



PUC
RIO

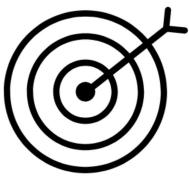
AULA 1

Bancos de dados: fundamentos, requisitos e modelos de dados

A PROFESSORA

Veronica dos Santos

Formada em informática pela UERJ (1998). Tem MBA em engenharia de software pela Escola Politécnica da UFRJ (2007). Iniciou sua carreira de TI em 1995, atuando como analista de sistemas e de requisitos no Citibank, Light-Rio e Embratel. Ingressou no IBGE em 2005, atuando como administradora de banco de dados até 2011. Concluiu o mestrado em informática pela Unirio, na área de concentração de banco de dados. Em 2011, assumiu a gerência de banco de dados, até 2017. Em 2023, concluiu o doutorado em informática na PUC-Rio, com pesquisa sobre grafos de conhecimento e busca exploratória. Desde 2022, é professora assistente da disciplina de banco de dados na pós-graduação *lato sensu* da PUC-Rio.



Objetivos de aprendizag em da aula

Ao final desta aula, você irá:

- Adquirir conhecimento geral sobre sistemas gerenciadores de banco de dados, seu uso e importância.
- Identificar requisitos de dados de uma aplicação.
- Identificar entidades, atributos, relacionamentos e cardinalidades para gerar um modelo conceitual.

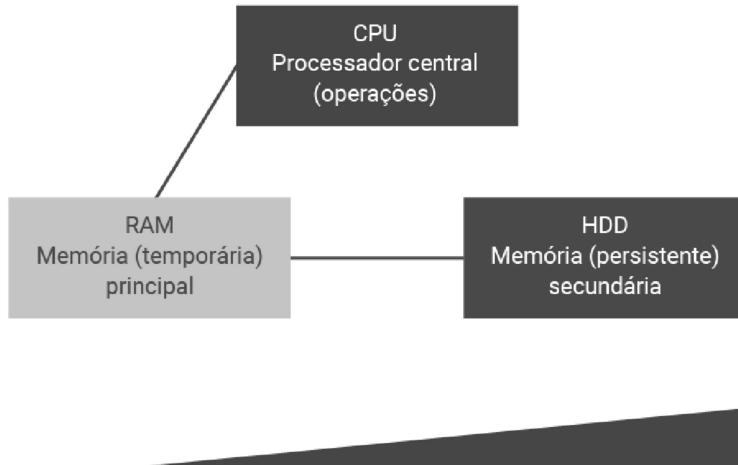
Que história é essa?

Esta disciplina tem por objetivo ensinar os conceitos básicos de bancos de dados, com ênfase na teoria e prática. Nesta aula, serão vistos conceitos introdutórios para o desenvolvimento de sistemas de bancos de dados, incluindo modelos de dados e linguagem SQL, para projeto e concepção de novos bancos de dados que respondam a requisitos impostos por aplicações.

Entendendo o contexto

As arquiteturas dos computadores ditos convencionais – *desktops*, *laptops* e mesmo *smartphones* – apresentam três componentes principais: a **CPU** (*central processing unit*), responsável pelas operações; a memória principal, ou primária, também conhecida pelo acrônimo **RAM** (*random access memory*); e o **HDD** (*hard disk drive*), que representa o componente de armazenamento secundário. Os termos primário e secundário são relevantes para quem usa bancos de dados.

Componentes básicos de um computador



A CPU só acessa diretamente dados na RAM, que é bem menor, em bytes, do que um HDD. Como nem sempre todos os dados de que precisamos cabem na RAM, o sistema operacional deve buscá-los no HDD e substituir os dados presentes na RAM pelos dados recém obtidos do HDD. No caso de grandes quantidades de dados, esses procedimentos precisam ser feitos muitas vezes, e o custo, em termos de tempo de processamento, é alto, com consequências diretas no desempenho do sistema como um todo.

Os **sistemas gerenciadores de banco de dados (SGBDs)** surgiram pela necessidade de lidar eficientemente com volumes crescentes de dados, garantir a qualidade e consistência dos dados, proporcionar acesso concorrente e seguro e simplificar o desenvolvimento e a manutenção de aplicações que dependem de dados armazenados.



Aprenda mais

O **Integrated Data Store (IDS)**, desenvolvido no final da década de 1960 por Charles W. Bachman, é considerado um dos primeiros SGBDs. Tratava-se de um SGBD que utilizava o modelo hierárquico, ou seja, **os dados eram organizados em uma estrutura de árvore**. Cada registro de dados poderia ter registros-filhos associados, formando uma hierarquia. Bachman recebeu o Prêmio Turing em 1973 por seu trabalho pioneiro em bancos de dados.

Visão geral de bancos de dados

É verdade que muitas pessoas se referem aos bancos de dados como um “bando” de dados. É o caso de aplicações contábeis que usam muitas planilhas de cálculo ou, ainda, em referência à gestão de coleções pessoais de livros ou fotografias.

