um processo que começa com um pedido e termina com uma entrega. A Figura 15.15 mostra como isso pode ser feito, e não há nenhuma complicaçāo.

É possível encontrar mais de uma coluna “quem” quando existem outros participantes na história. Por exemplo, um cliente pode pedir um produto na data do pedido para um vendedor. Assim, podem aparecer novas colunas do tipo “quem”. A Figura 15.16 mostra um exemplo com duas colunas do tipo “quem”: cliente e “com vendedor”.

Personagens adicionais dessa história possuem uma preposição que mostra uma relação com a história, enquanto o personagem principal se relaciona por meio do verbo.

### 15.3. Passo a Passo da Modelagem de Eventos de Negócio

CLIENTE	encomenda PRODUTO	DATA DE PEDIDO	
<who>	<what> MD	<quando> MD	
Antônio Alves Bia Betrand Antônio Alves	Suéter Azul Calça Vison Suéter Azul	10-maio-2011 29-jun-2011 10-maio-2011	

Figura 15.12.: Configuração da tabela BEAM\* com um exemplo de repetição de um item. Adaptado de:(Corr e Stagnitto, 2012)

#### Detalhes dos detalhes

Não devem ser confundidos detalhes adicionais da história com detalhes dos detalhes da história.

O participantes podem, nesse ponto, querer registrar na tabela detalhes adicionais sobre os detalhes que já estão lá.

Por exemplo, existindo o detalhe **produto** na tabela da Figura 15.16, um participante pode achar conveniente colocar uma coluna adicional “Tipo do Produto”. Isso não deve ser colocado na tabela como uma coluna, porém pode ser anotado sobre a coluna “Produto” e tratado mais tarde, na definição da dimensão.

#### 15.3.6. Definindo onde

Nesse passo o importante é contar onde a história ocorre, lembrando que, sendo um intervalo de tempo, pode ocorrer em mais de um lugar. A existência de mais de uma coluna “quando” definindo momentos de tempo diferentes dá a dica que estes momentos podem ocorrer em lugares diferentes.

A Figura 15.17 mostra a continuação do exemplo, onde são criadas duas colunas para marcar a loja onde foi feita o pedido e o endereço de entrega. É importante notar que não aparece o endereço da loja, pois seria novamente um caso de detalhe do detalhe, que não deve ser usado nesse instante.

## 15. O Método Beam\*

CLIENTE	encomenda	na	DATA
	PRODUTO	DE PEDIDO	
<who>	<what> MD	<quando> MD	
Antônio Alves	Suéter Azul	10-maio-2011	
Bia Betrand	Calça Vison	29-jun-2011	
Antônio Alves	Suéter Azul	10-maio-2011	
Carla Caril	Camisa Bege	14-out-2011	
<desconhecido>	Sapato Salto 28	10 anos atrás	

Figura 15.13.: Configuração da tabela BEAM\* com um dados faltando. Adaptado de:(Corr e Stagnitto, 2012)

### 15.3.7. Encontrando quantos

Ao chegar ao ponto da criação da história que indica “quantos” deixa-se de tratar das dimensões ao redor da Tabela Fato para tratar especificamente das informações que podem ser colocadas na mesma.

Isto porque essas colunas vão informar quantidades, que normalmente são aditivas e são exatamente os que precisamos na Tabela Fato.

Continuando com o exemplo, a história pode ser estendida com as colunas “quantidade” e “preço”, como na Figura 15.18

### 15.3. Passo a Passo da Modelagem de Eventos de Negócio

CLiENTE	encomenda PRODUTO	na DATA DE PEDIIDO
<who>	<what> MD	<quando> MD
Antônio Alves	Suéter Azul	10-maio-2011
Bia Betrand	Calça Vison	29-jun-2011
Antônio Alves	Suéter Azul	10-maio-2011
Carla Caril	Camisa Bege	14-out-2011
<desconhecido>	Sapato Salto 28	10 anos atrás
Colégio LMS	Camisa Branca	ontem
Empresa EPM	Calça+Camisa	ontem

Figura 15.14.: Configuração da tabela BEAM\* ao final deste passo. Adaptado de:(Corr e Stagnitto, 2012)

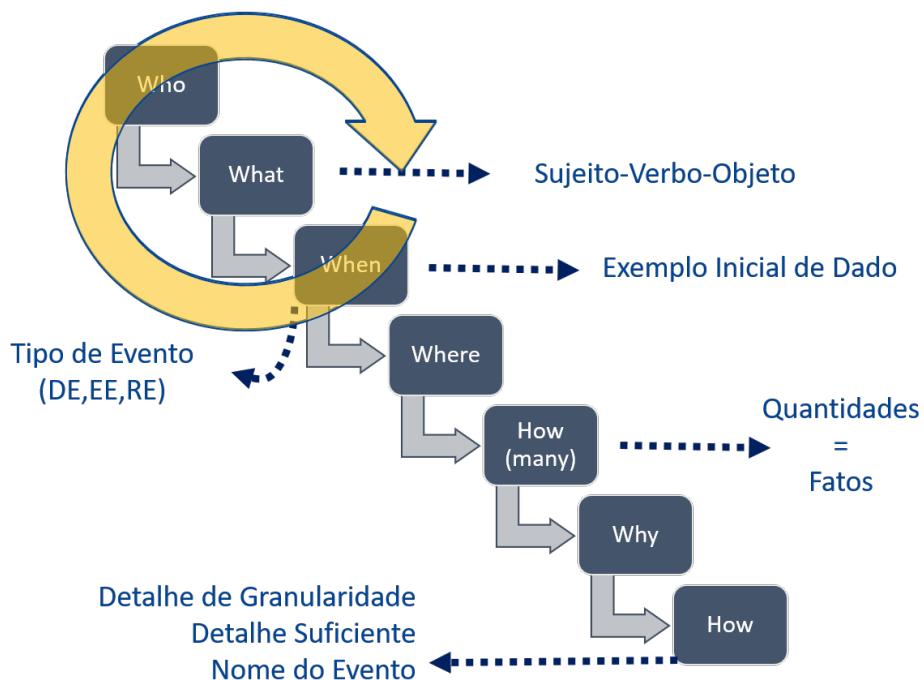


Figura 15.15.: Os três primeiros passos do processo podem ser repetidos para exaurir os conceitos de “quem” e “quanto”. Adaptado de:(Corr e Stagnitto, 2012)

## 15. O Método Beam\*

CLIENTE	encomenda	na:	para entrega em	com:
	PRODUTO	DATA DE PEDIDO	DATA DE ENTREGA	VENDEDOR
<who>	<what> MD	<when> MD	<when>	<who>
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	Zenaide Zimbro
Bia Betrand	Calça Vison	29-jun-2011	2-julho-2011	Xenia Xavier
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	Zenaide Zimbro
Carla Caril	Camisa Bege	14-out-2011	não aplicável	Yan Yakiza
<desconhecido>	Sapato Salto 28	10 anos atrás	não aplicável	William Wonder
Colégio LMS	Camisa Branca	ontem	daqui a 3 dias	N/A
Empresa EPM	Calça+Camisa	ontem	daqui a 10 dias	Equipe Comercial

Figura 15.16.: Uma tabela BEAM\* com dois “quem” e dois “quando”. (Corr e Stagnitto, 2012)

CLIENTE	encomenda	na:	para entrega em	na:	para
	PRODUTO	DATA DE PEDIDO	DATA DE ENTREGA	LOJA	ENDERECO
<who>	<what> MD	<when> MD	<when>	<where>	<where>
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	SEDE	Rua Seca 23, 22010-222
Bia Betrand	Calça Vison	29-jun-2011	2-julho-2011	amazon.com	Rua Arco Íris 123, 10021-101
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	SEDE	Rua Seca 23, 22010-222
Carla Caril	Camisa Bege	14-out-2011	não aplicável	1-800-roupas	Av. Américas 2, 20021-11
<desconhecido>	Sapato Salto 28	10 anos atrás	não aplicável	1-800-roupas	Rua Salto 199, 42921-394
Colégio LMS	Camisa Branca	ontem	daqui a 3 dias	COPA 2	Não aplicável
Empresa EPM	Calça+Camisa	ontem	daqui a 10 dias	IPANEMA	Não aplicável

Figura 15.17.: Uma tabela BEAM\* descreve uma história que acontece em dois lugares, um em cada ponto no tempo. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012)

### 15.3. Passo a Passo da Modelagem de Eventos de Negócio

CLIENTE	encomenda	na:	para entrega em:	por	
	PRODUTO	DATA DE PEDIDO	DATA DE ENTREGA	QUANTIDADE	PRECO
<who>	<what> MD	<quando> MD	<quando>	<unidade>	<R\$, US\$>
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	1	R\$ 200
Bia Betrand	Calça Vison	29-jun-2011	2-julho-2011	1	US\$ 95
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	1	R\$ 200
Carla Caril	Camisa Bege	14-out-2011	não aplicável	1	R\$ 140
<desconhecido>	Sapato Salto 28	10 anos atrás	não aplicável	50	R\$ 200
Colégio LMS	Camisa Branca	ontem	dáqui a 3 dias	100	R\$ 30400
Empresa EPM	Calça+Camisa	ontem	dáqui a 10 dias	100	R\$ 20000

Figura 15.18.: Introduzindo na tabela BEAM\* as informações quantitativas. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012)

## 15.4. Definindo por que

Este tipo de coluna pode ser um pouco mais difícil de criar porque exige dos participantes o conhecimento de possíveis motivações para o evento que tenham acontecido ao longo do período estudado e que possam estar registrada na base operacional.

