

# **Banco de Dados**

**Conceitos Básicos**

**2023**

# Introdução à Banco de Dados

## (Dado → Informação)

### Dado

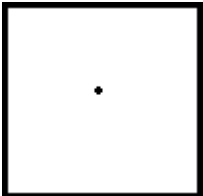
Mecanismo utilizado para descrever uma parte de algo maior que foi segmentado.

### Informação

É o resultado de um processo que é executado em um conjunto de dados, que estão organizados em uma estrutura de maneira lógica. Uma alteração na estrutura em que os dados são alocados pode gerar informações distintas após o processamento. Exemplo, a sequência de números “98765 – 4321”, pode representar um número de telefone ou, se houver uma pequena alteração na estrutura, “98765 – 4321 =?”, pode se transformar em uma expressão matemática. Note que os dados não foram alterados apenas a estrutura e com isso quando houve um processamento a informação mudou.

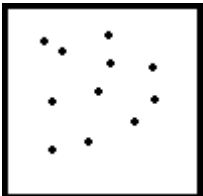
De acordo com nível do processamento, um dado pode representar uma informação ou uma informação pode representar apenas um dado de algo maior.

### Representação

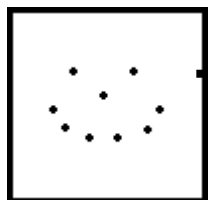


Um ponto em um quadrado branco não representa informação alguma.

Se for definido que esse ponto representa um dado, podemos concluir que, um conjunto de dados **não** representa uma informação.



Porém, se os dados forem organizados de maneira lógica, simplesmente, pode no levar a uma informação anormal.



O que você vê  
nesta figura?

Se você respondeu – um rosto – isso é estranho, pois, ninguém tem um rosto assim. Porém se for feita uma análise na figura e um processamento dos dados de acordo com a estrutura montada em seu cérebro através das suas lembranças e a pergunta for: - A qual sentimento lhe remete essa figura? Se você respondeu alegria ou felicidade, saiba que, o que foi feito, foi um processamento dos dados que estão organizados em uma estrutura de maneira lógica, que resultou em **informação**.

Ainda com esse raciocínio note que é possível guardar um sorriso, por meio de uma foto, por exemplo, porém alegria ou felicidade é a informação obtida a partir do processamento desses dados e são coisas que não dá para guardar, por isso temos banco de dados, ou seja, os dados são guardados e não as informações.

O **dado** quando processado gera a **informação** que, quando analisada resulta em **conhecimento** que por sua vez, permite a quem o detenha tomar **decisões**.

**Dado → Informação → Conhecimento → Decisão.**

## A história dos bancos de dados

Antigamente as empresas armazenavam dados em fichas de papel que eram organizadas em arquivos físicos através de pastas. No início, cada entidade (clientes, funcionários, produtos, etc.) era um arquivo de dados que eram acompanhados de um “software simples” para manipular os dados do arquivo, esses softwares permitiam realizar operações de cadastro, alteração, exclusão e consulta nos arquivos digitais. Mas as entidades precisavam relacionar-se, por exemplo um produto é fornecido por um fornecedor.

Na década de 60 a empresa IBM investiu fortemente em pesquisas para solucionar estes problemas dos bancos de dados digitais primitivos. Em junho de 1970, o pesquisador Edgar Frank “Ted” Codd da IBM, apresentou o modelo relacional no artigo intitulado “*A Relational Model of Data for Large Shared Data Banks*”, onde o autor apresentou uma forma de usuários sem conhecimento técnico armazenarem e extraírem grandes quantidades de informações de um banco de dados. Ele desconectou a estrutura lógica do banco de dados do método de armazenamento físico. Este sistema se tornou padrão desde então e o seu artigo foi o grande impulso para a evolução dos bancos de dados. Logo após, o Dr. Peter Chen propôs o modelo Entidade-Relacionamento (ER) para projetos de banco de dados dando uma nova e importante percepção dos conceitos de modelos de dados. Assim como as linguagens de alto nível, a modelagem ER possibilita ao projetista concentrar-se apenas na utilização dos dados, sem se preocupar com estrutura lógica de tabelas.

