



# Banco de Dados Relacional: Modelagem e Normalização

IFBA – Campus Ilhéus  
Curso: Back End com Python  
Prof.<sup>a</sup>: Karine Prado

# Objetivo da Aula

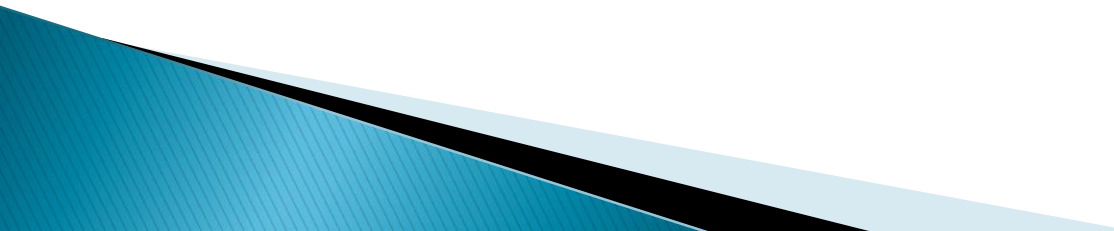
- ▶ **Objetivo:** Ensinar os alunos a estruturar dados de forma eficiente para evitar redundância e inconsistência.



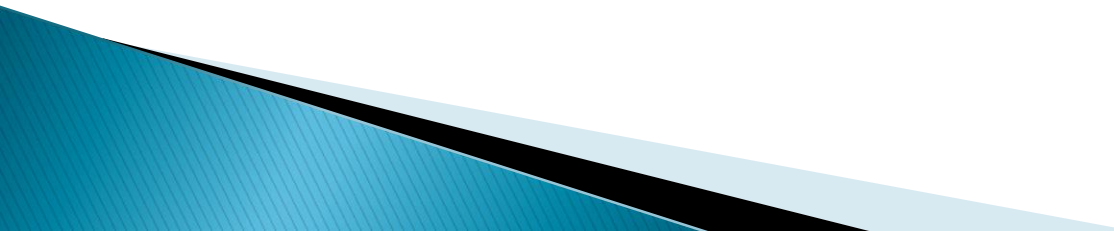
# Conceitos de Modelagem de Dados

- ▶ A modelagem de dados é o processo de representar graficamente as estruturas de dados que serão usadas em um banco de dados. Ela ajuda a entender quais dados serão armazenados, como eles se relacionam e como serão organizados.

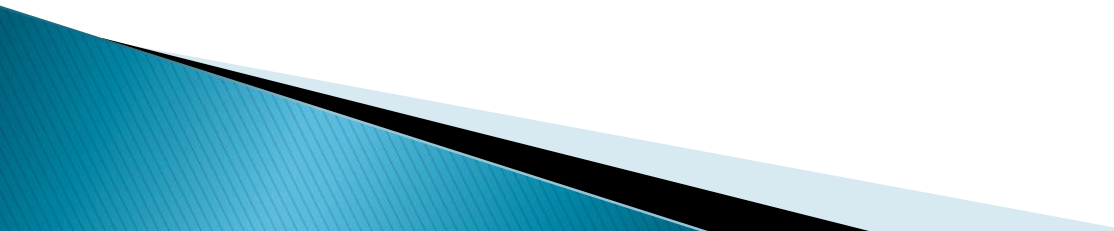
# Conceitos de Modelagem de Dados

- ▶ Entidade: representa um objeto real (ex: Aluno, Curso).
  - ▶ Atributo: característica da entidade (ex: nome, CPF).
  - ▶ Relacionamento: ligação entre entidades.
- 