Em termos computacionais, um **banco de dados** é uma coleção de dados armazenados, organizados e inter-relacionados que atende às necessidades de vários usuários, dentro de uma ou mais organizações, por meio de uma ou várias aplicações. Esses dados representam aspectos do mundo real (minimundo ou universo de discurso) de modo logicamente coerente e com algum significado inerente. Cada banco de dados é projetado, construído e instanciado (“povoado”) para objetivos específicos. Uma **instância de banco de dados** corresponde ao conjunto de dados armazenados em um banco de dados em um determinado instante de tempo, cuja consistência é garantida pelo SGBD.

A área de banco de dados se diferencia das demais por, pelo menos, dois aspectos:

Aspectos que diferenciam a área de banco de dados



O fato de sempre lidar com **grandes volumes de dados**, com desenvolvimento de algoritmos e programas de gestão, acesso e manuseio específicos para lidar com vários níveis hierárquicos de memória (p. ex.: secundária).



A necessidade de **persistência** dessas grandes quantidades de dados, que exigem técnicas específicas de gestão e armazenamento, garantindo integridade e consistência dos dados.

Para entender os aspectos do projeto de um banco de dados, deve-se começar pelo conceito de modelo de dados. Neste curso, é dada ênfase ao **modelo de dados relacional**, padrão de fato e de mercado para a maioria dos sistemas de informação existentes. Entretanto, muitas das definições e termos aqui citados se aplicam aos modelos de dados em geral, inclusive os não relacionais (NoSQL).



Fique ligado

O site [DB-Engines Ranking](#) classifica os SGBDs de acordo com sua popularidade. A popularidade de um sistema é calculada usando alguns parâmetros, como número de menções do sistema em sites (medido como número de resultados de pesquisa em mecanismos de busca) e interesse geral no sistema (frequência de buscas no Google Trends).

O ranking é atualizado mensalmente e, em dezembro de 2023, havia sete SGBDs relacionais entre os Top 10. Além disso, o SQLite, que será usado nos exercícios das aulas 2 e 3, está na 11^a posição. O site permite que os usuários filtrem por categorias e também comparem os SGBDs em relação a suas características.

Top 10 SGBDs, de acordo com a DBEngines (dez/23)

417 systems in ranking, December 2023

Rank	Dec 2023	Nov 2023	Dec 2022	DBMS	Database Model	Score		
						Dec 2023	Nov 2023	Dec 2022
1.	1.	1.	1.	Oracle	Relational, Multi-model	1257.41	-19.62	+7.10
2.	2.	2.	2.	MySQL	Relational, Multi-model	1126.64	+11.40	-72.76
3.	3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	903.83	-7.59	-20.52
4.	4.	4.	4.	PostgreSQL	Relational, Multi-model	650.90	+14.05	+32.93
5.	5.	5.	5.	MongoDB	Document, Multi-model	419.15	-9.40	-50.18
6.	6.	6.	6.	Redis	Key-value, Multi-model	158.35	-1.66	-24.22
7.	7.	↑ 8.	Elasticsearch		Search engine, Multi-model	137.75	-1.87	-7.18
8.	8.	↓ 7.	IBM Db2		Relational, Multi-model	134.60	-1.40	-12.02
9.	↑ 10.	9.	Microsoft Access		Relational	121.75	-2.74	-12.08
10.	↑ 11.	↑ 11.	Snowflake		Relational	119.88	-1.12	+5.11
11.	↓ 9.	↓ 10.	SQLite		Relational	117.95	-6.63	-14.49