No final da década de 70 que foi criada a linguagem de consulta estruturada (*SQL - Structured Query Language*) que se tornou a linguagem padrão para bancos de dados relacionais. Nos anos 80 surgiram outros bancos de dados, a Oracle apresentou o Oracle 2 e a IBM o SQL/DS (que se tornou DB2), ambos sistemas comerciais de bancos de dados. Na sequência vieram SQL Server, MySQL, etc...

## Tipos de Bancos de Dados

Os tipos de banco de dados são divididos em duas categorias: os **Modelos Relacionais** e os **Não-Relacionais**.

Os bancos de dados relacionais são fundamentados no paradigma da orientação a conjuntos, uma vez que sua base é construída em cima da teoria dos conjuntos. Esses bancos armazenam dados em estruturas chamadas tabelas, compostas por colunas e linhas. Esses são bancos do tipo SQL. Exemplos, *Oracle*, *SQL Server*, *MySQL* e *PostgreSQL*.

Os bancos de dados não-relacionais surgem como solução para situações nas quais os bancos relacionais não atendem de forma satisfatória. Ambientes com dados mistos — como imagens, mapas e tabelas — que não podem ser facilmente organizados (armazenados) em linhas e colunas necessitam de uma solução não-relacional. Esses são bancos do tipo *NoSQL*. Exemplos, *MongoDB*, *Redis* e *Cassandra*.

## SGBD

Sistema Gerenciador de Banco de Dados ou DBMS (Data Base Management System), trata-se de um sistema para gerenciar uma base de dados ou banco de dados. Um SGBD representa conjuntos de softwares utilizados para o gerenciamento de uma base de dados, ou seja, são os programas utilizados para controlar, organizar, acessar e proteger as informações de uma empresa. Os dados presentes em uma base de dados devem obedecer a diversas regras conhecidas como restrições de integridade. Entre os principais SGBD's existentes no mercado, basicamente todos possuem funcionalidades em comum:

- Inserir, excluir, acessar, visualizar, selecionar, ordenar, juntar ou intercalar registros;
- Alterar estruturas de campos;
- Inserir, remover e estabelecer relações entre tabelas;
- Importar ou exportar dados entre outras bases de dados;
- Realizar consultas, elaborar formulários e relatórios na base de dados;
- Criar usuários, com permissões de acesso diferenciadas.

Os principais SGBD's (<https://db-engines.com/en/ranking>):

- **Oracle** – Oracle
- **MySQL** – Oracle
- **SQL Server** – Microsoft
- **PostgreSQL** – PostgreSQL Global Development Group
- **MongoDB** – MongoDB, Inc
- **DB2** – IBM
- **Cassandra** – Facebook depois fundação Apache
- **Access** – Microsoft
- **SQLite** – Dwayne Richard Hipp

## Conceitos bancos de dados

Um banco de dados, basicamente é um local onde, de uma maneira estruturada, pode ser guardados dados para que em um futuro sejam juntados e organizados afim de se obter informação. Essas estruturas (tabelas) são criadas, organizadas e mantidas através de ferramentas de banco de dados.

Exemplo:

The diagram shows a table with the following structure:

Aluno			
RM	Nome	Telefone	Email
912345	João da Silva	11 98765-3212	john@email.com
915432	Ana Maria de Sá	11 97856-1234	ans@email.com

Annotations:

- 1: Points to the table name **Aluno**.
- 2: Points to the attribute headers **RM**, **Nome**, **Telefone**, and **Email**.
- 3: Points to the data rows (records) of the table.

No exemplo é mostrada uma estrutura, que é conhecida como tabela, que é utilizada para armazenar dados. Onde:

- 1 – **Aluno** → Nome da tabela.
- 2 – **RM, Nome, Telefone e Email** → São os atributos (Colunas).
- 3 – **Dados dos alunos** → São os registros.

## Modelagem de Dados

### Conceitos

Modelar é o processo de criar um modelo que demonstre as características e comportamentos para o funcionamento de um software, com isso será cria uma maneira de fácil entendimento do projeto, além da possibilidade de verificação de futuros erros que poderão vir a ocorrer durante o processo de desenvolvimento. Os modelos de dados permitem demonstrar como serão construídas as estruturas de dados que servirão de base para os processos do negócio, como esses dados serão organizados e quais os relacionamentos que serão necessários no sistema.

