Conversão do Modelo Conceitual para o Modelo Lógico

Traduzindo Diagramas E/R em Tabelas de Banco de Dados

Aula 03 - Unidade Curricular 3

Banco de Dados e Modelagem

SQL e Estruturas de Dados

Objetivos da Aula

Ao final desta aula, você será capaz de:

OBJETIVO 1

Compreender a Diferença entre Modelos

Entender claramente a distinção entre o **modelo conceitual** (diagrama E/R - o que pensamos) e o **modelo lógico** (tabelas SQL - o que o computador entende), reconhecendo o papel de cada um no processo de desenvolvimento de banco de dados.

OBJETIVO 2

Dominar as Regras de Conversão

Aplicar com segurança as **quatro regras de ouro** da conversão: Entidade → Tabela, Atributo → Coluna, Chave Primária → PK, e Relacionamento → Chave Estrangeira (FK), traduzindo corretamente diagramas E/R em estruturas SQL.

OBJETIVO 3

Mapear Relacionamentos Corretamente

Implementar corretamente relacionamentos **1:N** (um para muitos) adicionando chaves estrangeiras no lado "N", e relacionamentos **N:M** (muitos para muitos) criando tabelas associativas com chaves primárias compostas.

OBJETIVO 4

Definir Tipos e Restrições

Escolher **tipos de dados apropriados** (VARCHAR, INT, DECIMAL, DATE) para cada coluna e aplicar **restrições adequadas** (NOT NULL, UNIQUE, DEFAULT) para garantir integridade e qualidade dos dados no banco.

Do Conceitual ao Lógico

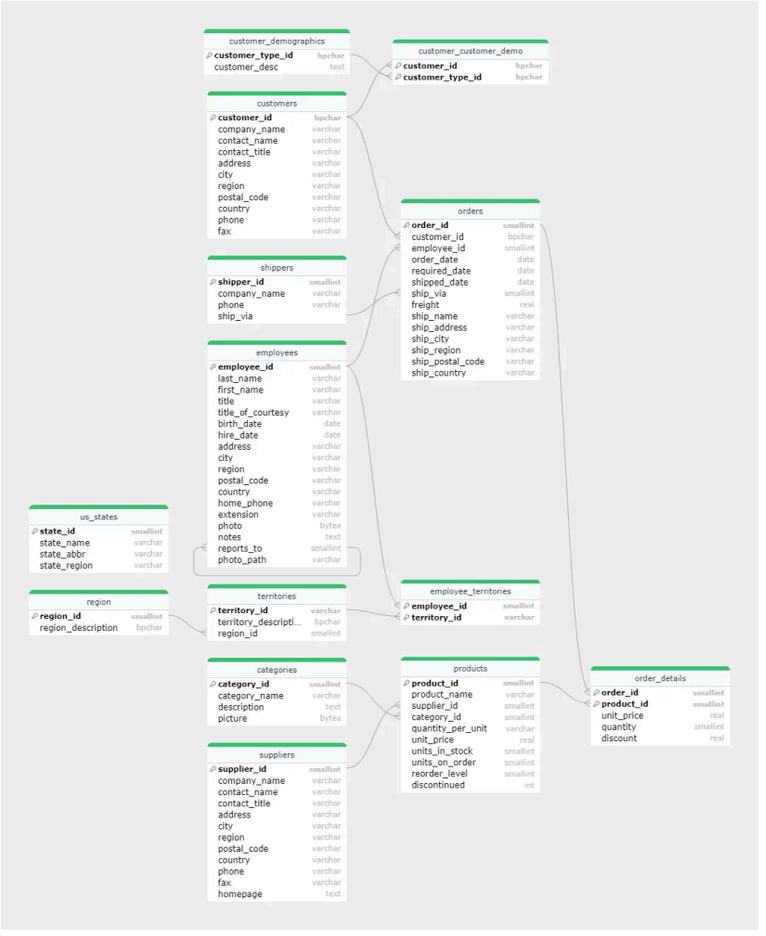
> ***"O modelo conceitual (o desenho) é o que pensamos. O modelo lógico (as tabelas) é o que o computador entende."***

🎯 Objetivo da Conversão

Traduzir o **Diagrama E/R** (Entidade-Relacionamento) criado na aula anterior para a **estrutura SQL** que será implementada no banco de dados. Esta conversão transforma conceitos abstratos em tabelas concretas que o sistema de gerenciamento de banco de dados pode processar.