Objetivo da modelagem de dados

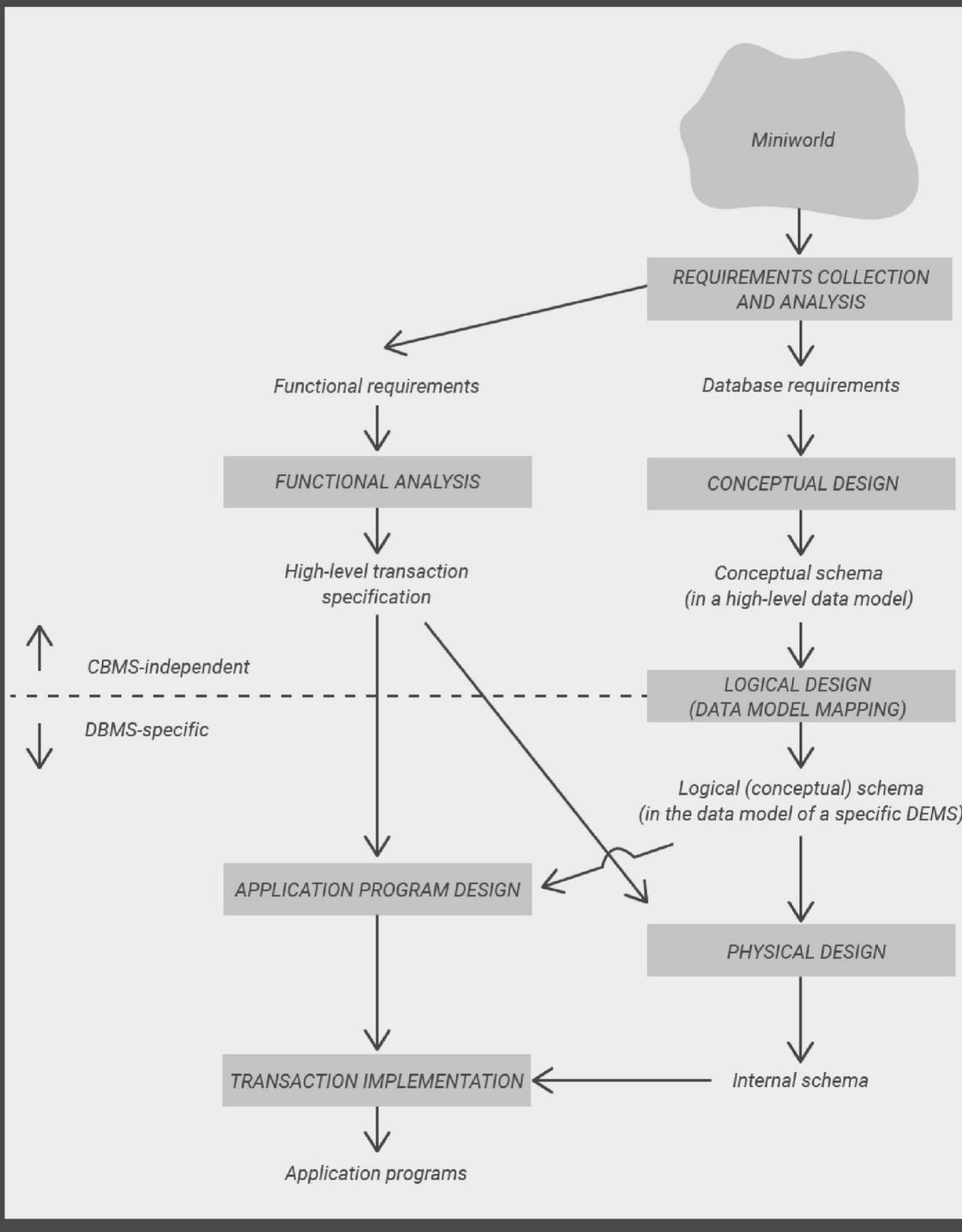
É muito comum, ao chegar em uma empresa, encontrar sistemas e bancos de dados já em operação. Então, pode surgir o questionamento: como foram projetados e implementados esses bancos de dados que são utilizados? De fato, é importante entender bem o processo de construção para que seja possível criar novos sistemas e bancos de dados para aplicações de

interesse, bem como para realizar manutenções em sistemas e bancos de dados existentes para atender a novos requisitos.

Processo de modelagem de dados

A seguir, você vai ter uma visão geral do processo envolvido no projeto de um banco de dados relacional. Um analista de sistemas, se possível acompanhado de um DBA (*database administrator*), deve realizar um levantamento completo dos dados que serão representados e manuseados pela aplicação em estudo. Conhecer os requisitos funcionais, em particular as consultas que serão feitas futuramente pelos usuários, também ajuda na especificação, evitando manutenções caras em termos de tempo e complexidade de implementação. Observe que as fases iniciais são realizadas independentemente do SGBD que será escolhido para a implementação do banco de dados.

Processo de projeto de BD



Fonte: Adaptado de Elmasri e Navathe (2010).

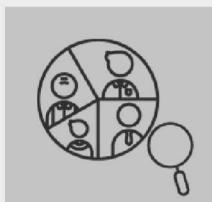
Análise de requisitos de dados (funcionais e não funcionais)

O levantamento e a análise de requisitos são a primeira etapa desse processo. Os requisitos do banco de dados são obtidos por meio de entrevistas com os produtores e os usuários dos

dados, bem como pela análise de documentos e especificações de processos. Essas informações devem ser usadas para produzir uma especificação formal de requisitos, que inclui os dados exigidos para o processamento, os relacionamentos de dados, as regras de negócio e, também, os requisitos não funcionais, como desempenho, segurança e qualidade de dados.

Os objetivos fundamentais da etapa de levantamento e análise de requisitos de dados são apresentados a seguir:

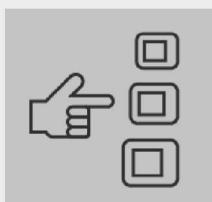
Objetivos fundamentais da etapa de levantamento e análise de requisitos



Identificar e definir os requisitos de dados essenciais para o desenvolvimento da aplicação.



Descrever a informação relativa aos elementos de dados, incluindo os relacionamentos entre eles.