### Modelo de Entidade e Relacionamento

O modelo entidade relacionamento (MER) é um modelo de dados para descrever os dados ou aspectos de informação de um negócio ou seus requisitos de processo. Por meio do MER, é possível ter uma maneira simples de desenhar a estrutura de banco de dados, destacando os seus componentes que são suas **Entidades** (Tabelas) e os seus respectivos **Relacionamentos**.

As Entidades são representações de algo no mundo físico (Produto, Aluno, Veículo e etc.) ou representam algo que seja essencial para o negócio (Venda, Locação, Gênero e etc.). Os relacionamentos indicam que há uma ligação entre duas entidades, ou algo que faça com que essas entidades tenham algo em comum.

Nesse modelo uma Entidade é representada por retângulo e um relacionamento é representado por um losango.



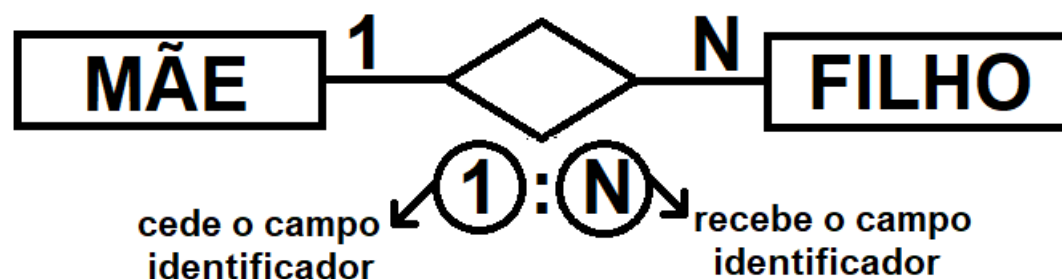
Com esse modelo é possível ter uma ideia bastante simples sobre o funcionamento do sistema, além de ser uma ferramenta que pode ser utilizada para que o desenvolvedor possa demonstrar algumas funcionalidades, de maneira com que outras pessoas envolvidas no processo consigam entender.

O relacionamento e as dependências de cada entidade são demonstrados através das suas cardinalidades.

### Relacionamento *um-para-muitos* (1:N)

Na cardinalidade 1:N, a entidade que possui o lado “1” é definida como a entidade que cederá o seu campo identificador (campo chave) para entidade que tem o lado “N”, ou seja, essa entidade deve ser criada primeiro no sistema e a entidade com lado “N” dependerá diretamente da entidade com lado “1”.

Exemplo:



No caso do relacionamento 1:N, a entidade “**MÃE**” pode ter vários filhos (N) e a “**FILHO**” pode ter apenas uma mãe (1), assim a entidade **FILHO** é dependente direta da entidade **MÃE**. O campo identificado (campo chave) da entidade **MÃE** deverá ser atribuído na entidade **FILHO** para eles possam se relacionar.

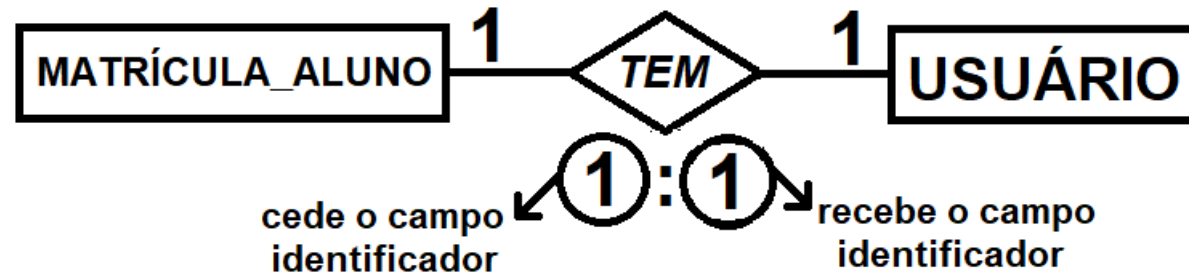
MÃE	
idMae	NomeMae
1010	Johzeffa Maryah
2020	Giudézia Kelly

FILHO		
idFilho	NomeFilho	idMae
10	Dion Nyzio	1010
20	Amavi da Doustros	2020
30	Zebedeu de Sá	1010

## Relacionamento *um-para-um* (1:1)

Na cardinalidade 1:1, ambas as entidades possuem o lado “1”, nesse caso uma análise deve ser feita do sistema afim de identificar qual é a entidade mais importante (lado obrigatório) para o negócio, assim identificador (campo chave) dessa entidade deve ser cedido à outra entidade (lado opcional), ou seja, essa entidade deve ser criada primeiro no sistema e a outra entidade com lado “1” dependerá diretamente da primeira entidade.