⚠️ Importância da Conversão Correta

Uma conversão **incorreta** pode resultar em perda de dados, redundância desnecessária, dificuldade de manutenção e problemas de integridade. Seguir as regras de conversão garante um banco de dados **eficiente, organizado e confiável**.



Exemplo de conversão: Diagrama E/R transformado em código SQL

As Regras de Ouro da Conversão E/R → SQL

1 ENTIDADE → TABELA

Cada Entidade do Diagrama E/R torna-se uma Tabela no modelo lógico.

EXEMPLO:

```
E/R: Entidade CLIENTE
SQL: Tabela Cliente
```

2 ATRIBUTO → COLUNA

Cada Atributo da Entidade torna-se uma Coluna na Tabela correspondente.

EXEMPLO:

```
E/R: Atributos Nome, CPF, Email
SQL: Colunas Nome, CPF, Email
```

3 CHAVE PRIMÁRIA → PK DA TABELA

A Chave Primária da Entidade se torna a PK (Primary Key) da Tabela, identificando unicamente cada registro.

EXEMPLO:

```
E/R: ID_Cliente (sublinhado)
SQL: ID_Cliente INT PRIMARY KEY
```

4 RELACIONAMENTO → CHAVE ESTRANGEIRA (FK)

O Relacionamento entre Entidades é implementado pela adição de uma Chave Estrangeira (Foreign Key). **Esta é a regra mais crucial!**

EXEMPLO 1:N:

```
Cliente (1) → Pedido (N)
SQL: Tabela Pedido recebe FK_ID_Cliente
```

Tipos de Dados em SQL

Escolher o tipo correto garante integridade, otimiza espaço e melhora performance

Tipos de Texto

VARCHAR(n)

Texto de tamanho **variável**. Usa apenas o espaço necessário.

CHAR(n)

Texto de tamanho **fixo**. Sempre usa n caracteres.

TEXT

Texto **longo** (até 65.535 caracteres).

Tipos Numéricos

INT

Números **inteiros**. Ideal para IDs, idade, quantidade.

DECIMAL(p,s)

SEMPRE use para **valores monetários**! Precisão exata.

FLOAT/DOUBLE

Decimais com precisão **aproximada**. Nunca use para dinheiro!

Tipos Data/Hora

DATE

Apenas a **data** (YYYY-MM-DD). Ex: 2024-10-15

DATETIME

Data e hora (YYYY-MM-DD HH:MM:SS).

TIME

Apenas a **hora** (HH:MM:SS).

Exemplos Práticos de Aplicação

Coluna	Tipo de Dado	Justificativa
Nome	VARCHAR(100)	Nomes raramente excedem 100 caracteres
CPF	VARCHAR(11)	CPF tem 11 dígitos (use VARCHAR, não INT, pois pode começar com 0)
Preco	DECIMAL(10,2)	Valores monetários com 2 casas decimais (ex: R$ 1234.56)
Idade	INT	Idade é sempre número inteiro
Data_Nascimento	DATE	Só precisa da data, não da hora
Data_Pedido	DATETIME	Precisa da data e hora exatas do pedido

Restrições Básicas em SQL

Garantindo Integridade e Qualidade dos Dados

NOT NULL

Garante que a coluna **NUNCA ficará vazia**. O valor deve ser sempre fornecido.

QUANDO USAR:

- Campos obrigatórios
- Informações essenciais
- Chaves primárias (sempre)

Exemplos:

• Nome do cliente
• CPF
• E-mail
• Data de cadastro

UNIQUE

Garante que todos os valores na coluna serão **DIFERENTES** (únicos).