Determinar os tipos de operações que devem ser executadas no banco de dados. Com isso, é possível identificar se o banco de dados terá carga predominantemente transacional (incluir, excluir, alterar e consultar elementos de dados atomicamente), analítica (incluir, excluir, alterar e consultar elementos de dados em lote) ou mista.



Definir restrições críticas (p. ex.: desempenho, integridade, segurança) e administrativas que devem ser aplicadas ao banco de dados resultante.



Especificar quaisquer restrições relacionadas ao projeto e à implementação, abrangendo tecnologias, hardware, software, linguagens de programação, políticas, padrões ou interfaces externas específicas, de acordo com a organização ou o escopo do projeto.

Esses objetivos visam estabelecer uma compreensão clara e abrangente dos requisitos de dados da aplicação a ser desenvolvida, fornecendo uma base sólida para o projeto e a implementação eficiente do banco de dados.

Então, suponha que você foi contratado para modelar os dados de uma aplicação que gerencia a inscrição de alunos em cursos de uma plataforma de cursos autoinstrucionais, ou

seja, sem professores. Durante a etapa de levantamento, você identificou os seguintes requisitos:

Requisitos de dados

SOBRE OS ESTUDANTES

Cada estudante, ao se cadastrar, recebe um número de matrícula que o identifica. Para o cadastro, é necessário informar: nome, endereço completo (CEP, logradouro, cidade, estado, número e complemento), data de nascimento, sexo e e-mail. Todos os dados são obrigatórios. O número de matrícula a ser gerado pelo sistema é formado por oito dígitos, sendo automaticamente incrementável.

Cada estudante pode se inscrever em vários cursos simultaneamente. Ao longo de um curso, o aluno deve ter seu progresso registrado em relação às atividades (leitura, assistir a vídeos, realizar exercícios, participar do fórum, completar quizzes, etc.) completadas com sucesso. Além disso, o aluno será considerado concluinte e terá direito ao certificado caso realize todas as atividades (ou seja, se atingir 100% de progresso) e acerte mais de 70% das questões da atividade final. Os alunos serão considerados desistentes caso estejam há mais de três meses sem realizar atividades em cursos em andamento.

SOBRE OS CURSOS

Cada curso tem um código, nome, descrição do objetivo, ementa, até cinco palavras-chave, idioma, categoria e carga horária. Todos os dados são obrigatórios. A categoria deve ser selecionada de uma lista fornecida pelo administrador do sistema. A carga horária mínima é de 4 horas e a máxima é de 80 horas por curso. O código identifica o curso e é formado por oito dígitos de um sequencial gerado automaticamente.

Além disso, cada curso tem um período em que as inscrições estão abertas. As inscrições podem ser reabertas sem limite de vezes.

RELATÓRIOS

Alunos inscritos no mês de referência.

Cursos com inscrições abertas no mês de referência (*).

Cursos não concluídos com o respectivo progresso por aluno (**).

Cursos concluídos com a respectiva nota final por aluno (**).

Alunos concluintes por curso com a respectiva nota final.

Alunos não concluintes por curso com o respectivo progresso e data da última atividade realizada.

Alunos desistentes por curso.

Cursos com alunos com 100% de progresso e nota final abaixo de 70% com a data da última atividade realizada.

(*) (**) O relatório pode ser gerado tanto pelos alunos quanto pelo administrador do sistema.

(**) No caso do aluno, o relatório deve exibir informações somente deste. No caso do administrador do sistema, o relatório deve exibir informações dos alunos selecionados.

Esses requisitos serão essenciais para a identificação dos elementos de dados necessários para a modelagem e construção do banco de dados dessa aplicação, como você verá a seguir.

O que são modelos de dados

Um **modelo de dados** é composto por um conjunto de elementos que descrevem a estrutura de um banco de dados de acordo com um nível de abstração. Na literatura especializada sobre modelagem de sistemas de informação, podem ser encontrados **até três níveis de abstração**.

Em uma abordagem *top-down* (ou *forward engineering*), pode-se partir de modelos mais distantes da implantação de um sistema, os chamados modelos de dados conceituais, até os modelos de dados de implementação, que podem envolver a escolha de um SGBD específico e têm a preocupação de representar os dados de maneira a incluir requisitos como segurança e desempenho.

Os três níveis da modelagem de dados

Seguindo os passos ilustrados no processo de projeto de BD, antes de projetar um banco de dados para qualquer modelo lógico, não apenas relacional, sugere-se elaborar um modelo estritamente conceitual que conte cole todos os dados do banco e a maneira como eles se relacionam.

Na etapa do projeto conceitual, o modelo de dados gerado é independente do modelo de dados lógico a ser escolhido e do SGBD particular que será considerado.

Nessa etapa, o modelo gerado é mais próximo da percepção do usuário em relação ao domínio com foco estritamente nos dados e em como eles se relacionam, valores que são considerados válidos, regras de negócio e restrições de integridade que devem ser controladas pela semântica da aplicação.