Exemplos de motivações ligados a área de venda são: promoções, lançamentos de produtos, descontos, campanhas publicitárias, etc. Outras áreas podem ter motivações diferentes. Um hospital, por exemplo, pode estar estudando a procura a seu pronto atendimento, e as motivações podem estar relacionadas a temperatura ou clima do dia, existência de grandes eventos, feriados longos, etc.

Possivelmente, estas colunas são exploratórias, isto é, o conhecimento do negócio permite fazer uma premissa que esses são motivos válidos para influenciar, de alguma forma, a ocorrência dos eventos. O sistema criado permitirá analisar o verdadeiro impacto desses motivos. A Figura 15.19 mostra um exemplo onde descontos são identificados.

CLIENTE	encomenda PRODUTO	na: DATA DE PEDIDO	para entrega em: DATA DE ENTREGA	por: PROMOÇÃO	com: DESCONTO
<who>	<what> MD	<quando> MD	<quando>	<why>	<R\$, US\$>
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	Sem Promoção	0
Bia Betrand	Calça Vison	29-jun-2011	2-julho-2011	Promoção	10%
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	Sem Promoção	0
Carla Carlil	Camisa Bege	14-out-2011	não aplicável	Lançamento	US\$10
<desconhecido>	Sapato Salto 28	10 anos atrás	não aplicável	Pro Teste	R\$ 1000
Colégio LMS	Camisa Branca	ontem	daqui a 3 dias	Novo Negócio	R\$ 5000
Empresa EPM	Calça+Camisa	ontem	daqui a 10 dias	Novo Negócio	R\$ 6000

Figura 15.19.: Introduzindo na tabela BEAM\* as informações sobre por que. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012)

## 15.5. Definindo como

A última coluna a ser criada e preenchida tem relação com como a história acontece ou é registrada no sistema da empresa.

Nesses dados, por exemplo, é possível guardar como uma venda é paga, ou seja, o método de pagamento, cartão, dinheiro, etc, ou como ela é registrada, como um pedido, uma nota fiscal, etc. A Figura 15.20 mostra a Tabela BEAM\* com a identificação do número do pedido.

Este é o último passo de preenchimento das colunas.

	<i>encomenda</i>	<i>na:</i>	<i>para entrega em:</i>		<i>por</i>	
<b>CLIENTE</b>	<b>PRODUTO</b>	<b>DATA DE PEDIDO</b>	<b>DATA DE ENTREGA</b>	<b>QUANTIDADE</b>	<b>PREÇO</b>	<b>ID PEDIDO</b>
<who>	<what> MD	<quando> MD	<quando>	<unidade>	<R\$, US\$>	<how>
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	1	R\$ 200	12312
Bia Betrand	Calça Vison	29-jun-2011	2-julho-2011	1	US\$ 95	11255
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	1	R\$ 200	66343
Carla Caril	Camisa Bege	14-out-2011	não aplicável	1	R\$ 140	23435
<desconhecido>	Sapato Salto 28	10 anos atrás	não aplicável	50	R\$ 200	25232
Colégio LMS	Camisa Branca	ontem	daqui a 3 dias	100	R\$ 30400	63345
Empresa EPM	Calça+Camisa	ontem	daqui a 10 dias	100	R\$ 20000	53435

Figura 15.20.: Introduzindo na tabela BEAM\* as informações sobre como. Por espaço algumas colunas já encontradas foram suprimidas. Fonte:(Corr e Stagnitto, 2012)

### 15.5.1. Escolhendo a Granularidade

Com todas as colunas definidas e todos os exemplos colocados, temos a Tabela BEAM\* praticamente completa para uma história. O próximo passo é definir a granularidade

O objetivo é escolher um conjunto de colunas que crie uma identificação única do evento, no nível de detalhe desejado. Pode ser que uma única coluna já garanta, ou que mais de uma seja necessária. Essa escolha é anotada na Tabela BEAM\* com o código **GD**, para o inglês *granularity detail*.

Se, definido um nível de detalhe, seja possível criar mais de uma linha na Tabela BEAM\* mostrando exemplos que para os quais as colunas escolhidas tem todos o mesmo valor, é necessário verificar se eles podem ser agregados de alguma forma.

No exemplo deste capítulo, foram escolhidas as colunas PRODUTO e ID PEDIDO. Isto significa que se em um pedido um produto aparecer duas vezes, todos os valores devem ser agregados.

## 15. O Método Beam\*

### 15.5.2. Dando nome ao evento

O próximo e último passo desta parte do processo BEAM\* é dar o nome a história de dados e identificar o tipo do evento. A Tabela 15.2 mostra novamente os nomes dos tipos de eventos com seus códigos.

Tabela 15.2.: Tipos de eventos e seus códigos. Baseado em:(Corr e Stagnitto, 2012)

Tipo de Evento	Código
Evento Discreto	ED
Evento Evolutivo	EE
Evento Recorrente	RE

Para indicar que a Tabela BEAM\* está totalmente definida, são traçadas duas linhas na sua margem direita. A Figura 15.21 mostra a versão final.

PEDIDO DE CLIENTE <DE>						
CLIENTE	encomenda	na:	para entrega em			
	PRODUTO	DATA DE PEDIDO	DATA DE ENTREGA	QUANTIDADE	PREÇO	ID PEDIDO
<who>	<what> MD   GD	<quando> MD	<quando>	<unidade>	<R\$, US\$>	< how >   GD
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	1	R\$ 200	12312
Bia Betrand	Calça Vison	29-jun-2011	2-julho-2011	1	US\$ 95	11255
Antônio Alves	Suéter Azul	10-maio-2011	12-maio-2011	1	R\$ 200	66343
Carla Caril	Camisa Bege	14-out-2011	não aplicável	1	R\$ 140	23435
<desconhecido>	Sapato Salto 28	10 anos atrás	não aplicável	50	R\$ 200	25232
Colégio LMS	Camisa Branca	ontem	daqui a 3 dias	100	R\$ 30400	63345
Empresa EPM	Calça+Camisa	ontem	daqui a 10 dias	100	R\$ 20000	53435

Figura 15.21.: Finalizando a Tabela BEAM\* com nome e tipo. Fonte:(Corr e Stagnitto, 2012)

Além de manter um registro fotográfico detalhado do passo a passo, é aconselhável documentar a Tabela BEAM\* final em uma planilha eletrônica. Esta planilha pode ser usada para continuar o trabalho nos próximos passos.

### 15.5.3. Resumo do passo a passo

Resumindo o passo a passo, é importante:

- iniciar sempre pela história prioritária no momento;
- realizar os três primeiros passos, que definem quem-o que-quando até esgotar as possibilidades;
- definir os outros detalhes na ordem prescrita;
- definir a granularidade da tabela;
- dar o nome e determinar o tipo da tabela, e
- registrar fotograficamente e, possivelmente, copiar a tabela final para uma planilha eletrônica.

## 15.6. Modelando Dimensões

O próximo grande passo do método BEAM\* visa definir as dimensões a partir dos detalhes dos eventos. Nesse passo, então, os detalhes serão transformados em dimensões, por meio da elicitação e definição de todos os campos de cada dimensão.

Para isso são realizados quatro passos:

1. iniciar a tabela dimensão com os dados levantados na Tabela BEAM\*;
2. especificar os atributos da dimensão;
3. especificar as hierarquias existentes nas dimensões, e
4. Controlar como as colunas podem mudar.

