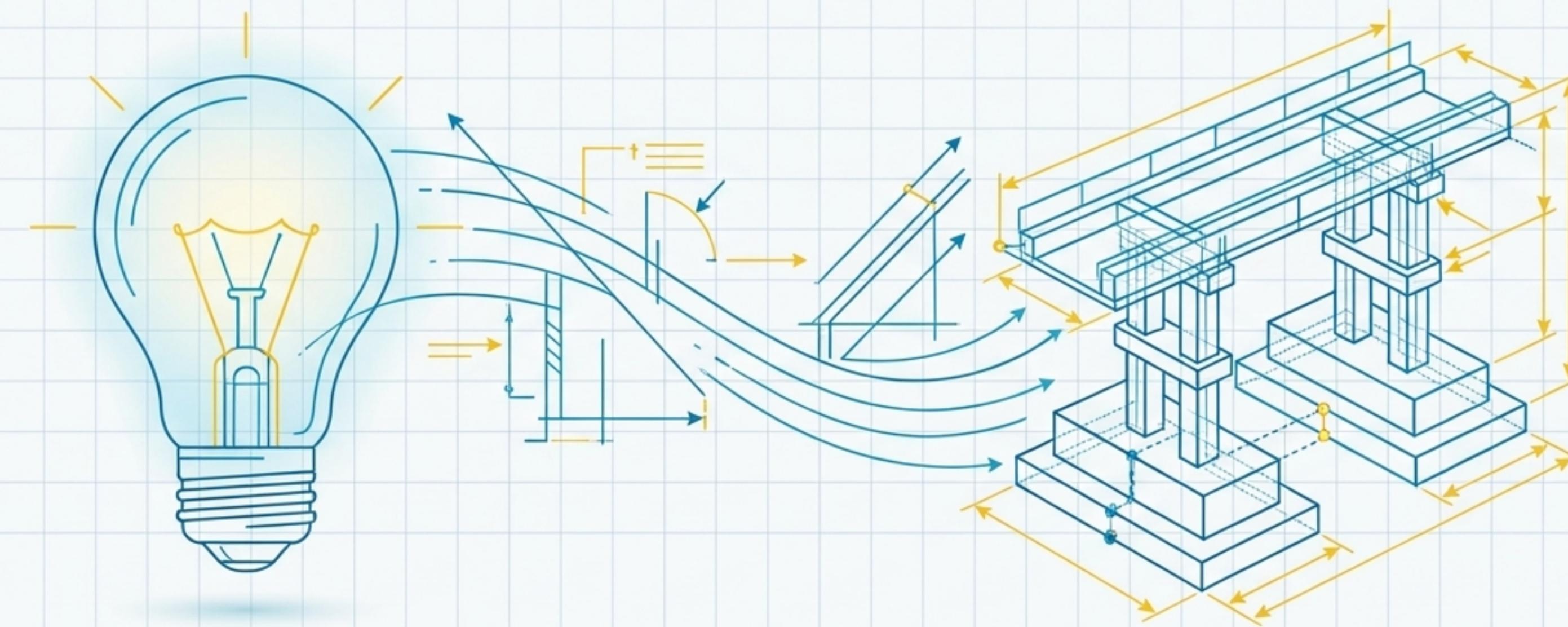


Do Esboço à Estrutura: Construindo Modelos de Dados Lógicos e Resilientes

Um guia para transformar modelos conceituais em esquemas relacionais robustos, normalizados e prontos para implementação.