O resultado do projeto conceitual é um esquema conceitual de dados que pode ser mapeado para diversos **modelos lógicos**, como modelos em rede, de grafos ou orientados a documentos ou objetos. Esse mapeamento também é possível e bem estabelecido no caso do modelo relacional que está sendo enfatizado e será

discutido em mais detalhes ao longo das aulas. A escolha do modelo lógico a ser utilizado depende dos requisitos mapeados na fase inicial do processo.

Ao final do projeto lógico, estará disponível um esquema lógico associado a algum modelo lógico conhecido que tenha a especificação completa exigida por um modelo de dados, a saber: estrutura de organização e linguagens de manuseio dos dados. Nesta disciplina, serão enfatizados os esquemas gerados para o modelo relacional, com tabelas, atributos e restrições de integridade associadas. Nesse ponto, é necessário escolher um SGBD para implementar o projeto físico correspondente ao esquema lógico definido e que atenda aos demais requisitos de dados. O esquema físico ficará armazenado no catálogo do SGBD e sofrerá alterações com menos frequência do que a instância do banco de dados correspondente.

Modelo de entidade e relacionamento

O modelo de entidade e relacionamento (MER), proposto em 1976 por Peter Chen, é um dos mais utilizados pelos profissionais do mercado e, certamente, o processo de modelagem conceitual mais difundido. Trata-se de um modelo conceitual de dados que contempla apenas a parte de **representação da estrutura dos dados**, ou seja, não oferece mecanismos de manuseio destes. Uma das grandes vantagens do MER é permitir mais de uma representação para o mesmo conjunto de requisitos.

A modelagem por entidades e relacionamentos tem um alto grau de abstração e foi proposta com o intuito de capturar a semântica de aplicações do mundo real.

É um modelo simples e intuitivo, porém formal, que define como representar dados e relacionamentos entre eles, com regras e especificações que devem ser satisfeitas para que a modelagem seja considerada correta.

Em seguida, a partir da descrição dos requisitos de dados, você vai ver como é possível chegar ao diagrama de entidade e relacionamento (DER) correspondente. Ao longo dessa etapa, o dicionário de dados será preenchido com os elementos de dados identificados e suas características, conforme o *template* a seguir:

Dicionário de dados (*template* sugerido)

Entidades				
Nome				
Descrição				
Nome do atributo	Tipo	Domínio de valores válidos	Obrigatório (sim ou não)	Regras de negócio.
Relacionamentos				
Nome				
Entidades envolvidas				
Cardinalidade	Entidade A		Entidade B	
Descrição				
Nome do atributo	Tipo	Domínio de valores válidos	Obrigatório (sim ou não)	Regras de negócio.

Entidades e atributos

O primeiro elemento de dados que é preciso identificar são as **entidades**. Como qualquer outra abstração, definir entidade não é trivial. Pode-se dizer que se trata de algo (ou uma “coisa”) que pode ser inequivocamente identificado por um conjunto de propriedades ou características em comum. As propriedades relevantes de cada instância em uma entidade são representadas por meio de **atributos**. É fundamental entender que não existem **entidades** sem **atributos**, pois são estes que as definem. Os nomes das entidades servem apenas para facilitar sua identificação em um diagrama.

Outra forma de identificar **entidades** é selecionar da descrição dos requisitos os substantivos que representam **categorias (classes)** de objetos do mundo real de interesse para a aplicação em desenvolvimento. Esses objetos podem ser agrupados por suas **propriedades (atributos)** em comum, já que compartilham a mesma estrutura. Nos requisitos de dados, foram identificadas duas categorias de elementos de dados: **ALUNOS** e **CURSOS**.

Para identificar os atributos de cada categoria, é necessário ler os requisitos e identificar os substantivos que representam as características dos objetos dessa categoria. Os atributos podem ser mono ou multivalorados, simples ou compostos. Os atributos monovalorados são aqueles que apresentam um valor atômico por instância da entidade. Os atributos multivalorados são aqueles que podem representar mais de um valor, de mesma natureza semântica, em seu conteúdo.

A entidade ALUNOS tem os seguintes atributos monovalorados: número de matrícula, nome, endereço completo, data de nascimento, sexo e e-mail. Já a entidade CURSOS tem os seguintes atributos monovalorados: código, nome, descrição do objetivo, ementa, idioma, categoria e carga horária, além de um atributo multivalorado: palavras-chave (até cinco ocorrências).

A maioria dos atributos identificados é do tipo simples, mas o atributo endereço da entidade ALUNOS é do tipo composto.

Um atributo é estruturado ou composto quando também contém mais de um valor, como os multivalorados, porém de naturezas (semânticas) distintas.

O atributo endereço é composto pelos seguintes atributos simples: CEP, logradouro, número, complemento, cidade e estado.

Depois de identificados todos os atributos das entidades, é importante definir qual atributo deve ser usado como **identificador de cada entidade**. O atributo identificador é aquele que apresenta um valor único para cada instância de entidade. No caso de ALUNOS, é a matrícula, e para CURSOS, o código. Note que nem sempre essa informação está claramente definida nos requisitos e pode ser necessário recorrer aos usuários e aos documentos de referência coletados no levantamento de requisitos para localizar o melhor atributo ou o conjunto de atributos que irá exercer esse papel.