Exemplo:



No caso do relacionamento 1:1, a entidade “**MATRÍCULA\_ALUNO**” pode ter apenas uma conta de usuário associada a ela, e a entidade “**USUÁRIO**” pode ter apenas uma matrícula associada a cada registro dela. No negócio de uma escola, por exemplo, a matrícula é mais importante do que uma conta de usuário para um aluno, então a entidade **MATRÍCULA\_ALUNO** deve ser criada primeiro e a entidade **USUÁRIO** dependerá dessa entidade.

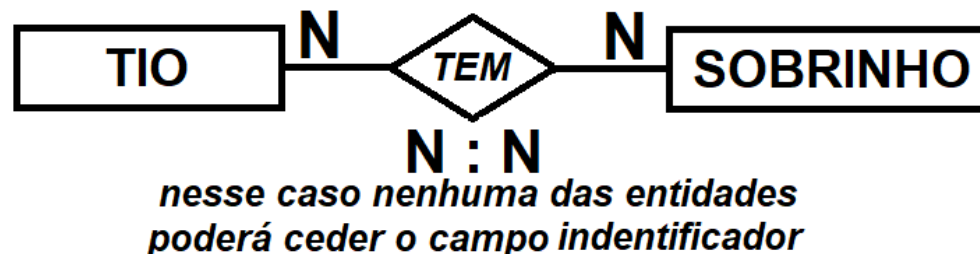
MATRÍCULA_ALUNO	
<b>rm</b>	NomeAluno
96101	Dion Nyzio
96202	Amavi da Doustros
96303	Zebedeu de Sá

USUARIO		
<b>idUsuario</b>	NomeUsuario	<b>rm</b>
6101	Dion_96101	96101
6202	Amavi_96202	96202
6303	Zebedeu_96303	96303

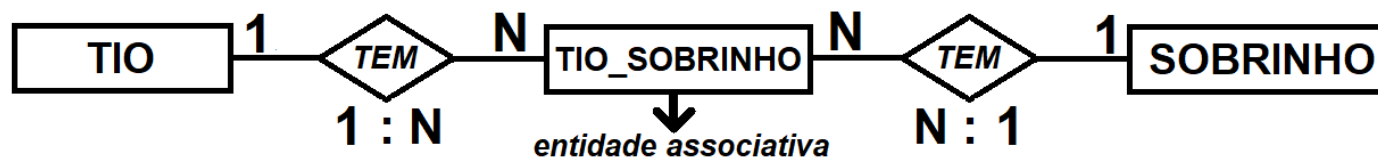
## Relacionamento *N-para-N* ou *Muitos-para-Muitos* (N:N)

Na cardinalidade N:N, ambas as entidades possuem o lado “N”, nesse caso, essa situação pode representar um erro, se isso for para próxima etapa do desenvolvimento do banco de dados. Nesse momento deve ser feito um desmembramento do relacionamento, e uma nova entidade entre as duas que geraram a cardinalidade N:N deve ser criada, essa nova entidade é chamada de **Entidade Associativa** e ela receberá os campos identificadores das outras entidades.

Exemplo:



Solução:



No caso do relacionamento N:N, o “**TIO**” pode vários sobrinhos e o “**SOBRINHO**” pode ter vários tios, então como nenhum dos lados pode ceder o seu campo identificador, a solução é criar uma entidade associativa e colocar os campos identificadores nessa nova entidade.

TIO	
idTio	NomeTio
10	Donald
20	Patinhas

TIO_SOBRINHO	
idTio	idSobrinho
10	23
10	33
10	43
20	23
20	33
20	43

SOBRINHO	
idSobrinho	NomeSobrinho
23	Huguinho
33	Zezinho
43	Luisinho

## MER – Modelo de Entidade e Relacionamento



Depois a análise do negócio de uma pequena empresa que presta serviços de banho e tosa para cães e gatos, foram levantadas as necessidades para o desenvolvimento de um sistema para otimizar as rotinas de cadastros, de venda e agendamento da empresa.