QUANDO USAR:

- CPF
- E-mail
- Matrícula
- Código de produto

Diferença PK vs UNIQUE:

• PK: 1 por tabela
• UNIQUE: várias colunas

DEFAULT

Define um **valor PADRÃO** caso a informação não seja fornecida.

QUANDO USAR:

- Status inicial
- Datas automáticas
- Valores comuns
- Flags booleanas

Exemplos:

• Status: 'Pendente'
• Data: CURRENT_DATE
• Ativo: TRUE

Exemplo Prático: Tabela Cliente com Restrições

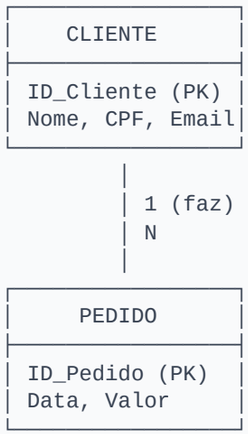
Coluna	Tipo de Dado	Restrições	Valor Padrão	Justificativa
ID_Cliente	INT	PK, NOT NULL	AUTO_INCREMENT	Chave primária sempre obrigatória
Nome	VARCHAR(100)	NOT NULL	-	Nome é obrigatório para cadastro
CPF	VARCHAR(11)	NOT NULL, UNIQUE	-	CPF obrigatório e único por pessoa
Email	VARCHAR(100)	NOT NULL, UNIQUE	-	E-mail obrigatório e único para login
Telefone	VARCHAR(15)	-	-	Telefone é opcional
Data_Cadastro	DATE	NOT NULL	CURRENT_DATE	Data preenchida automaticamente
Ativo	BOOLEAN	NOT NULL	TRUE	Cliente ativo por padrão

Mapeamento de Relacionamentos 1:N (Um para Muitos)

Conceito e Regra de Ouro

Relacionamento 1:N significa que uma entidade do lado "1" se relaciona com **MUITAS** entidades do lado "N", mas cada entidade do lado "N" se relaciona com **APENAS UMA** entidade do lado "1". **REGRA: A Chave Estrangeira (FK) vai para o lado "N" (lado "muitos")**.

Exemplo: Cliente → Pedido



Interpretação:
- Um cliente → MUITOS pedidos
- Cada pedido → UM cliente

Passo a Passo da Conversão

PASSO 1: Converter Entidades em Tabelas

PASSO 2: Converter Atributos em Colunas

PASSO 3: Identificar: Cliente (1) — Pedido (N)

PASSO 4: FK vai para o lado "N" (Pedido)

PASSO 5: Copiar PK de Cliente como FK em Pedido

RESULTADO: Adicionar **FK_ID_Cliente** na tabela Pedido

ANTES: Tabelas sem relacionamento

Tabela: Cliente

ID_Cliente	Nome	CPF	Email
1	Ana Silva	12345678901	ana@email.com
2	Bruno Costa	98765432109	bruno@email.com

Tabela: Pedido

ID_Pedido	Data_Pedido	Valor_Total	Status
100	2024-01-10	150.00	Pago
101	2024-01-12	200.00	Pendente

DEPOIS: Com FK implementada

Tabela: Cliente (sem alteração)

ID_Cliente	Nome	CPF	Email
1	Ana Silva	12345678901	ana@email.com
2	Bruno Costa	98765432109	bruno@email.com

Tabela: Pedido (com FK adicionada)

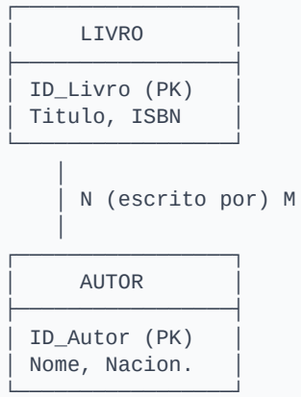
ID_Pedido	Data_Pedido	Valor_Total	Status	FK_ID_Cliente
100	2024-01-10	150.00	Pago	**1**
101	2024-01-12	200.00	Pendente	**2**

Mapeamento de Relacionamentos N:M (Muitos para Muitos)

Conceito e Solução com Tabela Associativa

Relacionamento N:M significa que uma entidade se relaciona com **MUITAS** entidades do outro lado, e vice-versa. **SOLUÇÃO: Criar uma TABELA ASSOCIATIVA (ou tabela de ligação) com as PKs de ambas as entidades como FKs, formando uma PK composta**.