Além disso, **cada atributo identificado precisa da definição de seu domínio**, ou seja, o conjunto de valores válidos que podem ser atribuídos ao atributo para cada entidade. Esse domínio inclui também identificar se o atributo é obrigatório ou não e como devem ser tratados os valores fora do domínio. Veja o exemplo do campo “sexo”. Muitos formulários estabelecem somente dois valores possíveis: “feminino” e “masculino”, mas algumas pessoas não se sentem confortáveis em fornecer essa informação. Por ser um atributo obrigatório, isso impediria o cadastro ou induziria essas pessoas a responderem qualquer um dos dois. Uma alternativa encontrada em algumas aplicações é incluir a opção “Prefiro não informar”.

Acompanhe o dicionário de dados das entidades e atributos do exemplo identificados até o momento:

Dicionário de dados (entidades e atributos)

Entidades

Nome	ALUNO			
Descrição	Pessoa física que se cadastra na plataforma para realizar inscrição em cursos.			
Nome do atributo	Tipo	Domínio de valores válidos	Obrigatório (sim ou não)	Regras de negócio.
Matrícula	Mono simples identificador	Números inteiros 8 dígitos	Sim	Geração automática.
Nome	Mono simples	Texto até 100 caracteres	Sim	
Endereço	Mono composto		Sim	Composto por CEP, logradouro, bairro, cidade, estado, número e complemento.
CEP	Mono simples	Números inteiros 8 dígitos	Sim	Validar junto ao site dos Correios.
Logradouro	Mono simples	Texto até 100 caracteres	Sim	
Bairro	Mono simples	Texto até 100 caracteres	Sim	Obter por meio do CEP.
Cidade	Mono simples	Texto até 100 caracteres	Sim	
Estado	Mono simples	Texto até 100 caracteres	Sim	
Número	Mono simples	Texto até 15 caracteres	Sim	Números inteiros ou "sem número".
Complemento	Mono simples	Texto até 25 caracteres	Não	

Data de nascimento	Mono simples	Data	Sim	
Sexo	Mono simples	Feminino Masculino Prefiro não informar	Sim	
E-mail	Mono simples	Texto até 100 caracteres	Sim	Verificação da regra de formação de e-mails.
Nome	CURSO			
Descrição	Conteúdo de aprendizado autoinstrucional disponível na plataforma para alunos se inscreverem.			
Nome do atributo	Tipo	Domínio de valores válidos	Obrigatório (sim ou não)	Regras de negócio.
Código	Mono simples identificador	Números inteiros 8 dígitos	Sim	Geração automática.
Nome	Mono simples	Texto até 100 caracteres	Sim	
Objetivo	Mono simples	Texto até 500 caracteres	Sim	
Ementa	Mono simples	Texto até 500 caracteres	Sim	
Palavras-chave	Multi simples	Texto até 25 caracteres	Sim	Até cinco ocorrências por curso.
Idioma	Mono simples	Texto até 25 caracteres	Sim	Verificação de acordo com as ISOs 391 e 392.
Categoria	Mono simples	Texto até 25 caracteres	Sim	Verificar na lista de valores fornecida pelo administrador do sistema.
Carga horária	Mono simples	Inteiro com 2 dígitos Mínimo 4 e máximo 80	Sim	
Data início inscrição	Mono simples	Data	Não	Maior ou igual à data corrente. Quando não preenchida, significa que não foi definido o início da inscrição.
Data final inscrição	Mono simples	Data	Não	Maior que a data de início da inscrição. Quando não preenchida, significa que não foi definido o fim da inscrição.

Relacionamentos, atributos e cardinalidades

Como o próprio nome diz, em um MER há entidades e **relacionamentos**.

Relacionamentos são associações entre duas ou mais instâncias de entidades com um significado dentro da aplicação a ser desenvolvida.

Cada tipo de relacionamento define um conjunto de associações entre dois ou mais tipos de entidades. No caso da aplicação do exemplo que está sendo modelado, devem-se procurar nos requisitos de dados quais relacionamentos (normalmente representados por meio de verbos) existem entre as entidades ALUNO e CURSO. O relacionamento identificado na frase “Cada estudante pode se inscrever em vários cursos simultaneamente” é INSCRITO_EM (ALUNO, CURSO).



Fique Ligado

Relacionamentos, assim como entidades, também podem ter propriedades inerentes a cada instância da associação entre as instâncias das entidades. Essas propriedades são necessárias para descrever o relacionamento dentro do domínio da aplicação que está sendo modelada. No caso da aplicação de controle de inscrições na plataforma, podem-se identificar os seguintes atributos (analisando o texto dos requisitos de dados): data da inscrição, percentual de progresso, nota final e data da última atividade.