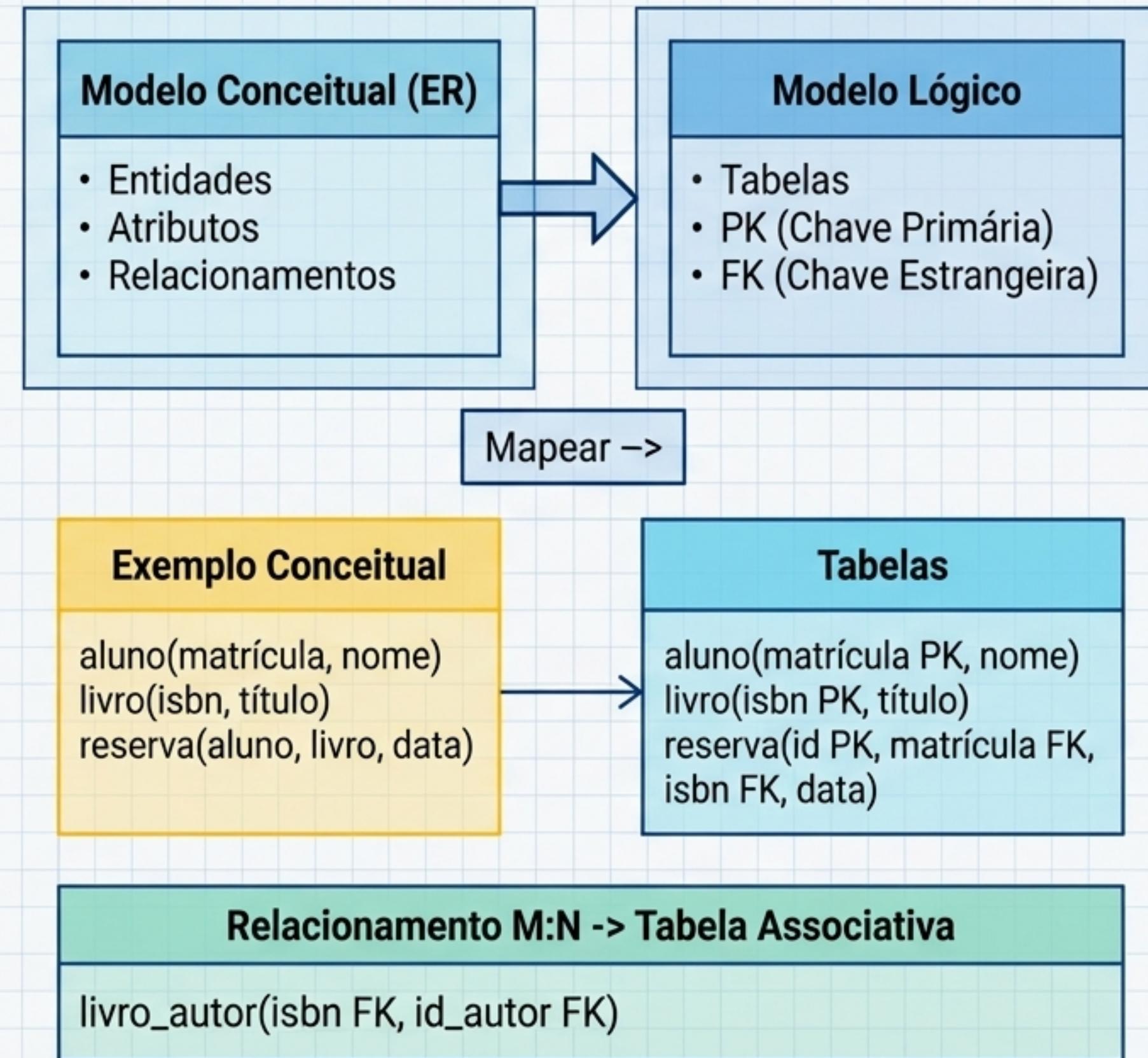


A Transição Essencial: Do Modelo Conceitual ao Lógico

A jornada de um banco de dados começa com uma visão (o Modelo Entidade-Relacionamento) e se materializa em um plano de construção detalhado (o Modelo Lógico Relacional). Esta etapa traduz entidades e relacionamentos em uma estrutura formal de tabelas, chaves e restrições.

Diretrizes Gerais:

- Definir chaves primárias (PK) claras.
- Garantir integridade referencial com chaves estrangeiras (FK).
- Refinar o esquema através da normalização (1FN–3FN).
- Adotar convenções de nomenclatura, como `snake_case` sem acentos.

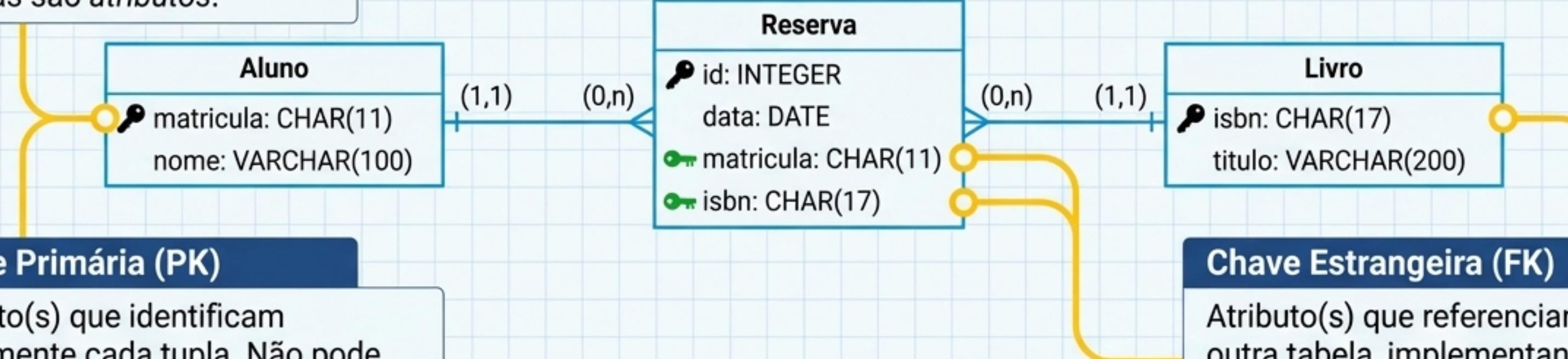


A Estrutura Fundamental: Tabelas, Chaves Primárias e Estrangeiras

No Modelo Lógico, cada componente tem uma função precisa. As tabelas armazenam os dados, e as chaves garantem que cada registro seja único e que os relacionamentos entre tabelas sejam consistentes.