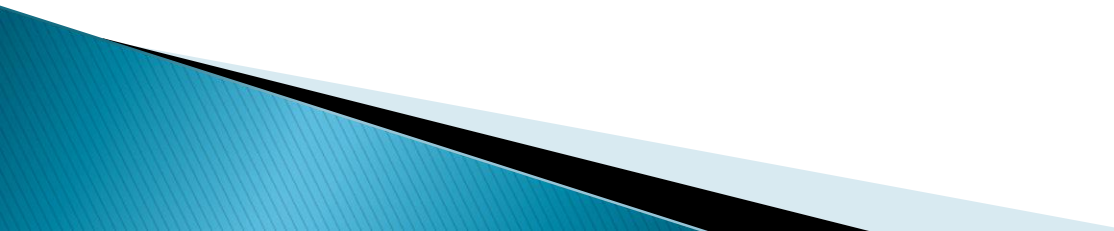
# Conceitos de Modelagem de Dados

- ▶ A modelagem é feita em níveis:
    - Conceitual: visão geral dos dados, sem se preocupar com o tipo de banco ou sistema.
    - Lógico: representação mais próxima da estrutura relacional (tabelas, chaves, tipos de dados).
    - Físico: como os dados serão armazenados fisicamente no SGBD (índices, partições etc.).
- 

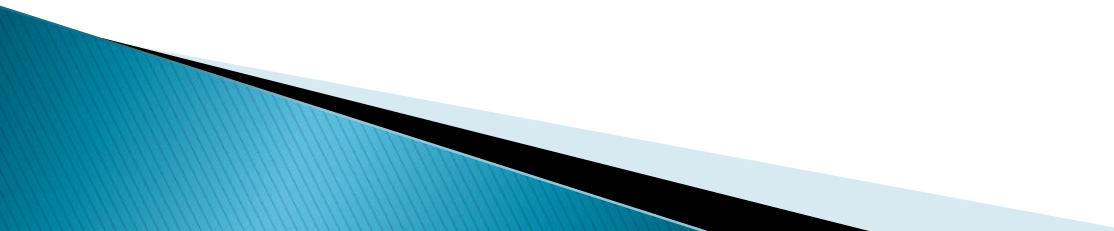
# O que é Normalização?

- ▶ Processo que organiza os dados para evitar redundâncias.
  - ▶ Evita anomalias na inserção, exclusão e atualização.
  - ▶ Trabalha em etapas chamadas de Formas Normais.
- 

# 1ª Forma Normal (1FN)

- ▶ Elimina grupos de dados repetidos.
  - ▶ Garante que cada campo tenha um valor atômico.
  - ▶ Ex ruim: uma coluna chamada "Telefones" com dois números separados por vírgula.
  - ▶ Correto: separar em outra tabela.
- 

## 2ª Forma Normal (2FN)

- ▶ Requer que a tabela esteja na 1FN.
  - ▶ Remove dependências parciais (todos os atributos não-chave devem depender de toda a chave primária, e não só de parte dela.)
    - Isso só acontece em tabelas onde a chave primária é composta (ou seja, formada por mais de uma coluna).
- 



# 2ª Forma Normal (2FN)

## ▶ Exemplo – Tabela Inscrições:

<u>AlunoID</u>	<u>CursoID</u>	NomeAluno	NomeCurso	DataInscrição
1	101	João	Python	10/06/2025
1	102	João	Java	11/06/2025
2	101	Maria	Python	12/06/2025

## ▶ O que está errado?

- NomeAluno depende só do AlunoID, e não do par completo (AlunoID, CursoID).
- NomeCurso depende só do CursoID.
- Ou seja, temos dependências parciais.

# 2ª Forma Normal (2FN)

- ▶ Como normalizar para 2FN?
  - Separar as informações em 3 tabelas: Aluno, Curso e Inscrição.

Agora, todos os atributos em "Inscrição" dependem da chave composta completa. Isso é 2FN!

<u>AlunoID</u>	NomeAluno
1	João
2	Maria

<u>CursoID</u>	NomeCurso
101	Python
102	Java

<u>AlunoID</u>	<u>CursoID</u>	DataInscrição
1	101	10/06/2025
1	102	11/06/2025
2	101	12/06/2025

# 3ª Forma Normal (3FN)

- ▶ Requer que a tabela esteja na 2FN.
- ▶ Elimina dependências transitivas, ou seja, quando um atributo depende de outro atributo que não é chave primária.