Uma característica particular dos relacionamentos é apresentar, além dos atributos, **cardinalidades mínimas e máximas**. Essas cardinalidades são uma característica que limita as possíveis combinações de instâncias de entidades que podem participar no conjunto de relacionamentos. A participação – ou seja, se uma entidade obrigatoriamente participa do tipo de relacionamento com outra entidade – é definida com a cardinalidade mínima, que pode ser 0 caso a participação não seja obrigatória. Nessa aplicação, tem-se que:

- um aluno pode se matricular em um, nenhum ou vários cursos;
- um curso pode ter nenhum ou vários alunos matriculados.

Pela descrição dos requisitos, é possível que um aluno se cadastre na plataforma, mas seu cadastro não está condicionado a se inscrever em algum curso. Além disso, se um aluno está inscrito, está fazendo ou já concluiu um curso, isso não impede que ele se inscreva em outros. Da mesma forma, um curso pode ser cadastrado na plataforma e ainda não ter as inscrições abertas, caso em que não teria nenhum aluno inscrito. Mesmo depois de abertas as inscrições, pode ocorrer de nenhum aluno se inscrever no período de abertura. Como não foi especificado um limite de alunos inscritos por curso, a cardinalidade mínima e máxima tanto de ALUNO quanto de CURSO, no relacionamento “Inscrito em”, é [0,n].

Dicionário de dados (relacionamentos)

Relacionamentos

Nome	Inscrito em			
Entidades envolvidas	ALUNO CURSO			
Cardinalidade	ALUNO	Mínimo 0 Máximo N	CURSO	Mínimo 0 Máximo N
Descrição	Um aluno pode se matricular em um, nenhum ou vários cursos. Um curso pode ter nenhum ou vários alunos matriculados.			
Nome do atributo	Type	Domínio de valores válidos	Obrigatório (sim ou não)	Regras de negócio.
Data inscrição	Mono simples	Data	Sim	Criada com a data corrente. Não sofre atualização.
Percentual de progresso	Mono simples	Real	Não	Valor inicial igual a 0. Atualizada sempre que o aluno conclui uma atividade no curso. Corresponde ao número de atividades concluídas dividido pelo número de atividades do curso.
Nota final	Mono simples	Inteiro	Não	Valor inicial igual a 0. Só pode ser atualizada depois que o progresso das atividades atingir 100%.
Data da última atividade	Mono simples	Data	Não	Atualizada com a data corrente sempre que o aluno conclui uma atividade no curso.

Construção de um diagrama de entidade e relacionamento

Um **esquema** corresponde à descrição gráfica ou textual da estrutura de um banco de dados de acordo com um determinado modelo de dados. Como forma de linguagem, adotou-se uma representação visual conhecida como **diagrama de entidade e relacionamento (DER)**, que facilita a compreensão do resultado da modelagem e as trocas entre usuários-desenvolvedores e usuários não técnicos.

Assim, o DER representa o esquema conceitual definido pelo MER utilizando símbolos com semântica formalmente definida.

Observe no exemplo:

Construtos do modelo E-R

Símbolo	Significado
	<i>ENTITY</i>
	<i>WEAK ENTITY</i>
	<i>RELATIONSHIP</i>
	<i>IDENTIFYING RELATIONSHIP</i>
	<i>ATTRIBUTE</i>
	<i>KEY ATTRIBUTE</i>
	<i>MULTIVALUED ATTRIBUTE</i>
	<i>COMPOSITE ATTRIBUTE</i>
	<i>DERIVED ATTRIBUTE</i>
	<i>TOTAL PARTICIPATION OF E₂ IN R</i>
	<i>CARDINALITY RATIO 1: N FOR E₁ E₂ IN R</i>
	<i>STRUCTURAL CONSTRAINT (min, max) ON PARTICIPATION OF E IN R</i>



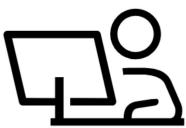
Fique Ligado

Outros tipos de recursos de modelagem conceitual, como relacionamentos múltiplos, entidade fraca, entidade associativa, atributos derivados, generalização ou especialização e autorrelacionamentos (papéis), serão estudados na disciplina de banco de dados.

Algumas representações visuais podem ter variações. Será adotada a notação utilizada pela ferramenta brModelo (figura a seguir). Para utilizar essa ferramenta, é preciso baixar o arquivo .zip da área de [download do site](#) para seu computador, descompactar o conteúdo e entrar na pasta dist. Nessa pasta, basta dar duplo clique sobre o arquivo da aplicação (brModelo.jar).