Esse subprocesso, Modelar Dimensão, do processo Modelstorming definido na Figura 15.3, está descrito na Figura 15.22

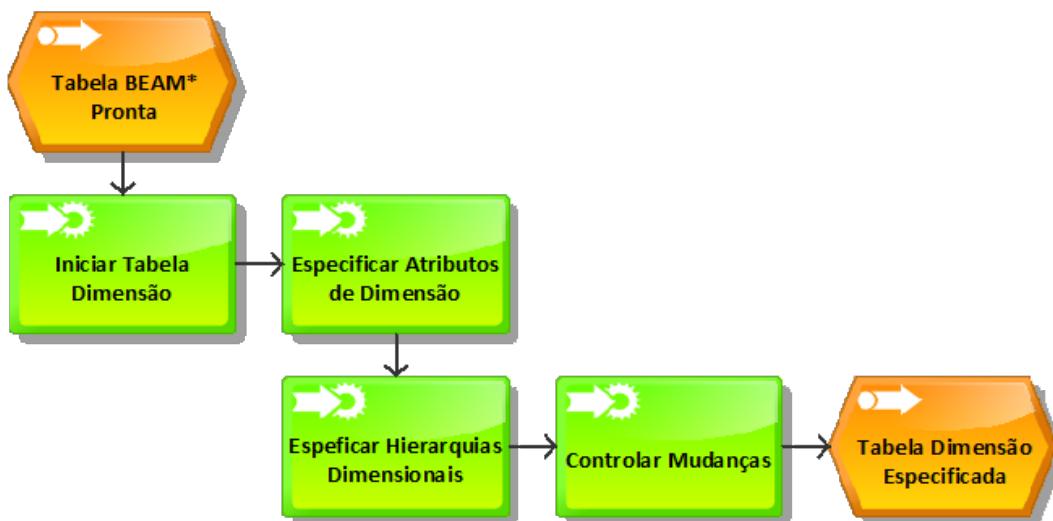


Figura 15.22.: Modelar Dimensão descrito em ARIS-EPC.

No caso do exemplo, os detalhes que serão considerados como parte da história são:

- cliente (quem),
- produto (o que),

## 15. O Método Beam\*

- data do pedido (quando),
- data da entrega (quando),
- vendedor (quem),
- loja (onde),
- endereço de entrega (onde),
- promoção (por que), e
- id pedido (como).

Normalmente, em busca das dimensões, **não são tratadas** os detalhes do tipo **quanto**, que deverão pertencer a Tabela Fato.

### 15.6.1. Codificação Usada

Na prática, serão criadas tabelas que mostram como é cada dimensão, semelhantes as Tabelas BEAM\*. e as colunas destas tabelas serão anotadas com alguns códigos que permitem entender melhor algumas de suas características.

O códigos apresentados na Tabela 15.3 são usados neste passo.

Tabela 15.3.: Códigos usados na modelagem dos campos da dimensão

Código	Significado	Explicação
MD	<b>Mandatory</b>	Campo obrigatório
BK	<b>Business Key</b>	Chave única do negócio que identifica o membro da dimensão
DC	<b>Defined by Characteristic</b>	Campo que precisa ser definido por uma característica, um tipo
X1, X2, ...	<b>eXclusive</b>	Campos exclusivos de alguns tipo e não usados para outros
FV	<b>Fixed Value</b>	Única mudança possível é uma correção
HV	<b>Historical Value</b>	Mudanças acontecem e são rastreadas
CV	<b>Current Value</b>	Dispensa histórico, basta o valor atual

Partindo da Tabela BEAM\* serão criadas novas tabelas que representam as dimensões. A ordem de leitura da Tabela BEAM\* para gerar as Tabelas Dimensão é da esquerda para a direita, logo, no exemplo, é preciso começar pela dimensão CLIENTE.

Definida a Tabela Dimensão que será elicitada, é feito então um novo passo a passo para gerar a definição dos seus campos.

## 15.6.2. Início da tabela

O primeiro passo é transferir a informação usada na Tabela BEAM\* para o início de um novo desenho.

Desta forma se inicia a Tabela Dimensão dando um nome a mesma (Cliente), normalmente usando o próprio nome definido na Tabela BEAM\* para o detalhe. Os exemplos usados na Tabela BEAM\* são, a seguir, transferidos e usados para definir o nome do campo.

No exemplo, o nome escolhido é **Nome do Cliente**. Nesse momento, como o trabalho ainda é feito com os usuários, deve-se pensar apenas em nomes lógicos e não em regras arbitrárias usadas na organização no processo de molagem física do banco de dados.

Definido o nome da coluna, usa-se uma linha abaixo dele para fazer marcações que indicam características dessa coluna. No caso de “Nome do Cliente” o exemplo exige que o campo seja **mandatório**, o que indicado pela notação **MD**. Esse trabalho pode ser verificado na Figura 15.23.

<b>Cliente</b>		
<b>Nome do Cliente</b>		
<b>MD</b>		
Antônio Alves		
Bia Bertrand		
Carla Caril		
Unknown		
Colégio LMS		
Empresa EPM		

Figura 15.23.: Início da definição da dimensão Cliente. Fonte: (Corr e Stagnitto, 2012)

### 15.6.3. Granularidade e chaves

A partir da definição do campo e da informação principal que o caracteriza, é necessário definir a granularidade e, por consequência, as colunas que definem a chave principal do campo.

Esta chave deve possuir as características de serem únicas, estáveis e obrigatórias. Além disso, é essencial que sejam chaves ligadas ao negócio. São marcadas com o código **BK**, significando **Business Key**.

Para definir a chave de negócios, o exemplo escolhe o identificador único que um cliente possui na organização sendo modelada, o que pode ser visto na Figura 15.24. Normalmente toda chave é mandatária.

### Cliente

Nome do Cliente	ID Cliente
<b>MD</b>	<b>BK, MD</b>
Antônio Alves	C0012
Bia Bertrand	C9533
Carla Caril	C2321
Unknown	N/A
Colégio LMS	B0012
Empresa EPM	B2323

Figura 15.24.: Definição da chave de negócios para a dimensão Cliente. Fonte: (Corr e Stagnitto, 2012)

Estes dois passos definem o início da Tabela Dimensão. Os próximos passos buscam a completar a informação para a Dimensão poder auxiliar as consultas das partes interessadas.

### 15.6.4. Os Atributos de Dimensão

Para descobrir que atributos são necessários para a Tabela Dimensão, é importante pesquisar como essas informações serão usadas, isto é, quais os dados são interessantes

para estar nos relatórios, gráficos e consultas em geral que serão obtidas do Data Warehouse.

Os participantes devem buscar atributos fazendo as seguintes perguntas:

- que atributos serão usados como colunas em relatórios;
- que atributos serão usados para ordenar relatórios;
- que atributos serão usados para filtrar relatórios, e
- que atributos serão, e usados para agrupar itens de relatório e fazer totalizações ou outras operações de agregação, como média, porcentagem, etc?

Também é importante aplicar a cada dimensão as perguntas 5W2H, como:

- quem está associada a esse item,
- que datas são importantes para esse item,
- onde está esse item,
- que quantidades definem item, e
- por que esse item aparece?

Essas perguntas são apenas exemplo. No caso de uma Venda, por exemplo, na Dimensão “Cliente”, é possível perguntar onde mora o cliente, em que data ele nasceu, quantas pessoas há na sua família, etc.

## Atributos de Tipo

Alguns tipos de atributos podem ser buscados ativamente. Atributos muito importantes são aqueles que classificam o assunto do domínio de alguma forma, ou seja, aqueles que definem um “tipo de” em relação ao atributo. Exemplos são tipo de cliente, marca de produto, tipo de exame, etc.

A Figura 15.25 mostra a escolha de um Tipo de Cliente. O atributo foi definido como mandatório. Os tipos identificados são “Consumidor” e “Empresa”.

## Atributos Múltiplos

Alguns atributos, descrito informalmente, podem parecer atributos múltiplos. Um exemplo claro disso é a possibilidade de um mesmo evento de negócios usar vários atributos do tipo “endereço”, como “endereço de entrega” e “endereço de cobrança”.

Quando o número de atributos múltiplos é fixo, como no caso exemplificado, eles podem ser colocados na mesma tabela dimensão, pois não é problema a quantidade de campos que a tabela tem.

Porém, quando o atributo tem múltiplos valores e não é possível determinar sua cardinalidade, na prática eles não podem fazer parte da dimensão como definida no momento. A solução pode ser colocá-los em outra dimensão, colocá-los em outro modelo dimensional ou alterar a granularidade da dimensão.