Algumas regras de negócio foram definidas pelos proprietários da empresa:

- O atendimento aos **animais** é feito mediante a um **agendamento** prévio ou diretamente, se houver disponibilidade.
- A empresa precisa de um controle sobre as raças dos animais, assim, a **raça** precisa corresponder à sua **espécie**.
- O **cliente** precisa ter pelo menos **dois telefones para contato** e apenas um cliente é responsável pelo animal.
- Cada **funcionário** acessará o sistema usando sua conta interna de **usuário**.
- Uma **venda** poderá conter mais de um **serviço** e poderá ser **paga** em mais de uma maneira.
- A empresa vende serviços, como banho, tosa, higienização bucal entre outros. A empresa não comercializa produtos físicos.

Depois de feita a análise e listadas as necessidades/regras do negócio, temos as seguintes entidades:

- Cliente
- Telefone de Contato
- Funcionário
- Usuário
- Animal
- Raça
- Espécie
- Venda
- Forma de Pagamento
- Serviço
- Agenda

**Treine um pouco:**

Determine as cardinalidades dos relacionamentos a seguir:



## Normalização de Dados – Conceitos

Normalização de dados é o processo passo que examina os atributos de uma entidade, com o objetivo de evitar anomalias observadas na manipulação dos registros. Com a aplicação das regras de normalização de dados, algumas tabelas acabam sendo divididas em duas ou mais tabelas. Este processo causa a simplificação dos atributos de uma tabela, colaborando com a estabilidade do modelo de dados, reduzindo a necessidade de manutenções.

O Processo de normalização aplica uma série de regras às tabelas de um banco de dados, o que permite verificar se elas estão corretamente projetadas. Embora existam cinco formas normais (ou regras de normalização), um banco de dados pode ser considerado estável quando ele se encontra na terceira forma normal.

1. **Primeira Forma Normal** (ou **1FN**): Nesta forma os atributos precisam ser atômicos, o que significa que as colunas não podem ter mais de um valor e nem ter várias colunas com o mesmo tipo de valor.

Ex.: 1-)

Cardápio	
Prato (PK)	Ingredientes
Feijoada	Feijão Preto, Linguiça, Bacon, Couve
Macarronada	Macarrão, Molho

Ex.: 2-)

Agenda			
Contato (PK)	Telefone1	Telefone2	Telefone3
João	98765-4321	78965-4512	
Maria	4123-4567	98754-2145	9875-5421

### Solução:

Identificar a coluna que possui dados repetidos e removê-los; construir outra tabela com os atributos removidos e fazer uma relação entre as duas tabelas.

2. **Segunda Forma Normal** (ou **2FN**): Primeiramente, para estar na 2FN é preciso estar também na 1FN. A Segunda Forma Normal define que os atributos **não chave**, devem depender unicamente da chave primária da tabela, ou quando uma coluna representa um conjunto de valores. Assim como as colunas da tabela que não são dependentes dessa chave devem ser removidas da tabela principal e uma nova tabela deve ser criada utilizando esses dados.

Ex.: 1-)

Locação			
ID_Locação (PK)	ID_Filme	Titulo	ID_Cliente
1010	54210	Rambo	10
1123	54321	Nemo	20

Ex.: 2-)

Filme			
ID_Filme (PK)	Titulo	Genero	Classificacao
1010	Rambo	Ação	18
1123	Nemo	Animação	Livre

### Solução:

Identificar os dados não dependentes da chave primária e removê-los; construir uma nova tabela com os dados em questão e manter o relacionamento entre os dados.

- 3. Terceira Forma Normal (ou 3FN).** Para estar na 3FN é preciso estar também na 2FN. A Terceira Forma Normal define que todos os atributos dessa tabela devem ser funcionalmente independentes uns dos outros, ao mesmo tempo em que devem ser dependentes exclusivamente da chave primária da tabela. Por exemplo, campos que armazenam resultados de cálculos matemáticos na própria tabela.

Ex.:

Item_Venda			
ID_ITEM (PK)	QTDE	Preço_Unit	Sub_Total
1234	10	2,50	25,00
3241	5	1,99	9,95

### Solução:

Identificar os dados dependentes de outros e removê-los da tabela. Esses atributos poderiam ser simplesmente excluídos, fazendo com que o resultado do cálculo seja de responsabilidade da aplicação, ou até ser movidos para uma nova tabela e referenciar com a principal.