Tabela (Relação)

Uma coleção de dados, onde as linhas são *tuplas* e as colunas são *atributos*.



Chave Primária (PK)

Atributo(s) que identificam unicamente cada tupla. Não pode haver repetição ou valores nulos.

Chave Estrangeira (FK)

Atributo(s) que referenciam a PK de outra tabela, implementando relacionamentos e garantindo a integridade referencial.

Decodificando o Diagrama Lógico: Símbolos e Cardinalidades

Cada elemento no diagrama lógico comunica uma regra precisa sobre a estrutura dos dados.

Símbolos de Chave

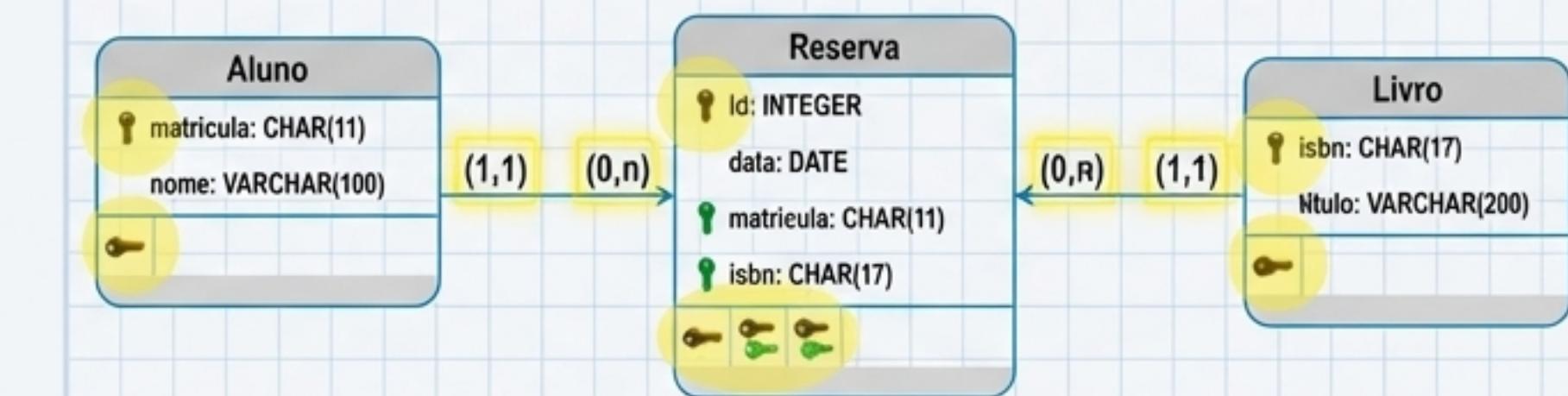
- **PK (Chave Preta)**: Identificador único da tabela (ex: `aluno.matricula`, `livro.isbn`).
- **FK (Chave Verde)**: Vínculo com outra tabela (ex: `reserva.matricula` referencia `aluno.matricula`).

Tipos de Dados

- **CHAR(n)**: Tamanho fixo. Ideal para dados padronizados como `matricula` CHAR(11) ou `isbn` CHAR(17).
- **VARCHAR(n)**: Tamanho variável. Usado para textos como `nome` ou `titulo`.

Direção e Cardinalidades

- As setas apontam da FK para a PK.
- "**Aluno (1,1) – (0,n) Reserva**": Uma reserva pertence a **exatamente um** aluno; um aluno pode ter **zero ou mais** reservas.
- "**Livro (1,1) – (0,n) Reserva**": Uma reserva é para **exatamente um** livro; um livro pode ser associado a **zero ou mais** reservas.



O Desafio da Integridade: Identificando Falhas Estruturais

Um design de banco de dados que não passa por um processo de refinamento (normalização) sofre de redundância e anomalias de dados (problemas em inserção, atualização e exclusão).