## Cliente

Nome do Cliente	ID Cliente
<b>MD</b>	<b>BK, MD</b>
Antônio Alves	C0012
Bia Bertrand	C9533
Carla Caril	C2321
Unknown	N/A
Colégio LMS	B0012
Empresa EPM	B2323

Figura 15.25.: Definição de um tipo para a Dimensão Cliente. Fonte: (Corr e Stagnitto, 2012)

Kimball et al. (2008) cita o exemplo do caso da Tabela Fato “saldo de conta de banco”. Esta Tabela Fato se relaciona de forma N:1 com a Tabela Dimensão “Conta de Banco”. Porém, em uma conta de banco deve haver um, mas podem haver mais correntistas, o que leva a impossibilidade de colocar esse dados, quem possui a conta, na tabela. A solução proposta, uma extensão ao Modelo Estrela, é usar Dimensões M:N, na forma de Tabelas Ponte(Kimball et al., 2008, p.6.4).

### Atributos exclusivos

Quando a dimensão tem um tipo, é possível que o valor do tipo defina a necessidade de um atributo que só existe se o item for daquele tipo. Por exemplo, no caso do exemplo, o “Cliente” do tipo “Consumidor” tem um atributo “Sexo”, enquanto o “cliente” do tipo “Empresa” em um atributo “CNPJ”.

Quando isso acontece, é necessário não só criar colunas para os atributos, mas também definir que coluna define a sua existência e associá-los a grupos distintos. Para isso se usa a marcação **DC**, para **Definition Column**, na coluna que define o tipo que divide os atributos, e a notação **X1,X2,...** para as colunas que são exclusivas de um tipo. O exemplo citado aparece na Figura 15.26.

### Cliente

Nome de Cliente	ID Cliente	Tipo de Cliente	Sexo	CNPJ
MD	BK, MD, DC	MD	X1, MD	X2, MD
Antônio Alves	C0012	Consumidor	M	N/A
Bia Bertrand	C9533	Consumidor	F	N/A
Carla Caril	C2321	Consumidor	F	N/A
Unknown	N/A	N/A	Desconhecido	N/A
Colégio LMS	B0012	Empresa	N/A	23929939291-1299
Empresa EPM	B2323	Empresa	N/A	40912839120-1453

Figura 15.26.: Definição dos atributos exclusivos. Fonte: (Corr e Stagnitto, 2012)

### 15.6.5. Prevendo mudanças

A última anotação em uma Tabela de Dimensão é a previsão de mudanças no mesmo. Isso é importante e Kimball et al. (2008) propõe vários métodos distintos para o tratamento de mudanças que podem ser aplicados após esse passo.

Cada coluna deve receber uma marcação adicional que indica como podem ocorrer mudanças na mesma. Os valores possíveis são:

**FV**, ou *Fixed Value*, que indica que o valor é fixo e mudanças podem ocorrer apenas por motivos de correção;

**HV** ou *Historical Value*, onde ocorrerão mudanças e todas devem ser rastreadas, o que pode exigir campos adicionais e outras estratégias de controle;

**CV** ou *Current Value*, onde o valor é alterado, mas só é preciso guardar o valor atual, e

**HV/CV ou CV/HV** onde há uma combinação de comportamentos segundo regras de negócio, porém o primeiro valor é o default.

No exemplo em questão, todos os valores são fixos. Em uma dimensão “Produto”, por exemplo, o “preço do produto” poderia ser definido como **CV** ou **HV** ou combinações, dependendo da necessidade das partes interessadas na organização.

### 15.6.6. Construindo a hierarquia da dimensão

A partir da tabela de dimensão, e possivelmente provando correções na mesma, é necessário agora documentar as hierarquias que existem nesse dimensão. É importante notar que o Método BEAM\* segue o Modelo Estrela, e não pretende criar o Snowflake, mas se for desejado pela organização, este é o ponto em que o modelo pode ser adaptado para levantar as hierarquias.

Existem vários tipos de hierarquia, como mostra a Figura 15.27. Nela aparecem hierarquias com pai único e com múltiplos pais, e hierarquias balanceadas, desequilibradas e de profundidade variável.

## 15. O Método Beam\*

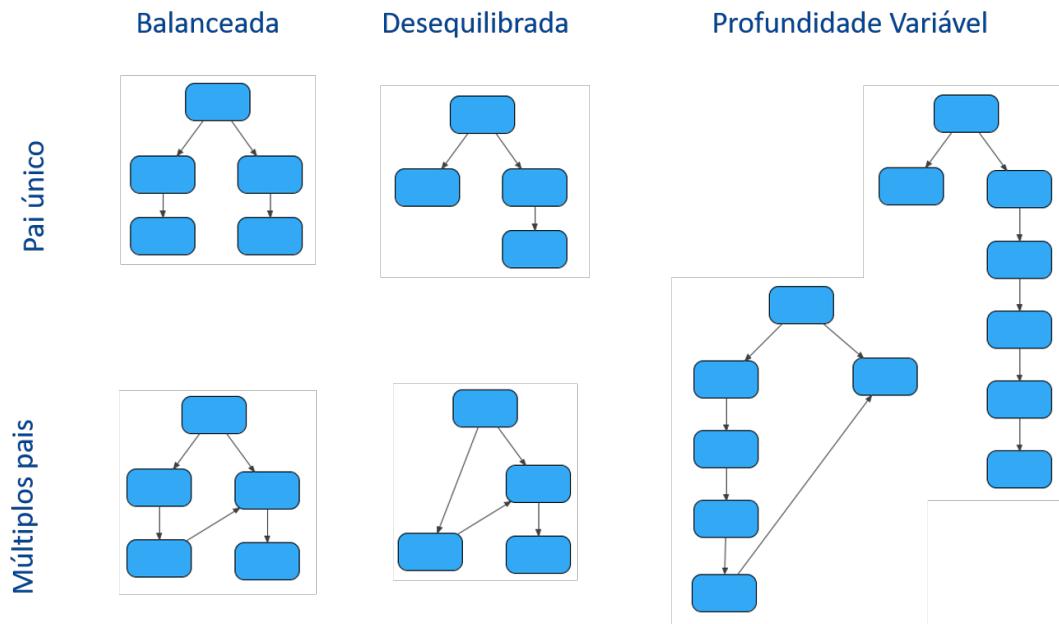


Figura 15.27.: Tipos de hierarquias para dimensões. Fonte: (Corr e Stagnitto, 2012)

Mapas de hierarquia mostram de maneira simples essas hierarquia e podem ser vistos nas Figuras 15.28 e 15.29. A Figura 15.29, especialmente, mostra duas notações alternativas para hierarquias com *loops*.

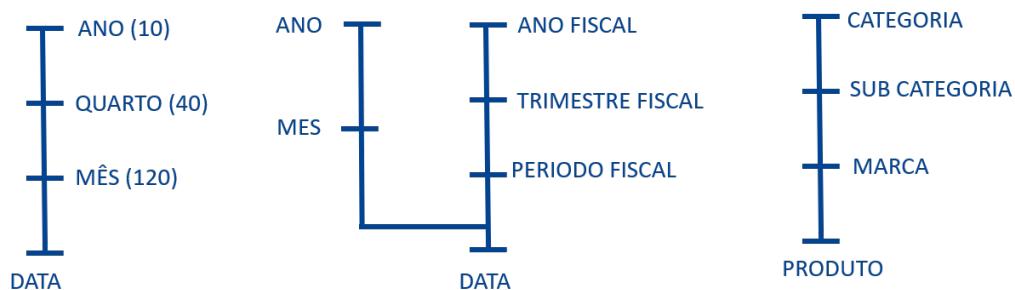


Figura 15.28.: Hierarquias para dimensões de data, com duas versões, e produto. Fonte: (Corr e Stagnitto, 2012)

Esses mapas podem ser usados para marcar como são feitas as consultas esperadas, como na Figura 15.30.

Para levantar as hierarquias é importante saber se os itens inferiores podem pertencer a mais um item superior, ou se um item superior pode ter mais de um item inferior simultaneamente.



Figura 15.29.: Notações para hierarquias com loops. Fonte: (Corr e Stagnitto, 2012)

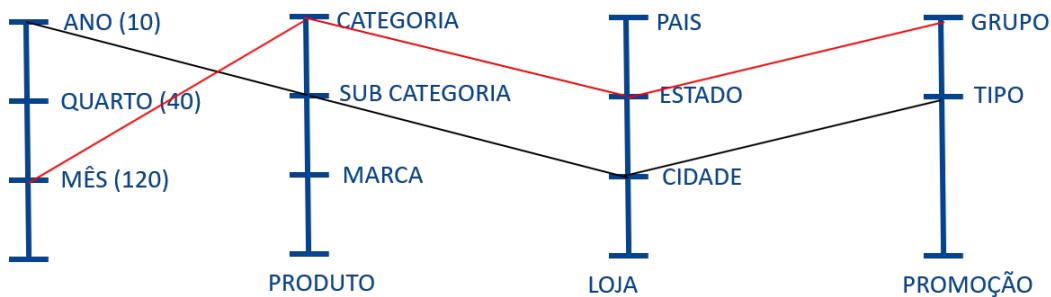


Figura 15.30.: Notações para consultas perpassando hierarquias. Fonte: (Corr e Stagnitto, 2012)

## 15.7. Matriz de Evento

Corr e Stagnitto (2012) apresenta o seguinte exemplo de Matriz de Evento.