Exemplo: Livro ↔ Autor



Problema:
- Livro → MUITOS autores
- Autor → MUITOS livros
- Não podemos colocar FK diretamente!

Passo a Passo da Conversão

PASSO 1: Converter Entidades em Tabelas

PASSO 2: Converter Atributos em Colunas

PASSO 3: Identificar: Livro (N) — Autor (M)

PASSO 4: Criar TABELA ASSOCIATIVA: Livro_Autor

PASSO 5: Adicionar FK_ID_Livro e FK_ID_Autor

PASSO 6: PK COMPOSTA: (FK_ID_Livro, FK_ID_Autor)

RESULTADO: 3 tabelas: Livro, Autor, Livro_Autor

Tabela: Livro (original)

ID_Livro	Titulo	ISBN
1	Clean Code	9780132350884
2	Design Patterns	9780201633610
3	Refactoring	9780134757599

Tabela: Autor (original)

ID_Autor	Nome	Nacionalidade
1	Robert Martin	EUA
2	Erich Gamma	Suíça
3	Martin Fowler	Reino Unido

Tabela: Livro_Autor (ASSOCIATIVA)

FK_ID_Livro	FK_ID_Autor
1	1
2	2
3	3

PK Composta: (FK_ID_Livro, FK_ID_Autor)
Interpretação:
- Livro 1 escrito por Autor 1
- Livro 2 escrito por Autor 2
- Livro 3 escrito por Autor 3

Atividade Prática - Parte 1: Análise do Diagrama E/R

Instruções Gerais

Trabalho em grupos de 3-4 alunos | Tempo estimado: **30 minutos** | Cada grupo receberá um dos cenários abaixo com o diagrama E/R completo

Cenário A: Sistema de Biblioteca

ENTIDADES:

- **LIVRO:** ID_Livro, Titulo, ISBN, Ano_Publicacao, Editora, Num_Paginas
- **AUTOR:** ID_Autor, Nome, Nacionalidade, Data_Nascimento
- **EMPRESTIMO:** ID_Emprestimo, Data_Emprestimo, Data_Devolucao, Status
- **USUARIO:** ID_Usuario, Nome, CPF, Email, Telefone, Data_Cadastro

RELACIONAMENTOS:

- **LIVRO ↔ AUTOR (N:M)** - Um livro pode ter muitos autores, um autor pode escrever muitos livros
- **LIVRO → EMPRESTIMO (1:N)** - Um livro pode ter muitos empréstimos
- **USUARIO → EMPRESTIMO (1:N)** - Um usuário pode fazer muitos empréstimos

Cenário B: Sistema de Loja Virtual

ENTIDADES:

- **CLIENTE:** ID_Cliente, Nome, CPF, Email, Telefone, Endereco
- **PEDIDO:** ID_Pedido, Data_Pedido, Valor_Total, Status, Forma_Pagamento
- **PRODUTO:** ID_Produto, Nome_Produto, Descricao, Preco_Unitario, Estoque
- **CATEGORIA:** ID_Categoria, Nome_Categoria, Descricao

RELACIONAMENTOS:

- **CLIENTE → PEDIDO (1:N)** - Um cliente pode fazer muitos pedidos
- **PEDIDO ↔ PRODUTO (N:M)** - Um pedido pode ter muitos produtos, um produto pode estar em muitos pedidos
- **CATEGORIA → PRODUTO (1:N)** - Uma categoria contém muitos produtos

Tarefa 1: Listar Todos os Componentes do Diagrama E/R

Atividade Prática - Parte 2

Detalhamento de Tabelas, Tipos de Dados e Restrições

Tarefa 2: Completar o Modelo Lógico

1

Definir Tipos de Dados

Para cada coluna, escolha o tipo apropriado:

- **VARCHAR(n)** para textos
- **INT** para inteiros
- **DECIMAL(10,2)** para valores monetários
- **DATE/DATETIME** para datas

Justifique cada escolha!