PASSO 1: Não Normalizada

Problema 1FN: telefones, autores multivalorados

biblioteca_completa

<u>id_emp</u>	<u>data_emp</u>	<u>matricula</u>	<u>nome</u>	<u>telefones</u>	<u>cod_ex</u>	<u>isbn</u>	<u>titulo</u>	<u>autores</u>
1	2022-03-05	02001	Nome Aluno	99999-1111, 88888-2222	21	12338636338	A Caratia Alredotar	Jahara Julis
2	2022-05-03	02002	Alexandre Vaian	99999-1111, 88888-2222	22	12345627798	São Ropado Rogaris	Jahara Faria
3	2022-06-31	02003	Ivananda Horras	000000-2227	21	12326270201	Polarálos Oridos	Ivananda Horras

Exemplo Base: A Tabela Não Normalizada

biblioteca_completa(id_emp, data_emp, matricula_aluno, nome_aluno, telefones, cod_exemplar, isbn, titulo, autores)

Problemas Estruturais Identificados

- Violacão da 1FN:** telefones e autores são atributos multivalorados (ex: "99999-1111, 88888-2222").
- Risco de Violacão da 2FN:** Atributos não-chave podem depender de apenas parte de uma chave composta.
- Violacão da 3FN:** Existem dependências transitivas (nome_aluno depende de matricula_aluno, não da chave do empréstimo).

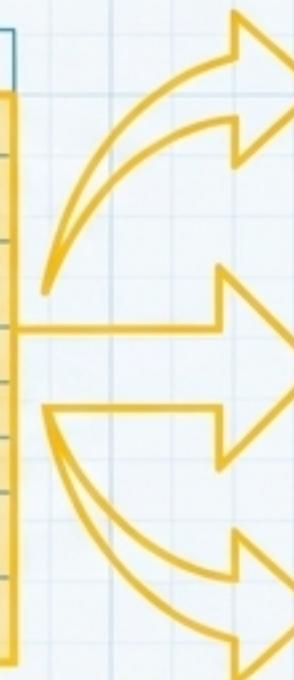
Reforço Estrutural 1: Primeira Forma Normal (1FN)

****Regra Fundamental**:** Todos os dados devem ser atômicos (indivisíveis). Cada célula da tabela deve conter um único valor.

****Problema em `biblioteca_completa`**:**

- "telefones": '99999-1111, 88888-2222"
- "autores": "Elmasri, Navathe"

biblioteca_completa								
id_emp	data_emp	matricula	nome	telefones	cod_ex	isbn	titulo	autores
1	2021-06-15	198001	Nome	99999-1111, 88888-2222	1	981839444	Elmasri, Navathe	Elmasri, Navathe
2	2021-06-13	198002	Annn Hanna	77777-3333	1	95329850	Elmasri, Navathe	R. Martin
3	2021-06-19	198003	Navathe	77777-3333	2	267145260	R. Martin	R. Martin
4	2021-06-13	198003	Navathe	77777-3333	3	707838770	R. Elanconlero	R. Martin
5	2021-05-13	198005	Mart frazit	77777-3333	4	153858370	Seh-Nzwner	R. Martin
6	2021-06-13	198007	Annn Hanna	77777-3333	5	962229850	Elmasri, Navathe	R. Martin
7	2021-06-13	198009	Annn Banna	77777-3333	6	797738777	R. Eichester	R. Martin



****Solução Aplicada (1FN)**:**

- "telefone_aluno(matricula_aluno FK, telefone)"
- "livro_autor(isbn FK, id_autor FK)"

emprestimo_v1					
id_emp	data_emp	matricula	nome	cod_ex	isbn
1	2021-05-17	190001	Nome	1	889990257
2	2021-05-17	180002	Navathe	2	983387870
3	2021-05-17	185003	Navathe	3	283302470

telefone_aluno	
matricula	telefone
190001	99999-1111
190002	88888-2222
187003	77777-3333

livro_autor_detalhado			
isbn	id_autor	titulo	nome_autor
981899144	1	Livro autor	Elmasri
267437360	2	Livro	Navathe
727367870	3	Livro	R. Martin

PASSO 1: Não Normalizada

PASSO 2: Após 1FN

****Resultado Parcial**:** Os atributos multivalorados foram separados, mas a tabela principal (`emprestimo_v1`) ainda contém outras dependências problemáticas.

Reforço Estrutural 2: Segunda Forma Normal (2FN)

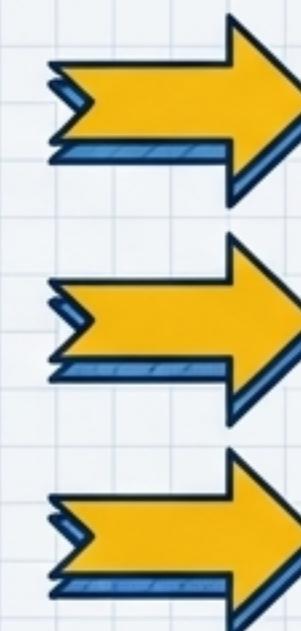
Regra Fundamental*: Nenhum atributo não-chave deve depender de apenas uma parte da chave primária composta.

Problema em `livro_autor_detalhado`:

- A tabela `livro_autor_detalhado(isbn, id_autor, titulo, nome_autor)` possui uma PK composta (`isbn`, `id_autor`).
- `titulo` depende apenas de `isbn`.
- `nome_autor` depende apenas de `id_autor`.