## 15. O Método Beam\*

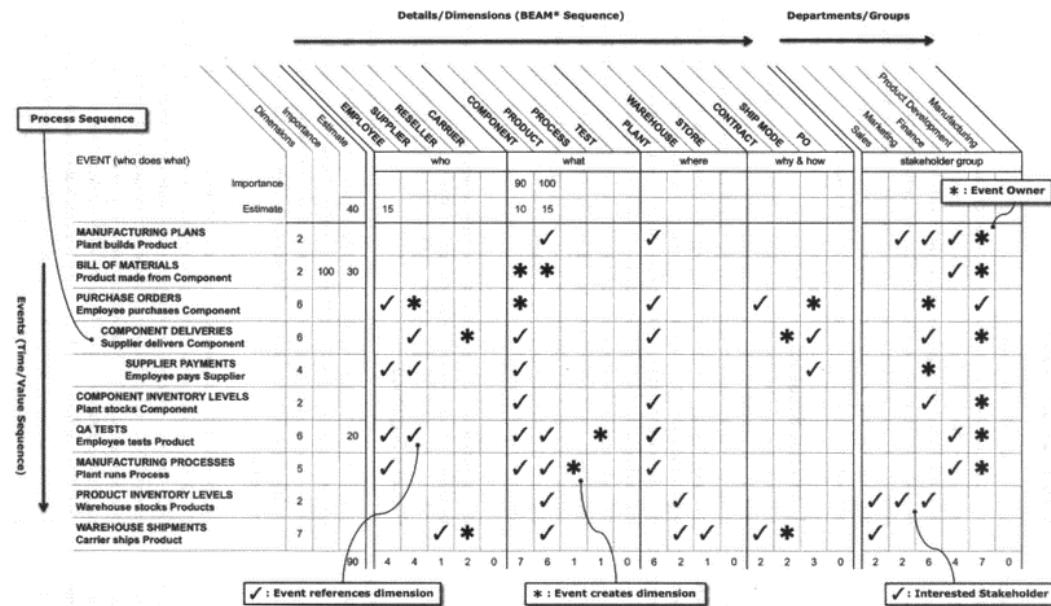


Figura 15.31.: Exemplo de Matriz de Evento. Fonte: (Corr e Stagnitto, 2012)

# Índice Remissivo

- future value*, 69  
*present value*, 69  
100-pontos, 63  
5W2H, 79  
5w2h, 79  
7W, 167  
  
abstração, 87, 88  
abstração de caixa-preta, 91  
assuntos, 140  
atributos, 9, 29  
  
Banco de Dados Relacional, 27  
Beam\*, 167  
benefício, 74  
BI, 138  
Bill Inmon, 139  
BK, 192  
Business Event Analysis and Modelling, 167  
Business Intelligence, 138  
Business Key, 192  
  
caixa-branca, 91  
Carta de Hierarquia, 168  
classe, 91  
classificar, 92  
classificação, 91  
  
coisas, 13  
Commercial Of The Shelf, 113, 132  
composição, 91, 93  
condição, 95  
COTS, 113, 132  
CREATE TABLE, 32  
custo de oportunidade, 55  
  
dados, 98  
Data Control Language, 31  
Data Definition Language, 31  
Data Definition Languagem, 31  
Data Manipulation Language, 31  
Data Marts, 137  
Data Query Language, 31, 33  
Data Transaction Language, 31  
Data Warehouse, 136, 140  
DC, 194  
DCL, 31  
DDL, 31  
decomposição, 94  
Definition Column, 194  
demanda, 57  
estados, 57  
DESCOBRIR, 13  
descrições, 13  
desejo, 56  
detalhes, 173, 176

## ÍNDICE REMISSIVO

- dice, 152  
dinheiro de brinquedo, 63  
DML, 31  
domínio, 9, 29  
DQL, 31  
drill across, 147  
drill down, 156  
drill down and roll up, 147  
drill up, 156  
DROP TABLE, 33  
DTL, 31  
  
entidade, 11  
entidades, 8  
Entreprise Resource Planning, 113, 133  
ER, 5  
ERP, 113, 123, 133  
especialização, 93  
especificações, 12  
esquema de relação, 29  
Evento de Negócio, 170  
evento discreto, 171  
evento evolutivo, 171  
evento recorrente, 171  
eventos, 12  
  
filtro, 152  
foco, 95  
foco/inibição, 91  
Framework 7W, 168  
Framework de Zachman, 82  
função utilidade, 55  
  
GD, 187  
generalização, 91, 93  
grau da relação, 30  
grupo, 179  
  
hiperonímia, 93  
hiponímia, 93  
histórias de dados, 170  
holónímia, 94  
holônimo, 94  
  
identificação, 91, 94  
  
informação, 97  
inibição, 95  
instanciação, 92  
instância, 8, 91  
instância de entidade, 8  
interativo, 123  
interações, 12  
intervalos, 13  
IRACIS, 74  
  
junção, 33  
juro, 69  
  
Kano, 64  
  
Linhas de Tempo, 168  
locais, 13  
  
mandatório, 191  
markup, 54  
Matriz de Eventos, 168  
medida, 163  
MER, 5  
meronímia, 94  
merônimo, 94  
metas, 111  
missão, 111  
modelagem dimensional, 163  
modelagem ágil de dados, 167  
modelo, 87, 88  
Modelo de Entidades e Relacionamentos, 5, 7  
modelo estrela, 163  
Modelo Estrela Aumentado, 168  
Modelo Relacional, 27  
momentos, 13  
montante, 69  
MoSCoW, 63  
  
necessidade, 56  
  
objetivos, 111  
objetos tangíveis, 12  
ocultação da informação, 91  
OLAP, 147  
OLTP, 147

## ÍNDICE REMISSIVO

- Online Analytical Processing, 147  
Online Transaction Processing, 147  
operações OLAP, 147  
ordem de descoberta, 169  
organização, 108  
organograma, 114  
  
pagamento, 69  
papéis, 13  
papéis exercidos, 12  
pessoas, 13  
pivot, 147, 160  
pivot table, 149  
planilhas eletrônicas, 149  
prazo, 69  
prestação, 69  
principal, 69  
princípio da diminuição da utilidade marginal, 55  
processamento de transações em tempo real, 147  
  
Ralph Kimball, 139  
reativo, 123  
refinamento sucessivo, 91, 95  
relação, 30  
remuneração, 69  
roll up, 156  
  
saldo, 69  
satisfação, 57  
separação de interesses, 91, 96  
SGDB, 31  
short codes BEAM\*, 169  
simplificação pelo Caso Normal, 91  
  
simplificação pelo caso normal, 95  
sistema, 101, 122  
Sistemas de Informação, 109  
sistemas de informação, 122  
sistemas de respostas planejadas, 123  
sistemas fonte, 144  
Sistemas Gerenciadores de Banco de Dados, 31  
slice, 152  
slice and dice, 147, 152  
SQL, 31  
Structured Query Language, 31  
  
Tabela Beam\*, 168, 169  
tabela dimensão, 163  
tabela fato, 141, 163  
tabelas dimensão, 141  
taxa, 69  
taxa de juros, 69  
tema, 178  
tipo de dados, 29  
tipo de entidade, 8  
  
utilidade, 55  
  
valor, 53, 57, 74  
valor atual, 69  
valor de resgate, 69  
valor descontado, 69  
valor futuro, 69  
valor presente, 69  
visão, 111  
  
X1, 194  
X2, 194



## Bibliografia

- Adiel Teixeira de Almeida Danielle Morais, Hannu Nurmi (2019). *Systems, Procedures And Voting Rules In Context: A Primer For Voting Rule Selection.* Advances In Group Decision And Negotiation Vol. 9. Springer.
- Almquist, Eric, Jamie Cleghorn e Lori Sherer (2018). “The B2B Elements of Value”. *Harvard Business Review*, pp. 72–81. URL: <https://hbr.org/2018/03/the-b2b-elements-of-value> (acesso em 09/02/2020).
- Almquist, Eric, John Senior e Nicolas Bloch (1 de set. de 2016). “The Elements of Value”. *Harvard Business Review*, pp. 46–53. URL: <https://hbr.org/2016/09/the-elements-of-value> (acesso em 09/02/2020).
- American Heritage (2019). *The American Heritage Dictionary of the English Language*. URL: <https://www.ahdictionary.com/> (acesso em 25/12/2019).
- Amy, Douglas J. (2000). *Behind the Ballot Box: A Citizen’s Guide to Voting Systems*. Greenwood Publishing Group.
- Anklesaria, Jimmy (2008). *Supply Chain Cost Management. The AIM & DRIVE Process for Achieving Extraordinary Results*. New York: Amacom - American Management Association.
- Bertalanffy, Ludwig von (1969). *General System Theory. Foundations Development Applications*. revised. George Braziller, Inc.
- Bertini, C., S. Ceri e Shamkant B. Navathe (1992). *Conceptual Database Design*. The Benjamin/Cummings Publishing Company.
- BeSeen (17 de ago. de 2015). *The value triangle – managing customers’ expectations*. BeSeen. URL: <https://www.beseen-marketing.co.uk/marketing-blog/the-value-triangle> (acesso em 09/02/2020).
- Biffl, Stefan et al., ed. (2006). *Value-Based Software Engineering*. Berlin, Heidelberg: Springer.
- Boehm, Barry W. et al. (2000). *Software Cost Estimation with Cocomo II with Cdrom*. 1st. USA: Prentice Hall PTR. ISBN: 0130266922.