# 3ª Forma Normal (2FN)

## ▶ Exemplo – Tabela Funcionário:

<u>FuncID</u>	NomeFunc	CPF	Departamento	LocalDepartamento
1	Ana	123.456.789	RH	Prédio A
2	João	987.654.321	TI	Prédio B

## ▶ O que está errado?

- LocalDepartamento depende de Departamento, e Departamento depende da chave FuncID.
- Logo, LocalDepartamento depende indiretamente da chave primária.
- Isso é uma dependência transitiva, o que viola a 3FN.

# 3ª Forma Normal (2FN)

- ▶ Como normalizar para 3FN?
  - Separar o departamento em uma tabela própria:

<u>Departamento</u>	LocalDepartamento
RH	Prédio A
TI	Prédio B

<u>FuncID</u>	NomeFunc	CPF	Departamento
1	Ana	123.456.789	RH
2	João	987.654.321	TI

Agora cada campo depende diretamente da chave da tabela em que está. Isso é 3FN!

# Impacto da Modelagem

## ▶ Performance:

- Modelagem bem feita melhora a eficiência nas consultas e evita dados duplicados;
- Porém, excesso de normalização pode tornar as consultas mais lentas, pois exige muitas junções (joins).

## ▶ Integridade dos Dados:

- Modelagem correta, com o uso de chaves primárias, estrangeiras e restrições, garante que os dados armazenados sejam coerentes e corretos.

# Exercícios

- » Para Identificar e Normalizar Tabelas Simples

# Exercício 1

- ▶ Tabela original (não normalizada):

<u>PedidoID</u>	ClienteNome	<u>ProdutoNome</u>	Quantidade	PreçoUnitário	ClienteTelefone
1	Ana Lima	Caneta	2	3.00	(71)99888-1234
1	Ana Lima	Lápis	3	1.50	(71)99888-1234

1. Identifique problemas de redundância.
2. Aplique 1FN, 2FN e 3FN.
3. Apresente as novas tabelas com chaves primárias e estrangeiras.



# Exercício 1 – Resolução

- ▶ Tabela original (não normalizada):

<u>PedidoID</u>	ClienteNome	<u>ProdutoNome</u>	Quantidade	PreçoUnitário	ClienteTelefone
1	Ana Lima	Caneta	2	3.00	(71)99888-1234
1	Ana Lima	Lápis	3	1.50	(71)99888-1234

## 1. Problemas de redundância:

O nome e o telefone da cliente se repetem para cada item do pedido. Isso gera risco de dados inconsistentes se forem alterados em apenas uma linha.

# Exercício 1 – Resolução

- ▶ Tabela original (não normalizada):

<u>PedidoID</u>	ClienteNome	<u>ProdutoNome</u>	Quantidade	PreçoUnitário	ClienteTelefone
1	Ana Lima	Caneta	2	3.00	(71)99888-1234
1	Ana Lima	Lápis	3	1.50	(71)99888-1234

## 2. Aplicação das Formas Normais:

1FN: Já está com dados atômicos (sem repetições em um mesmo campo). Nenhuma coluna contém múltiplos valores por célula.

# Exercício1 – Resolução

- ▶ Tabela original (não normalizada):

<u>PedidoID</u>	ClienteNome	<u>ProdutoNome</u>	Quantidade	PreçoUnitário	ClienteTelefone
1	Ana Lima	Caneta	2	3.00	(71)99888-1234
1	Ana Lima	Lápis	3	1.50	(71)99888-1234

## 2. Aplicação das Formas Normais:

2FN: ClienteNome e ClienteTelefone dependem apenas de PedidoID, e não do par completo.

Correção: Separar as informações do cliente em uma tabela associada ao pedido.