PASSO 2

isbn	autor	titulo	nome_autor
978-321	201	Database Systems	Elmasri
979-321	202	Database Systems	Navathe
978-013	203	Clean Code	R. Martin



Solução Aplicada (2FN):

- `livro(isbn PK, titulo)
- `autor(id_autor PK, nome)
- `livro_autor(isbn FK, id_autor FK) – Agora contém apenas chaves, resolvendo o relacionamento M:N.

livro

isbn	titulo
978-321	Database Systems
979-013	Clean Code

autor

id_autor	nome
201	Elmasri
202	Navathe
203	R. Martin

livro_autor

isbn	id_autor
978-321	201
979-321	202
978-013	203

Reforço Estrutural 3: Terceira Forma Normal (3FN)

****Regra Fundamental**:** Nenhum atributo não-chave deve depender de outro atributo não-chave.

PROBLEMA em `emprestimo_v1`:

Transitive Dependency:
nome depends on matricula

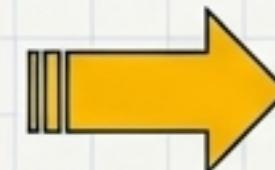
emprestimo_v1

id_emp (PK)	data_emp	matricula	nome	cod_ex	isbn
1	'2024-01-15'	1	'João'	501	'978-321'
2	'2024-01-16'	2	'Maria'	502	'978-013'

Transitive Dependency:
isbn depends on cod_ex

A PK é `id_emp`.

- `nome_aluno` depende de `matricula_aluno`, que por sua vez depende de `id_emp`.
- `titulo` (e `isbn`) depende de `cod_exemplar`, que por sua vez depende de `id_emp`.
- Dependências transitivas: $\text{id_emp} \rightarrow \text{matricula_aluno} \rightarrow \text{nome_aluno}$ e $\text{id_emp} \rightarrow \text{cod_exemplar} \rightarrow \text{isbn}$.



SOLUÇÃO APLICADA (3FN):

- **aluno**(matricula PK, nome)
- **exemplar**(cod_exemplar PK, isbn FK)
- **emprestimo**(...) – Agora referencia matricula_aluno e cod_exemplar via Fks.

aluno

matricula (PK)	nome
1	'João'
2	'Maria'

exemplar

cod_ex (PK)	isbn (FK)
501	'978-321'
502	'978-013'

emprestimo

id_emp (PK)	data_emp	matricula (FK)	cod_ex (FK)
1	'2024-01-15'	1	501
2	'2024-01-16'	2	502

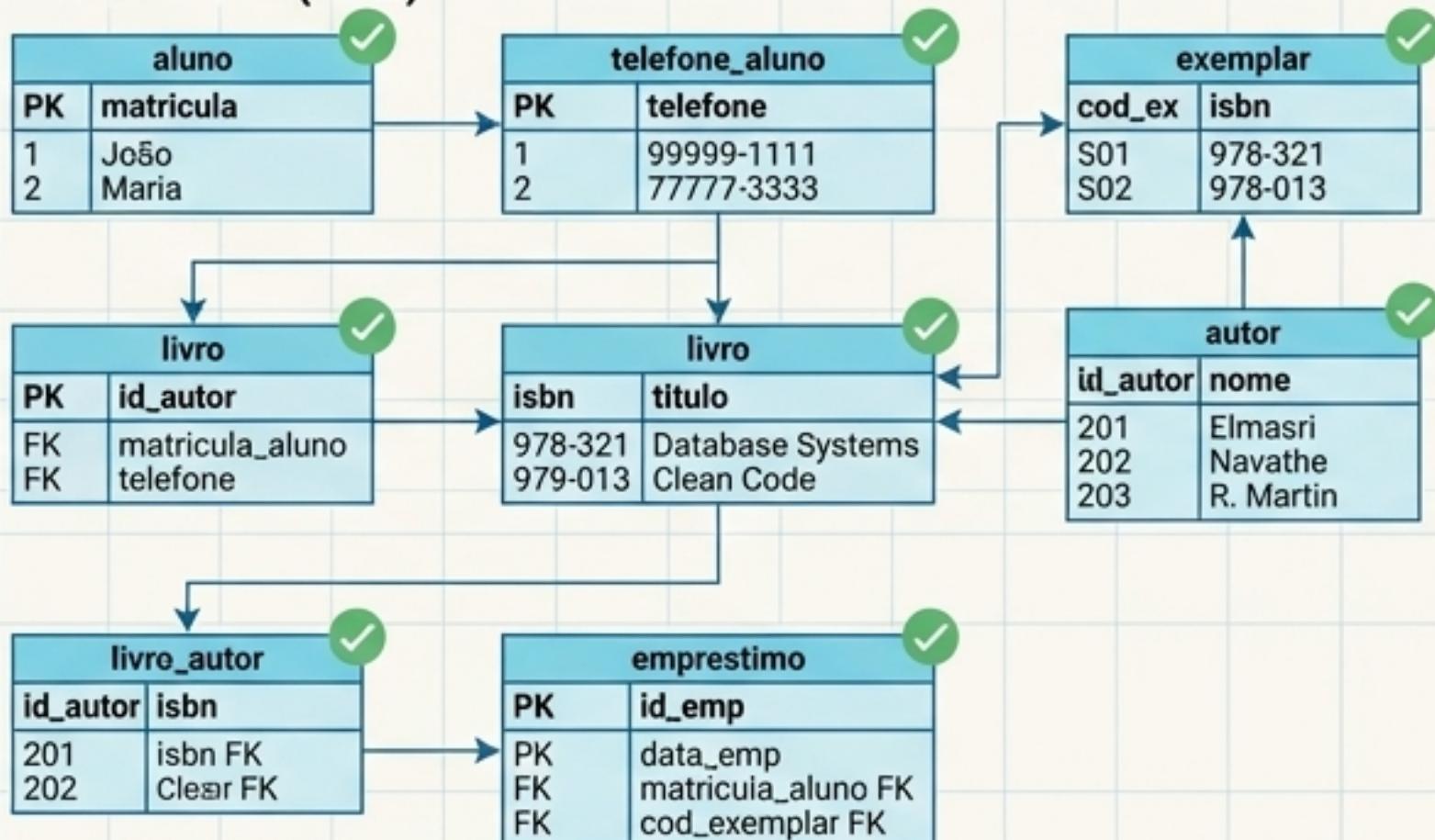
3FN aplicada: Dependência transitiva eliminada → aluno e exemplar separados