## Bibliografia

- Brooks, Frederick P. (1978). *The Mythical Man-Month. Essays on Software Engineering.* 1<sup>a</sup> ed. USA: Addison-Wesley Longman Publishing Co., Inc. ISBN: 0201006502.
- Bunge, Mario (1979). *Treatise on Basic Philosophy - Ontology II: A World of Systems.* Vol. 4. Treatise on Basic Philosophy. Springer Netherlands. 314 pp. ISBN: 978-90-277-0944-8.
- Cairncross, A. (1951). *Introduction to Economics.* Butterworth.
- Celko, Joe (2005). *Joe Celko's SQL for smarties: advanced SQL programming.* Third. The Morgan Kaufmann series in data management systems. Los Altos, CA 94022, USA: Morgan Kaufmann Publishers, pp. xxviii + 808. ISBN: 0-12-369379-9 (paperback). URL: <http://www.loc.gov/catdir/enhancements/fy0626/2005279919-d.html>; <http://www.loc.gov/catdir/enhancements/fy0626/2005279919-t.html>.
- Chaitin, Gregore (mar. de 2006). “The Limits of Reason”. *Scientific American.*
- Chen, Peter (1990). *Modelagem de Dados: A abordagem entidade-relacionamento para projeto lógico.* São Paulo: Makron Books.
- Chiavenato, Idalberto (2014). *Introdução a Teoria Geral da Administração.* 9<sup>a</sup> ed. Barueri, SP: Editora Manole.
- Claude E Shannon, Warren Weaver (1963). *The Mathematical Theory of Communication.* University of Illinois Press.
- Coad, Peter, Jeff de Luca e Eric Lefebvre (1999). *Java Modeling Color with Uml: Enterprise Components and Process with Cdrom.* 1st. USA: Prentice Hall PTR. ISBN: 013011510X.
- Codd, E. F. (jun. de 1970). “A relational model of data for large shared data banks”. *Communications of the ACM* 13.6, pp. 377–387. doi: 10.1145/362384.362685.
- Cohn, Mike (2004). *User Stories Applied: For Agile Software Development.* USA: Addison Wesley Longman Publishing Co., Inc. ISBN: 0321205685.
- (2005). *Agile Estimating and Planning.* Prentice Hall.
- Corr, Lawrence e Jim Stagnitto (2012). *Agile Data Warhouse Design. Agile Data Warhouse Design: Collaborative Dimensional Modeling from Whiteboard to Star Schema. Collaborative Dimensional Modeling from Whiteboard to Star Schema.* Leeds, UK: Decision One Press.
- Cougo, Paulo (1999). *Modelagem Conceitual e Projeto de Banco de Dados.* Rio de Janeiro: Campus.
- Dal Zot, Wili e Manuela Longoni de Castro (2015). *Matemática Financeira: fundamentos e aplicações.* Porto Alegre: Bookman, p. 151.
- Dalkey, Norman Crolee (out. de 1966). *Delphi.* Technical Report P-3704. Santa Monica, California: RAND Corporation. URL: <https://www.rand.org/pubs/papers/P3704.html> (acesso em 16/01/2020).
- Date, C. J. (2004). *An introduction to database systems.* Eighth. Boston, MA, USA: Pearson/Addison Wesley, pp. xxvii + 983 + 22. ISBN: 0-321-19784-4.
- Drucker, Peter F. (1974). *agement: Tasks, Responsibilities, Practices.* New York: Truman Talley Books.
- Elmasri, Ramez e Shamkant B. Navathe (2016). *Fundamentals of Database Systems.* 7th Edition. Boston: Pearson.

- Erdogmus, Hakan, John Favaro e Michael Halling (2006). "Valuation of Software Initiatives Under Uncertainty: Concepts, Issues, and Techniques". Em: *Value-Based Software Engineering*. Ed. por Stefan Biffl et al. Berlin, Heidelberg: Springer.
- Extra (18 de jun. de 2019). *Se quer levar mais de 10 quilos, pague, sem problema nenhum', diz Bolsonaro sobre fim do despacho gratuito*. URL: <https://extra.globo.com/noticias/economia/se-quer-levar-mais-de-10-quilos-pague-sem-problema-nenhum-diz-bolsonaro-sobre-fim-do-despacho-gratuito-23747656.html> (acesso em 30/01/2020).
- Fedotov, Alex (22 de fev. de 2019). *Septem Circumstantiae, five W's and H or 'six serving-men'*. URL: <https://alxfed.github.io/blog/posts/2019/02/22/Septem-Circumstantiae.html> (acesso em 08/01/2020).
- Franceschini, Fiorenzo (2016). *Advanced Quality Function Deployment*. CRC Press, p. 208. ISBN: 9781420025439.
- Gane, Chris P. e Trish Sarson (1979). *Structured Systems Analysis: Tools and Techniques*. 1<sup>a</sup> ed. Prentice Hall Professional Technical Reference. ISBN: 0138545472.
- Gibbs, Wayt W. (set. de 1994). "Software's Chronic Crisis". *Scientific American*, pp. 86–100.
- Greenlaw, Steven A., David Shapiro e Timothy Taylor (2017). *Principles of Microeconomics fpr AP ®Courses*. 2<sup>a</sup> ed. OpenStax, Rice University.
- Hayes, Adam (25 de jun. de 2019). *Internal Rate of Return – IRR*. Investopedia. URL: <https://www.investopedia.com/terms/i/IRR.asp> (acesso em 08/02/2020).
- Heuser, Carlos A. (2001). *Projeto de Banco de Dados*. Vol. 4. Série Livros Didáticos: Instituto de Informática da UFRGS. Porto Alegre: Editora Sagra Luzzatto.
- Higgins, J.M. (1994). *101 Creative Problem Solving Techniques: The Handbook of New Ideas for Business*. New Management Publishing Company. ISBN: 9781883629007. URL: [https://books.google.com.br/books?id=Q1%5C\\_9LcYV03kC](https://books.google.com.br/books?id=Q1%5C_9LcYV03kC).
- Holanda Ferreira, Aurélio Buarque de (1986). *Novo Dicionário da Língua Portuguesa*. 2<sup>a</sup> ed. Nova Fronteira.
- Houaiss, Antônio, Mauro Villar e Francisco Manoel de Mello Franco (2009). *Dicionário Houaiss da língua portuguesa*. Objetiva. ISBN: 978-972-759-664-5.
- IFPUG (2010). *Function Point Counting Practices Manual Release 4.3.1*. Standard. International Function Point Users Group.
- IIBA (2011). *Um guia para o Corpo de Conhecimento de Análise de Negócios (Guia BABOK) Version 2.0*. Ontário, Canadá: International Institute of Business Analysis.
- INEP, Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (5 de set. de 2016). *Microdados do Exame Nacional do Ensino Médio - Enem 1998*. <http://portal.inep.gov.br/basica-levantamentos-acessar>. Acessado em 25-10-2109.
- Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira - INEP. URL: [http://ftp.inep.gov.br/microdados/micro\\_enem1998.zip](http://ftp.inep.gov.br/microdados/micro_enem1998.zip) (acesso em 26/10/2019).
- Inmon, William H. (1992). *Building the Data Warehouse*. Wellesley, MA, USA: John Wiley & Sons. ISBN: 0-89435-404-3.
- (2005). *Building the Data Warehouse*. Fourth edition. Wiley Publishing, Inc. ISBN: 978-0-7645-9944-6.