## Normalização de Dados – Exemplo

Normalização de dados é o processo passo que examina os atributos de uma entidade, com o objetivo de evitar anomalias observadas na manipulação dos registros. Com a aplicação da normalização de dados, algumas tabelas acabam sendo divididas em duas ou mais tabelas. Este processo causa a simplificação dos atributos de uma tabela, colaborando com a estabilidade do modelo de dados, reduzindo a necessidade de manutenções.

### Exemplo:

Cardápio
<ul style="list-style-type: none"><li>▪ idCardapio ( PK ) – campo identificador da tabela [<b>Chave Primária</b>]</li><li>▪ nomePrato</li><li>▪ ingredientes</li><li>▪ acompanhamento1</li><li>▪ acompanhamento2</li><li>▪ tipoPrato</li><li>▪ preco</li></ul>

Tabela →	Cardápio						
Campos →	idCardapio (PK)	NomePrato	Ingredientes	Acompanhamento1	Acompanhamento2	TipoPrato	Preço
Registros →	10	Feijoada	Feijão, Carne de Porco e Temperos	Arroz, Pururuca, Couve e Laranja	Bisteca suína	Principal	25,00
	12	Lasanha	Macarrão, Queijo, Presunto e Molho	Arroz e Farofa	Batata Frita	Principal	22,00
	25	Salada Verão	Alface, Tomate e Cenoura	Molho	Azeite e limão	Entrada	12,00
	38	Strogonoff de Frango	Carne de Frango e Molho branco com um pouco de Ketchup	Batata palha e Arroz	Salada	Principal	20,00

### Regras da normalização:

- O campo **não** pode ter mais do que um valor por célula. [como ocorre no campo: **Ingredientes**]
- A tabela **não** pode ter dois campos que armazenam o mesmo tipo de dado. [como ocorre nos campos: **Acompanhamento1** e **Acompanhamento2**]
- O valor da célula deve **depender** unicamente **do campo identificador** (a chave primária) da tabela. [o que **não ocorre** no campo: **TipoPrato**]

### Solução:

- Criar outra tabela com os campos anormais e manter o relacionamento entre as duas.
- De acordo com as regras do negócio criar soluções específicas para cada situação.

Tabela →	Cardápio			
Campos →	idCardapio (PK)	NomePrato	Preço	idTipoPrato (FK)
Registros →	10	Feijoadada	25,00	2
	12	Lasanha	22,00	2
	25	Salada Verão	12,00	1
	38	Strogonoff de Frango	20,00	2

Tabela →	TipoPrato	
Campos →	idTipoPrato (PK)	NomeTipoPrato
Registros →	1	Entrada
	2	Principal
	3	Sobremesa

Tabela →	Ingrediente_Cardapio	
Campos →	idCardapio (FK)	idIngrediente (FK)
Registros →	10	23
	10	75
	10	40


Tabela →	Ingrediente	
Campos →	idIngrediente (PK)	NomeIngrediente
Registros →	23	Feijão
	28	Arroz
	45	Macarrão
	75	Carne de Porco
	40	Temperos

Tabela →	Acompanhamento_Cardapio	
Campos →	idCardapio (FK)	idAcomp (FK)
Registros →	10	11
	10	15
	10	40
	10	38
	10	47
	10	51

Tabela →	Acompanhamento	
Campos →	idAcomp (PK)	NomeAcomp
Registros →	11	Arroz
	15	Pururuca
	20	Batata Frita
	35	Salada
	38	Couve
	47	Laranja
	51	Bisteca Suína

## Criação de Tabelas

O processo de criação de tabelas (a codificação da tabela) está na terceira etapa da criação de um banco de dados, que é o Modelo Físico. Nessa etapa será codificado tudo o que foi levantado e modelado a respeito do banco de dados.