O Blueprint Otimizado: O Esquema em 3FN

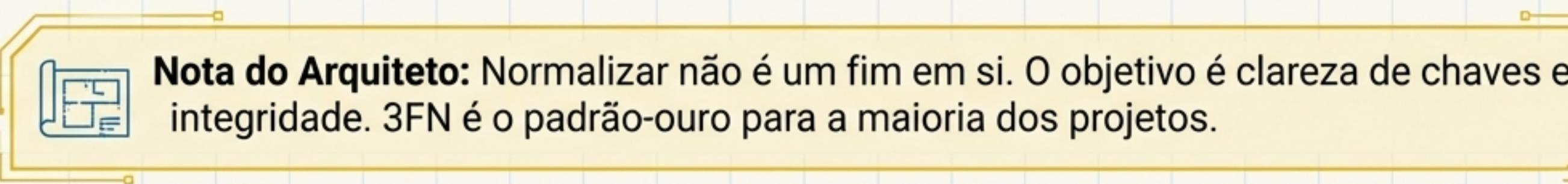
Após a aplicação das três formas normais, o resultado é um esquema onde cada atributo está em seu lugar correto, dependendo “somente da chave, da chave inteira e de nada mais que a chave”. A redundância é minimizada e a integridade dos dados é maximizada.

PASSO 4: 3FN (Final)



****Resultado Final – Todas as Tabelas em 3FN*:**

- aluno(matricula PK, nome)
- telefone_aluno(matricula_aluno FK, telefone)
- livro(isbn PK, titulo)
- autor(id_autor PK, nome)
- livro_autor(isbn FK, id_autor FK)
- livro_exemplar(isbn FK, id_exemplar FK)
- exemplar(cod_exemplar PK, isbn FK)
- emprestimo(id_emp PK, data_emp, matricula_aluno FK, cod_exemplar FK)



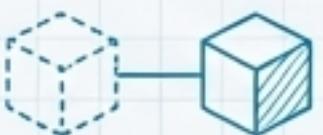
O Algoritmo de Mapeamento: Regras para a Construção

Siga estas regras sistematicamente para converter qualquer Modelo ER em um Modelo Lógico Relacional preciso.

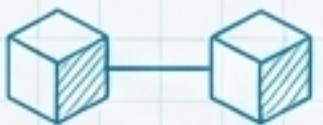
Regras de Mapeamento (ER → Lógico):



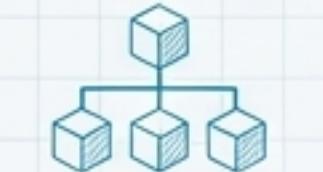
- 1. Entidade Regular:** Vira uma tabela com a PK definida.



- 2. Entidade Fraca:** Vira uma tabela com a PK da entidade proprietária como FK (formando uma PK composta).



- 3. Relacionamento 1:1:** A FK é adicionada em um dos lados da relação.



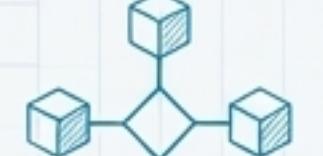
- 4. Relacionamento 1:N:** A PK do lado '1' é adicionada como FK no lado 'N'.