# Exercício 1 – Resolução

- ▶ Tabela original (não normalizada):

<u>PedidoID</u>	ClienteNome	<u>ProdutoNome</u>	Quantidade	PreçoUnitário	ClienteTelefone
1	Ana Lima	Caneta	2	3.00	(71)99888-1234
1	Ana Lima	Lápis	3	1.50	(71)99888-1234

## 2. Aplicação das Formas Normais:

3FN: ClienteTelefone depende de ClienteNome, que depende do PedidoID. Ou seja, temos uma dependência transitiva.

Correção: Criar uma tabela de Cliente separada com um ID.

# Exercício 1 – Resolução

## 3. Modelo Final (em 3FN)

Cliente	<u>ClienteID</u>	ClienteNome	ClienteTelefone
	1	Ana Lima	(71)99888-1234

Produto	<u>ProdutoID</u>	ProdutoNome
	1	Caneta
	2	Lápis

Pedido	<u>PedidoID</u>	ClienteID →
	1	1

ItemPedido	<u>PedidoID</u> →	<u>ProdutoID</u> →	Quantidade	PreçoUnitário
	1	1	2	3.00
	1	2	3	1.50

# Exercício 2

- ▶ Uma escola deseja armazenar informações sobre os cursos que oferece e os alunos matriculados. O analista criou a seguinte tabela inicial (não normalizada):

AlunoNome	AlunoCPF	CursoNome	Professor	ProfessorEmail	Turma	Horario	Nota
Ana Lima	111.222.333-44	Banco de Dados	Karine Prado	<a href="mailto:karine@ifba.edu.br">karine@ifba.edu.br</a>	T01	Seg e Qua 19h	9.5
Ana Lima	111.222.333-44	Python Básico	João Silva	<a href="mailto:joao@ifba.edu.br">joao@ifba.edu.br</a>	T02	Ter e Qui 19h	8.0
Lucas Souza	555.666.777-88	Banco de Dados	Karine Prado	<a href="mailto:karine@ifba.edu.br">karine@ifba.edu.br</a>	T01	Seg e Qua 19h	7.5

# Exercício 2

## ► Tarefas:

- Identifique problemas de redundância e dependências indevidas nessa tabela.
- Apresente o modelo final (em 3FN), com as tabelas resultantes, suas chaves primárias (PK) e chaves estrangeiras (FK).

# Exercício 2 – Solução

## ▶ Aluno

- AlunoID (PK)
- Nome
- CPF (UNIQUE)

## ▶ Professor

- ProfessorID (PK)
- Nome
- Email (UNIQUE)

## ▶ Curso

- CursoID (PK)
- Nome

## ▶ Turma

- TurmaID (PK)
- CursoID (FK → Curso.CursoID)
- ProfessorID (FK → Professor.ProfessorID)
- NomeTurma (ex.: "T01")
- Horario (ex.: "Seg e Qua 19h")

## ▶ Matricula (ou Inscricao)

- AlunoID (FK → Aluno.AlunoID)
- TurmaID (FK → Turma.TurmaID)
- Nota
- PK composta: (AlunoID, TurmaID)



# Exercício 3

AlunoNome	AlunoMatricula	CursoNome	EmpresaNome	EmpresaCNPJ	SupervisorNome	SupervisorEmail	CargaHoraria	Bolsa	DataInicio	DataFim
Ana Lima	2023001	Informática	TechData	12.345.678/0001-99	Carlos Souza	<a href="mailto:carlos@techdata.com">carlos@techdata.com</a>	200	800.00	01/09/2025	30/11/2025
Bruno Silva	2023002	Edificações	Constrular	98.765.432/0001-10	Paula Reis	<a href="mailto:paula@constrular.com">paula@constrular.com</a>	180	900.00	01/09/2025	30/11/2025
Ana Lima	2023001	Informática	Constrular	98.765.432/0001-10	Paula Reis	<a href="mailto:paula@constrular.com">paula@constrular.com</a>	150	700.00	01/02/2026	30/04/2026