## Bibliografia

- ISO/IEC (jul. de 2004). *ISO/IEC 5218:2004 Information technology — Codes for the representation of human sexes*. ISO/IEC.
- Kempe, Shannon e Paul Williams (ago. de 2012). *A Short History of Data Warehousing*. <https://www.dataversity.net/a-short-history-of-data-warehousing>. Acessado em 25/10/2019. URL: <https://www.dataversity.net/a-short-history-of-data-warehousing/>.
- Kenton, Will (25 de jun. de 2019). *Net Present Value (NPV)*. Investopedia. URL: <https://www.investopedia.com/terms/n/npv.asp> (acesso em 08/02/2020).
- Kerzner, Harold (2017). *Project Management. A system approach to planning, scheduling and controlling*. 12<sup>a</sup> ed. Hoboken, New Jersey: John Wiley & Sons.
- Kim, W.C. e R. Mauborgne (2005). *A Estratégia Do Oceano Azul*. Elsevier. ISBN: 9788535215243.
- Kimball, Ralph et al. (2008). *The Data Warehouse Lifecycle Toolkit: Practical Techniques for Building Data Warehouse and Business Intelligence Systems*. 2nd. Wiley.
- King, J. E. e Michael McLure (2014). *History of the Concept of Value*. Discussion Paper 14.06. The University of Western Australia, Department of Economics. URL: <https://ideas.repec.org/p/uwa/wpaper/14-06.html>.
- Kotler, Philip, Gary Armstrong et al. (2017). *Principles of Marketing*. 7th European Edition. Pearsppm.
- Kotler, Philip e Kevin Lana Keller (2012). *Marketing Management*. 14<sup>a</sup> ed. Boston: Prentice Hall.  
– (2013). *Administração de Marketing*. 14<sup>a</sup> ed. São Paulo: Pearson.
- Krugman, Paul e Robin Wells (2013). *Microeconomics*. 3<sup>a</sup> ed. New York: Worth Publishers.
- Laudon, Kenneth e Jane Laudon (2011). *Sistemas de Informação Gerenciais*. 9<sup>a</sup> edição. São Paulo: Pearson.
- Leffingwell, Dean e Don Widrig (1999). *Managing Software Requirements: A Unified Approach*. USA: Addison-Wesley Longman Publishing Co., Inc. ISBN: 0201615932.
- McMenamin, Stephen M. e John F. Palmer (1984). *Essential Systems Analysis*. USA: Yourdon Press. ISBN: 0917072308.
- Meskanen, T e H Nurmi (2006). “Distance from Consensus: A Theme and Variations”. Em: *Mathematics and Democracy. Studies in Choice and Welfare*. Ed. por Pukelsheim F. Simeone B. Berlin, Heidelberg: Springer. DOI: 10.1007/3-540-35605-3\_9.
- Microsoft (27 de out. de 2019). *Microsoft Excel 365*. [Computer Software]. M.
- Moorman, Jan (9 de out. de 2012). *Leveraging the Kano Model for Optimal Results*. Article No :882. UX Magazine. URL: <https://uxmag.com/articles/leveraging-the-kano-model-for-optimal-results> (acesso em 08/02/2020).
- Motl, Jan (2019). *AdventureWorks*. URL: %5Curl%7Bhttps://docs.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver15%7D.
- NIST (1993). *Integration Definition for Information Modeling (IDEF1X)*. Federal Information Processing Standards Publication FIPS PUB 184. National Institute of Standards e Technology.

- Nonaka, Ikujiro e Hirotaka Takeuchi (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. New York: Oxford University Press, xii, 284 p.
- OMG (19 de mai. de 2015). *OMG Business Motivation Model. version 1.3. specification formal/2015-05-19*. Object Management Group.
- Oracle Corporation (2019a). *License for the Sakila Sample Database*. <https://dev.mysql.com/doc/sakila/en/sakila-license.html>. Accessed: 2019-10-23.
- (2 de nov. de 2019b). *MySQL 8.0 Reference Manual: Including MySQL NDB Cluster 8.0*. Acessado em 2019-11-02. Oracle Corporation. URL: %5Curl%7Bhttps://dev.mysql.com/doc/refman/8.0/en/%7D.
- (2019c). *Sakila Sample Database*. <https://dev.mysql.com/doc/sakila/en/>. Accessed: 2019-10-23.
- Panko, Raymond R. (1998). “What We Know About Spreadsheet Errors”. *Journal of End User Computing's* 10.2 (Spring 1998). Revised Mauy 2008, p. 15.21. URL: %5Curl%7Bhttp://panko.shidler.hawaii.edu/SSR/Mypapers/whatknow.htm%7D (acesso em 04/11/2019).
- Parrochia, Daniel (2020). *Classification*. URL: <https://www.iep.utm.edu/classifi/> (acesso em 21/01/2020).
- PMI (2017). *Um Guia do Conhecimento em Gerenciamento de Projetos (Guia PM-BOK®)*. Sexta Edição. Newtown Square, PA: Project Management Institute. ISBN: 9788502223745.
- Pressman, Roger e Bruce Maxim (2016). *Engenharia de Software-8ª Edição*. McGraw Hill Brasil.
- Pressman, Roger S. e Bruce Maxim (23 de jan. de 2014). *Software Engineering: A Practitioner's Approach*. 8 edition. New York, NY: McGraw-Hill Education. 976 pp. ISBN: 978-0-07-802212-8.
- Ralph Kimball, Margy Ross (12 de jul. de 2013). *The Data Warehouse Toolkit*. Wiley John + Sons. 608 pp. ISBN: 1118530802. URL: [https://www.ebook.de/de/product/20197316/ralph\\_kimball\\_margy\\_ross\\_the\\_data\\_warehouse\\_toolkit.html](https://www.ebook.de/de/product/20197316/ralph_kimball_margy_ross_the_data_warehouse_toolkit.html).
- Ricardo, David (1996). *Princípios de Economia Política e Tributação*. Os Economistas. texto original de 1821. São Paulo: Editora Nova Cultural Ltda.
- Robertson, James e Suzanne Robertson (1998). *Complete Systems Analysis*. New York: Dorser House.
- (2006). *Mastering the Requirements Process*. 2ª ed. Addison-Wesley Professional. ISBN: 0321419499.
- Rosen, Gideon (2018). “Abstract Objects”. Em: *The Stanford Encyclopedia of Philosophy*. Ed. por Edward N. Zalta. Winter 2018. Metaphysics Research Lab, Stanford University.
- Rowley, Jennifer (2006). “Where is the wisdom that we have lost in knowledge?” *Journal of Documentation* 62 (2), pp. 251–270.
- (2007). “The wisdom hierarchy: representations of the DIKW hierarchy.” *Journal of Information Science* 33.2, pp. 163–180.
- Ruble, David A. (1997). *Practical Analysis & Design for Client/Server and GUI Systems*. Upper Saddle River: Yourdon Press.

## Bibliografia

- Satpathy, Tridibesh et al. (2016). *A Guide to the SCRUM BODY OF KNOWLEDGE (SBOK™ Guide). A Comprehensive Guide to Deliver Projects using Scrum.* 2016<sup>a</sup> ed. SCRUMStudy, VMEdiu, Inc.
- Scott, Dale (2016). *MySQL version of Northwind demo database.* <https://github.com/dalers/mywind>. Accessed: 2019-10-23.
- Serrat, Olivier (2017). “The Five Whys Technique”. Em: *Knowledge Solutions: Tools, Methods, and Approaches to Drive Organizational Performance*. Singapore: Springer Singapore, pp. 307–310. ISBN: 978-981-10-0983-9. DOI: 10.1007/978-981-10-0983-9\_32. URL: [https://doi.org/10.1007/978-981-10-0983-9\\_32](https://doi.org/10.1007/978-981-10-0983-9_32) (acesso em 10/02/2020).
- Shlaer, Sally e Stephen J. Mellor (1999). *Object-Oriented Systems Analysis, Modelling the World in Data*.
- Sloan, Michael C. (2010). “Aristotle’s Nicomachean Ethics as the Original Locus for the Septem Circumstantiae”. *Classical Philology* 105.3, pp. 236–251. ISSN: 0009837X, 1546072X. URL: <http://www.jstor.org/stable/10.1086/656196>.
- Smith, Adam (1996). *A Riqueza das Nações. Investigação sobre sua natureza e suas causas*. Os Economistas. texto original de 1776. São Paulo: Editora Nova Cultural Ltda.
- (2003). *The Wealth of Nations*. texto original de 1854. Bantam Classics. ISBN: 0553585975.
- Soto, Christiane (2019). *12 of the Biggest Spreadsheet Fails in History*. Acessado em 2019-11-04. Oracle Small-to-Medium Business Blog. URL: %5Curl%7B12%20of%20the%20Biggest%20Spreadsheet%20Fails%20in%20History%7D (acesso em 04/11/2019).
- Sowa, J.F. e J.A. Zachman (1992). “Extending and Formalizing the Framework for Information Systems Arquitecture”. *IBM Systems Journal* 31.3, p. 590.
- Stapleton, Jennifer, ed. (2003). *DSDM: Business Focused Developmnt*. 2<sup>a</sup> ed. London: DSDM Consortium, Addison Wesley.
- Strathern, Paul (2003). *Uma Breve História da Economia*. Zahar.
- United States Code (jul. de 2002). *Sarbanes-Oxley Act of 2002, PL 107-204, 116 Stat 745*. Codified in Sections 11, 15, 18, 28, and 29 USC.
- Wells, Dave e Eav Nahari (2019). *Modernizing the Legacy Data Warehoustr: What, Why and How*. Acessado em 2019-11-3. Cloudera. URL: %5Curl%7Bhttps://www.slideshare.net/cloudera/modernizing-the-legacy-data-warehouse-what-why-and-how-12319%7D (acesso em 03/11/2019).
- Yourdon, Ed (2006). *Just Enough Structured Analysis*. Ed Yourdon.
- Zachman, J.A. (1987). “A Framework for Information Systems Architecture”. *IBM Systems Journal* 26.3, p. 276.
- Zins, Chaim (fev. de 2007). “Conceptual Approaches for Defining Data, Information, and Knowledge: Research Articles”. *J. Am. Soc. Inf. Sci. Technol.* 58.4, pp. 479–493. ISSN: 1532-2882.
- (2012). *Critical Delphi Research Methodology*. URL: <http://www.success.co.il/critical-delphi.html> (acesso em 31/12/2019).