Etapas da criação do Banco de Dados																							
<b>Modelo Conceitual</b>	Levantamento de requisitos e modelagem do Banco de Dados. Criação do MER.	<ul style="list-style-type: none"> <li>• Espécie</li> <li>• Animal</li> </ul> 																					
<b>Modelo Lógico</b>	Descreve como os dados serão armazenados no banco de dados e com e quais tabelas se relacionarão.	<table border="1"> <thead> <tr> <th colspan="3">Espécie</th></tr> </thead> <tbody> <tr> <td>idEspecie</td><td>Interio</td><td>Chave Primária</td></tr> <tr> <td>nomeEspecie</td><td>Texto</td><td>Não Nulo</td></tr> </tbody> </table> <table border="1"> <thead> <tr> <th colspan="3">Animal</th></tr> </thead> <tbody> <tr> <td>idAnimal</td><td>Interio</td><td>Chave Primária</td></tr> <tr> <td>nomeAnimal</td><td>Texto</td><td>Não Nulo</td></tr> <tr> <td>idEspecie</td><td>Interio</td><td>Chave Estrangeira</td></tr> </tbody> </table>	Espécie			idEspecie	Interio	Chave Primária	nomeEspecie	Texto	Não Nulo	Animal			idAnimal	Interio	Chave Primária	nomeAnimal	Texto	Não Nulo	idEspecie	Interio	Chave Estrangeira
Espécie																							
idEspecie	Interio	Chave Primária																					
nomeEspecie	Texto	Não Nulo																					
Animal																							
idAnimal	Interio	Chave Primária																					
nomeAnimal	Texto	Não Nulo																					
idEspecie	Interio	Chave Estrangeira																					
<b>Modelo Físico</b>	Criação/codificação do Banco de Dados. Uso de um SGBD.	<pre>CREATE TABLE Espécie (     idEspecie INT PRIMARY KEY,     nomeEspecie VARCHAR (50) NOT NULL )</pre>																					

Quando uma Entidade é transposta para o modelo físico ela passa a ser tratada como tabela. Uma tabela é uma estrutura composta por linhas e colunas, onde cada coluna recebe um cabeçalho (nome do campo) e o cruzamento de uma linha com uma coluna forma uma célula. Assim, no momento da criação de uma tabela é rio definir qual é o tipo de dado que cada campo armazenará. O tipo de dado define se o conteúdo do campo será armazenado na forma de texto, número, datas e etc.

<b>Tipo de valor do campo</b>	<b>Tipo de Dados (Na descrição da tabela)</b>	<b>Tipos de Dados (No MS SQL Server)</b>
Cadeias de caracteres (Textos)	<b>Texto</b>	<ul style="list-style-type: none"> <li>• CHAR ( 1 – 8000 )</li> <li>• VARCHAR ( 1 – 8000 )</li> <li>• TEXT</li> </ul>
Numéricos exatos (Inteiros)	<b>Número Inteiro</b>	<ul style="list-style-type: none"> <li>• BIT</li> <li>• TINYINT</li> <li>• SMALLINT</li> <li>• INT</li> <li>• BIGINT</li> <li>• NUMERIC</li> </ul>
Numéricos exatos (Decimais)	<b>Número Decimal Moeda</b>	<ul style="list-style-type: none"> <li>• DECIMAL ( P, S ) P (Precisão), S (Escala)</li> <li>• SMALLMONEY</li> <li>• MONEY</li> </ul>
Data e hora	<b>Data e Hora Data Hora</b>	<ul style="list-style-type: none"> <li>• SMALLDATETIME</li> <li>• DATETIME</li> <li>• DATE</li> <li>• TIME</li> </ul>
Numéricos aproximados (Pontos flutuantes)	<b>Ponto Flutuante</b>	<ul style="list-style-type: none"> <li>• REAL</li> <li>• FLOAT</li> </ul>

Para descrever uma tabela é preciso informar como os seus campos serão criados, definindo as propriedades como nome, tipo de dados e tipo de restrição.

- **Nome**: o nome do campo deve ser auto explicativo, ou seja, pelo nome do campo já deve ser possível ter uma ideia do que ele armazena. Algumas regras e recomendações:
  - Não deve começar com números e nem caracteres especiais;
  - Deve ser evitado o uso de acentos e traços;
  - Se houver mais de um nome no campo, não pode ter espaços;
  - Não usar palavras reversadas do SGBD.
- **Tipo de Dado**: o tipo de dado descreve como (o formato) os valores serão armazenados no campo:
  - Texto
  - Número Inteiro
  - Número Decimal ou Moeda
  - Data e Hora, Data ou Hora
  - Ponto Flutuantes (valores com casas decimais variadas)