- 5. Relacionamento N:M:** Cria-se uma nova tabela associativa com as PKs de ambas as entidades como FKs (formando uma PK composta).



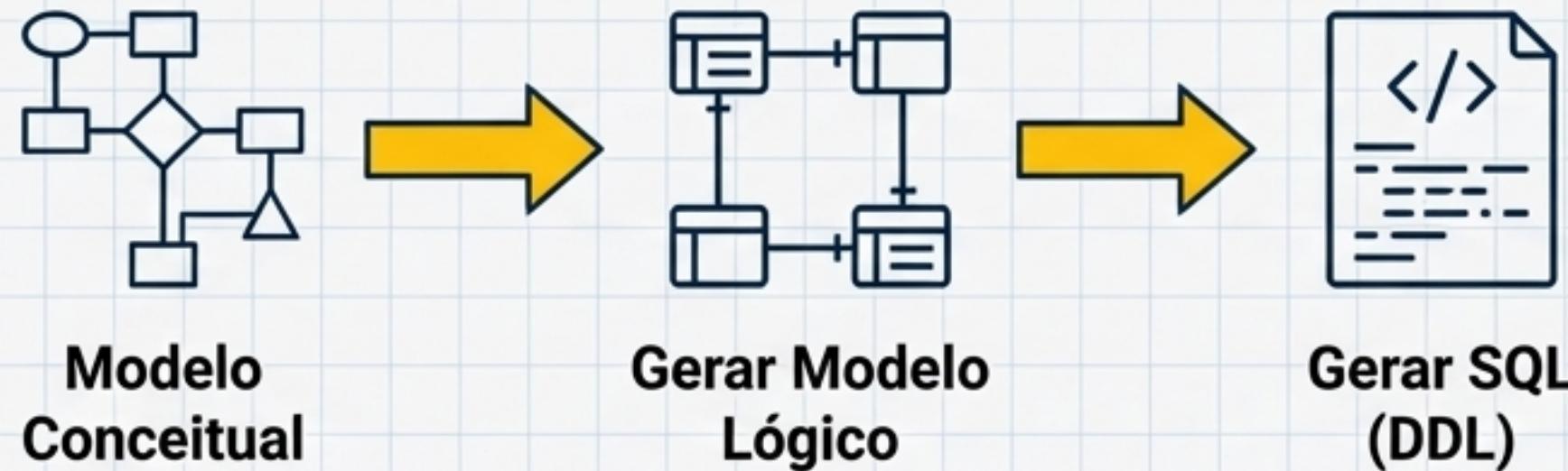
- 6. Atributo Multivalorado:** Cria-se uma nova tabela para o atributo, com uma FK para a entidade original.



- 7. Relacionamento N-ário:** Cria-se uma tabela com FKs para todas as entidades participantes.

Do Diagrama ao Código: Gerando o SQL DDL

Ferramentas CASE como o brModelo (Web) automatizam a conversão do modelo lógico em um script SQL (Data Definition Language), pronto para ser executado no banco de dados.



Fluxo de Trabalho no brModelo:

- *Modelo Conceitual:** Criar o diagrama ER.
- *Gerar Modelo Lógico:** Converter automaticamente para o modelo relacional.
- *Gerar SQL (DDL):** Exportar o script final, revisando tipos de dados e restrições.

**Exemplo de Código Gerado (DDL):

```
CREATE TABLE aluno (
    matricula INTEGER NOT NULL,
    nome VARCHAR(100) NOT NULL,
    PRIMARY KEY (matricula)
);
CREATE TABLE livro (
    isbn INTEGER NOT NULL,
    titulo VARCHAR(150) NOT NULL,
    PRIMARY KEY (isbn)
```

```
CREATE TABLE reserva (
    id_reserva INTEGER NOT NULL,
    aluno INTEGER NOT NULL,
    isbn INTEGER NOT NULL,
    data_reserva DATE NOT NULL,
    FOREIGN KEY (matricula) REFERENCES aluno(matricula)
    -- ...
);
```