## Exercício Um

O exercício tem como finalidade usar o software PENTaho Data Integration - Kettle Spoon - Hitachi Vantara - para criar um Data Warehouse a partir da base Sakila(Oracle Corporation, 2019c).

Para isso deve ser criada uma Tabela Fato Aluguel, baseada nas tabelas *Rental* e *Payment*, e as seguintes Tabelas Dimensão: Data, Cliente, Funcionario<sup>1</sup>, Loja, Linguagem e Filme. A Figura A.1 mostra o modelo dimensional a ser criado. A Tabela A.1 mostra a relação básica dessas tabelas do Data Warehouse com a base Sakila original.

Tabela A.1.: Relação entre as tabelas do DW e as tabelas da base Sakila.

Tabela do DW	Tabela Original
Aluguel	rental e payment
Data	rental e payment
Cliente	customer e address
Funcionario	staff e store
Loja	store e address
Filme	film
Linguagem	language e film

Há 3 ligações de Data com Aluguel: Data de Retirada, Data de Devolução, Data de Pagamento. Isso implica em 3 chaves estrangeiras na Tabela Fato para a mesma Tabela Dimensão. Não é necessário modelar o inventário, ou seja, que fita/CD foi retirado, só o filme. Aluguel e pagamento são de certa forma unificados, perceber que a ligação é 1:1 no modelo, mas que pode existir um sem o outro. Alguns campos devem ser calculados, o que é indicado na Tabela

---

<sup>1</sup>Sem acentos para não confundir os nossos bancos de dados.

### A. Exercício Um

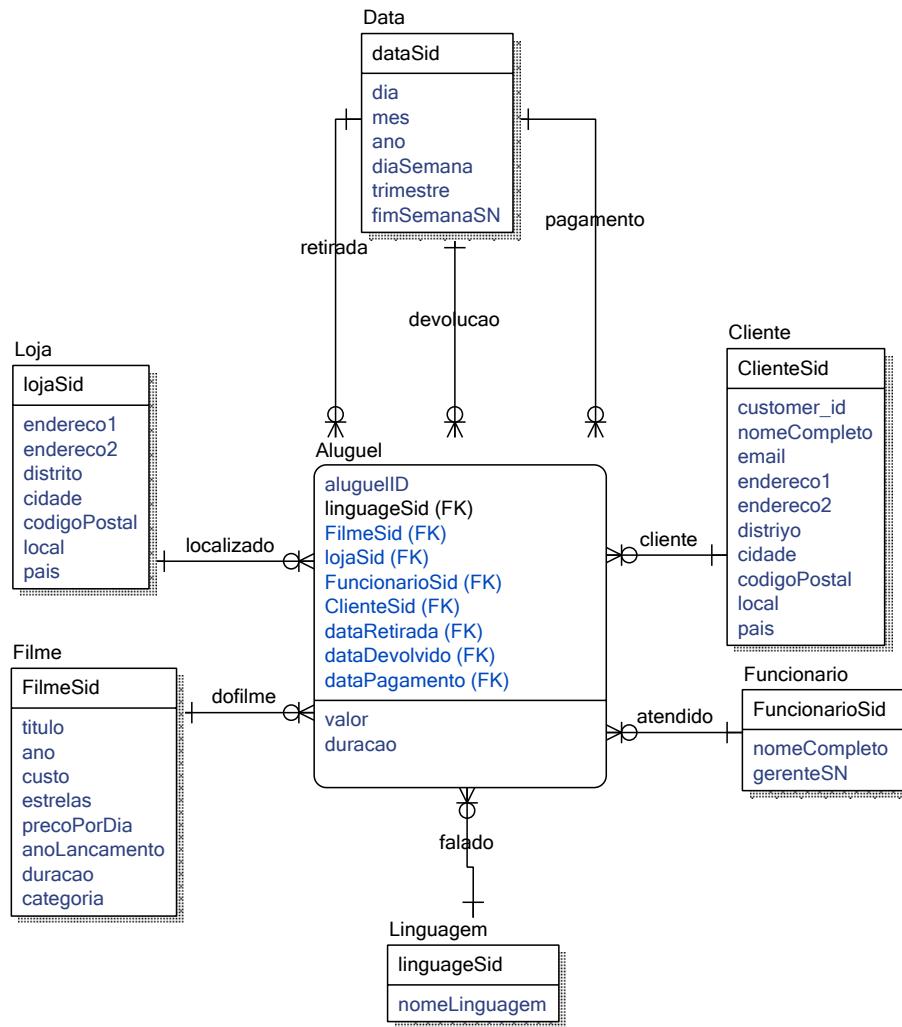


Figura A.1.: Modelo estrela para a Data Warehouse do exercício

Tabela	Campo	Cálculo
Aluguel	<b>duracao</b>	diferença entre as datas de entrega e retirada da fita
Funcionário	<b>gerenteSN</b>	pelo relacionamento entre <i>store</i> e <i>staff</i> .

Tabela A.2.: Campos a serem calculados

## A.1. Algumas questões

Por que, nesse modelo, ATOR não pode ser dimensão?

Nesse modelo, ator e aluguel tem um relacionamento NxM, o que não é facilmente compatível com o modelo estrela. Existem soluções que são complicadas para esse exercício, mas podem ser tentadas em outro contexto.

Nesse modelo é pedido que todos os relacionamentos FATO - DIMENSÃO sejam 1xN, como é o padrão.

Solução alternativa para ter essa informação no Data Warehouse: Criar um fato que fosse ALUGUEL/ATOR/FILME, possivelmente em outro modelo dimensional, ou criar

No caso do fato ALUGUEL/ATOR/FILME, não poderia ter o valor do aluguel, pois não se poderia somar (já que um filme tem vários atores, o valor do aluguel sairia multiplicado). Uma divisão pró-rata do valor do aluguel entre todas as instâncias desse tipo não seria AMIGÁVEL ao USUÁRIO, logo também não seria uma boa solução.

No caso de outro modelo dimensional com ALUGUEL/ATOR/FILME, não haveria valores financeiros ou de data, contando apenas o número de aluguéis (FACTLESS TABLE)

Aplicar a técnica de tabela ponto usada em saldo de conta de banco com cliente, onde uma conta pode ter vários cliente, mas que quebra o modelo estrela.

## A.2. Orientações

Voce precisa de 2 esquemas, o sakila e um outro, por exemplo, sakiladw no MySQL. As transformações devem levar de uma base para a outra.

As criações de tabelas devem ser feitas diretas no MYSQL, mas terão que ser apresentadas no trabalho final, guarde os s ou gere eles.

Todo o trabalho de transferência deve ser feito no PENTAHO.

Perguntas podem ser feitas na aula ou no Grupo Whats App

## A.3. Dicas de implementação

Como gerar corretamente chaves artificiais pode ser visto em [https://etl-tools.info/en/examples/surrogate-key\\_pdi.htm](https://etl-tools.info/en/examples/surrogate-key_pdi.htm)

Funções interessantes de SQL;

**DATEDIFF** calcula a diferença em dias entre duas datas.