- **Tipo de restrição:** o tipo de restrição do campo define quais as regras que devem ser respeitadas para que o valor possa ser armazenado no campo:
  - **Chave Primária** – determina que o campo será o campo identificador da tabela, esse será o campo de referência para todos os outros campos da tabela.
  - **Chave Estrangeira** – indica que o campo será usado para manter um relacionamento com outra tabela no Banco de Dados. Esse campo deve ter sido definido como Chave Primária na tabela que mantém o relacionamento.
  - **Único** – indica que o campo terá o seu valor único na tabela, ou seja, o valor do campo não poderá se repetir dentro da mesma tabela e no mesmo campo.
  - **Padrão** – define um valor padrão quando esse valor for omitido na inserção de valores no campo.
  - **Checar** (ou verificar) – verifica se os valores inseridos estão de acordo com a restrição definida no campo.
  - **Não Nulo** – indica que campo tem que possuir algum valor, não podendo ficar com valores nulos.
  - **Nulo** – indica que o campo não precisa, obrigatoriamente ter algum valor inserido, podendo receber valores nulos.

Também é possível definir que um campo do tipo numérico inteiro terá a propriedade de “**auto numérico**” definida pra ele. Isso é muito útil para os campos que são Chave Primária, e só é possível definir essa propriedade para apenas um campo por tabela.

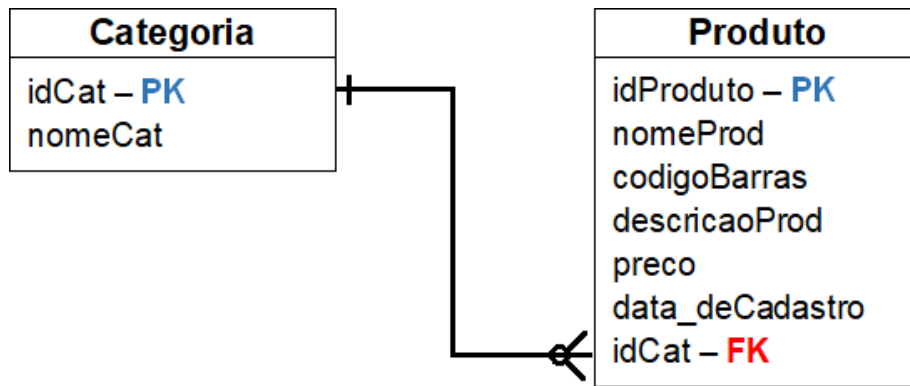
### **Exemplo de descrição das tabelas:**

Tabela: Espécie		
Campo	Tipo de Dado / Tamanho	Restrição
idEspécie	Número Inteiro – Auto Numérico	Chave Primária
nomeEspécie	Texto / 50	Não Nulo

Tabela: Animal		
Campo	Tipo de Dado / Tamanho	Restrição
idAnimal	Número Inteiro – Auto Numérico	Chave Primária
nomeAnimal	Texto / 50	Não Nulo
sexo	Texto / 50	Não Nulo
data_deNascimento	Data	Nulo
idEspécie	Número Inteiro	Chave Estrangeira

**Treine um pouco:**

De acordo com as tabelas mostradas a seguir crie a descrição de cada uma delas:



## Dicionário de Dados

Um dicionário de dados é uma coleção de metadados (informações sobre as estruturas dos dados) que contém definições e representações de elementos de dados. Em um SGBD, um dicionário de dados é um grupo de tabelas apenas para leitura ou consulta, que servem como base ou documentação para codificação das tabelas no banco de dados.

No Dicionário de Termos de Dados, a tabela “**Funcionário**” será representada dessa maneira:

Tabela: Funcionário			
Campo	Tipo de Dado (tamanho)	Restrição	Observação / Descrição
id	INT / <b>IDENTITY</b>	PRIMARY KEY	
nome	VARCHAR (100)	NOT NULL	
sexo	CHAR (1)	NOT NULL	
dataNasc	DATE	NOT NULL	
rg	VARCHAR (12)	NOT NULL	
cpf	CHAR (11)	NOT NULL	
dataCad	SMALLDATETIME	DEFAULT / NOT NULL	<b>GETDATE()</b>
cep	CHAR (8)	NOT NULL	
numResid	VARCHAR (10)	NOT NULL	
telefone	VARCHAR (20)	NOT NULL	
email	VARCHAR (100)	NULL	
cargo_id	INT	FOREIGN KEY	Tabela: Cargo
status	VARCHAR (10)	NOT NULL	ATIVO ou INATIVO