2

Definir Restrições

Aplique as restrições adequadas:

- **PK** para chaves primárias
- **FK** para chaves estrangeiras
- **NOT NULL** para campos obrigatórios
- **UNIQUE** para valores únicos (CPF, Email)

Valide cada restrição!

3

Definir Valores Padrão

Quando aplicável, defina valores DEFAULT:

- **CURRENT_DATE** para datas automáticas
- **'Ativo'** para status inicial
- **TRUE** para flags booleanas
- **0 ou 0.00** para valores numéricos

Automatize quando possível!

Modelo de Preenchimento: Tabela Cliente (Exemplo Completo)

Tabela	Coluna	Tipo de Dado	Restrições	Valor Padrão	Justificativa
Cliente	ID_Cliente	INT	PK, AUTO_INCREMENT	-	Identificador único numérico
	Nome	VARCHAR(100)	NOT NULL	-	Nome completo obrigatório
	CPF	VARCHAR(11)	NOT NULL, UNIQUE	-	CPF obrigatório e único por pessoa
	Email	VARCHAR(100)	NOT NULL, UNIQUE	-	E-mail obrigatório e único para login
	Telefone	VARCHAR(15)	-	-	Telefone é opcional
	Data_Cadastro	DATE	NOT NULL	CURRENT_DATE	Data preenchida automaticamente

Checklist de Validação - Verifique se você:

- ☑ Definiu tipos de dados para todas as colunas
- ☑ Identificou todas as PKs e FKs
- ☑ Aplicou NOT NULL nos campos obrigatórios
- ☑ Aplicou UNIQUE em CPF e Email
- ☑ Definiu valores DEFAULT apropriados
- ☑ Justificou todas as escolhas

Síntese e Consolidação

ENTIDADE
↓
TABELA

ATRIBUTO
↓
COLUNA

CHAVE PRIMÁRIA
↓
PK DA TABELA

RELACIONAMENTO
↓
CHAVE ESTRANGEIRA

Checklist de Validação do Modelo Lógico

- ✓ Todas as entidades foram convertidas em tabelas?
- ✓ Todos os atributos foram convertidos em colunas?
- ✓ Todas as PKs foram identificadas e definidas?
- ✓ Relacionamentos 1:N implementados com FKs no lado "N"?
- ✓ Relacionamentos N:M implementados com tabelas associativas?
- ✓ Todos os tipos de dados foram definidos corretamente?
- ✓ Restrições (NOT NULL, UNIQUE, DEFAULT) foram aplicadas?
- ✓ Todas as FKs referenciam as PKs corretas?

Importância do Modelo Lógico

O **modelo lógico** é o **esquema de implementação** do banco de dados. Este mapeamento é o **último passo antes de escrever o código DDL** (CREATE TABLE) em SQL.

Benefícios de um modelo lógico bem construído:

- Garante **integridade** dos dados
- Elimina **redundância** desnecessária
- Facilita **manutenção** e evolução
- Melhora **performance** das consultas
- Previne **anomalias** de dados
- Documenta a **estrutura** do sistema

→ Próximo Passo: Código DDL (CREATE TABLE)

Na próxima aula, você aprenderá a transformar este modelo lógico em comandos SQL reais, criando as tabelas, definindo as chaves primárias e estrangeiras, e implementando todas as restrições no banco de dados.

Parabéns pela conclusão da tradução do conceito para a lógica do banco de dados!