Elementos para construção do DER no brModelo



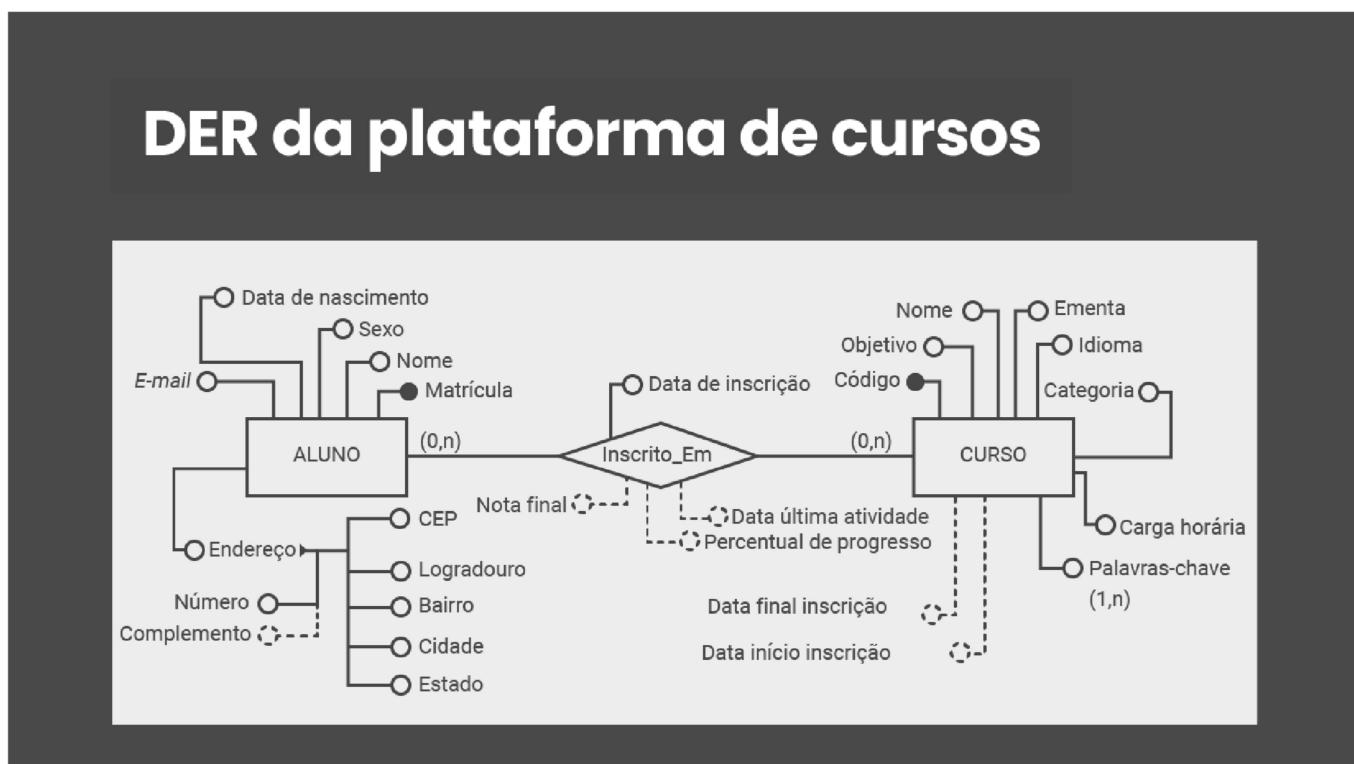


Utilização do brModelo para criar um DER

O vídeo a seguir mostra como a criação do DER é realizada utilizando essa ferramenta.



A imagem a seguir apresenta o DER resultante do vídeo.





Aprenda mais

Conheça um pouco da história do brModelo clicando [aqui](#).

Existe também uma versão web da ferramenta com suas funcionalidades essenciais ([clique aqui](#)).

Depois de elaborar o dicionário de dados e o DER, o analista de requisitos deve obter a **aprovação dos artefatos** por parte da área usuária. É recomendada uma reunião para apresentá-los, sanar as dúvidas de interpretação e anotar os itens para correção e ajustes. Com a versão final aprovada de ambos, a probabilidade de erros na análise de requisitos de dados diminuirá e os erros não serão propagados para as fases posteriores do projeto, diminuindo o retrabalho.

O mapeamento de um esquema conceitual em um esquema lógico do modelo relacional envolve transformar cada elemento de um esquema de entidades e relacionamentos (entidades, relacionamentos e atributos) em elementos equivalentes da modelagem relacional (tabelas, atributos e restrições estruturais).

Desde que o modelo de dados relacional foi proposto, na década de 1970, houve questionamentos sobre quais e quantas tabelas deveriam ser criadas para cada sistema computacional. Uma única tabela já atenderia aos requisitos? Se optar por múltiplas tabelas, quantas e quais devem ser propostas? Serão discutidas estratégias para obter um conjunto de relações considerado adequado para sistemas relacionais em aplicações reais.

Diferentemente do modelo conceitual, o modelo relacional contempla as duas partes que compõem qualquer modelo de dados: estrutura (no caso, relações matemáticas) e manuseio por meio de linguagens (p. ex., SQL ou álgebra relacional) projetadas para estruturas relacionais.

Esses mapeamentos e seus desdobramentos, bem como uma introdução à linguagem SQL, serão o foco da próxima aula.