Estudo de Caso: O Blueprint Completo da Biblioteca

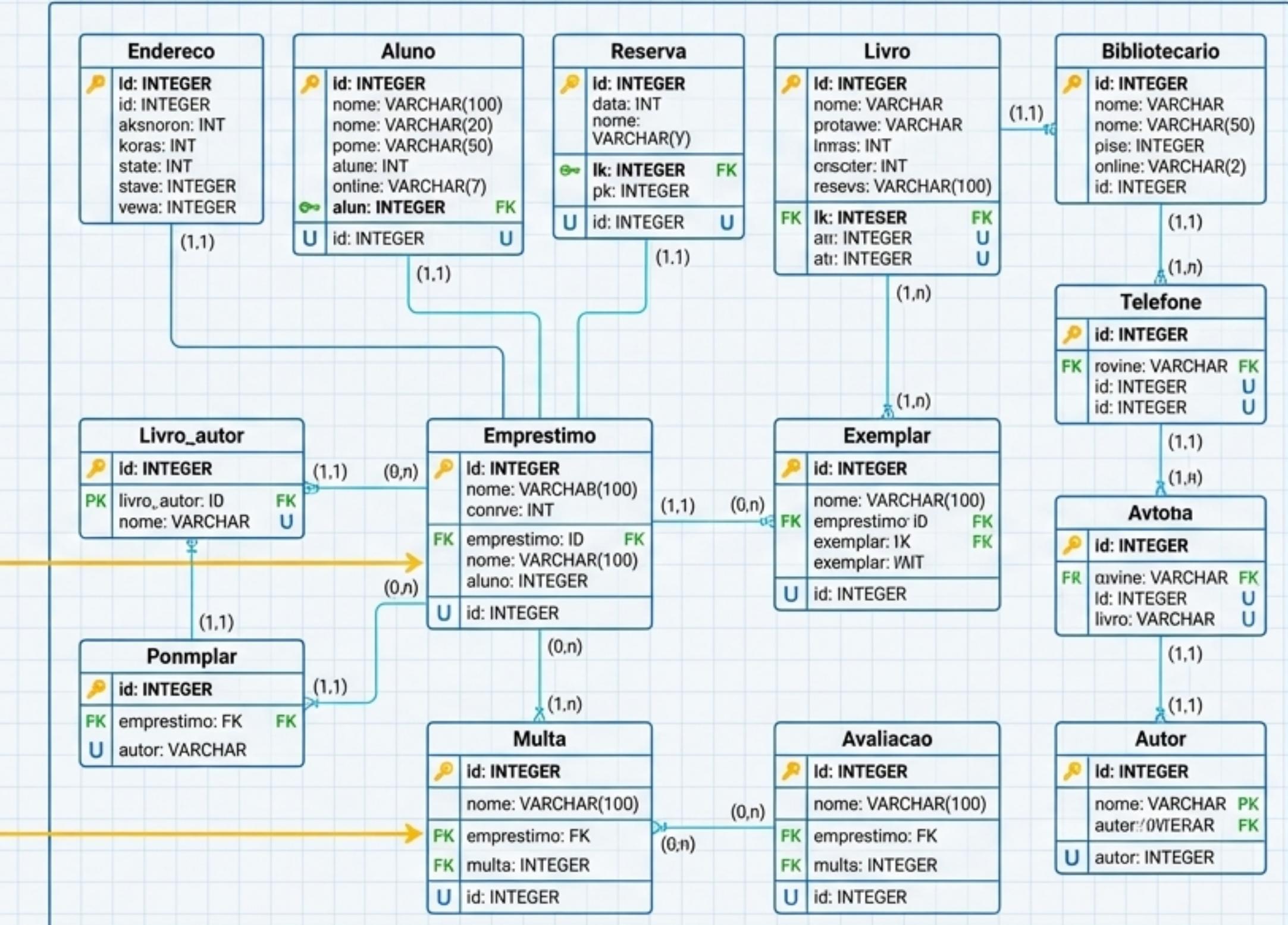
Aplicando todas as regras de mapeamento e normalização, o modelo conceitual da biblioteca se transforma neste esquema lógico detalhado. Observe a resolução de relacionamentos N:M ('livro_autor'), atributos multivvalorados ('telefone') e a definição precisa de todas as chaves e tipos de dados.

Notas de Projeto

***PK Composta vs. ID Próprio**:** `livro_autor` usa PK composta, pois a combinação é única. `emprestimo` precisa de um ID próprio, pois o mesmo aluno pode pegar o mesmo livro várias vezes.

Notas de Projeto

***Relacionamentos 1:1**:** Em `multa` e `avaliacao`, a PK é também uma FK para `emprestimo`, garantindo que um empréstimo tenha no máximo uma multa/avaliação.



Princípios Essenciais da Arquitetura de Dados

Um modelo lógico bem-sucedido não é apenas uma coleção de tabelas, mas uma **estrutura projetada com precisão** para garantir integridade, eficiência e escalabilidade.



Modelo Lógico: A ponte formal entre o conceito e a implementação, definindo tabelas, colunas tipadas e chaves.



Normalização (1FN, 2FN, 3FN): O processo de engenharia para eliminar redundância e anomalias, garantindo que os dados sejam atômicos e dependam exclusivamente de suas chaves primárias.



***Chaves (PK/FK):** A espinha dorsal da integridade referencial, garantindo unicidade e relacionamentos válidos.



Tabela Associativa: A solução padrão e elegante para resolver relacionamentos de muitos-para-muitos (N:M).

Próximos Passos: No próximo capítulo, implementaremos este modelo lógico em SQL, criando tabelas com tipos de dados, restrições e políticas de integridade.