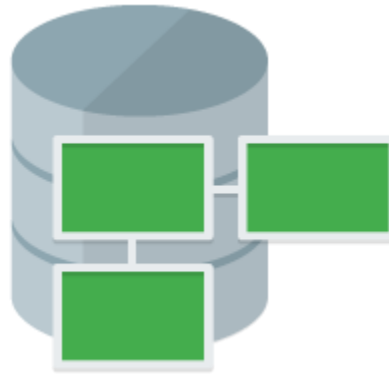




CWI SOFTWARE

módulo 2 | banco de dados



Modelagem de banco de dados

André Luís Nunes

Diagrama Entidade Relacionamento (DER)

- O diagrama ER é equivalente a uma planta de uma construção.



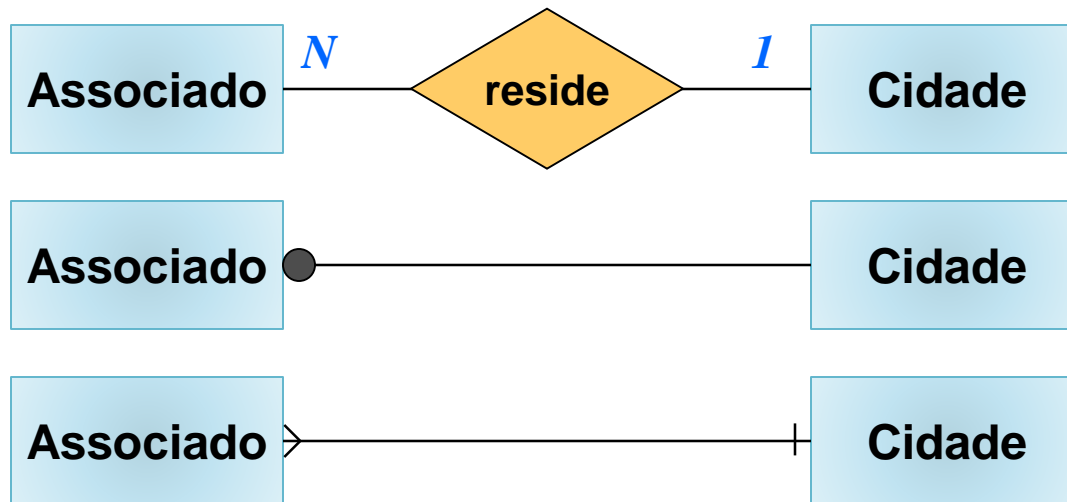
- Assim como na construção civil, nem sempre a planta é construída antes da casa, e nem sempre o modelo está atualizado.

Diagrama Entidade Relacionamento (DER)

- Itens de um diagrama Entidade-Relacionamento (ER):
 - ENTIDADE: representa normalmente uma tabela no banco de dados.
 - ATRIBUTOS: representa as colunas da tabela.
 - RELACIONAMENTO: identifica a relação entre as entidades.
 - CARDINALIDADE: indica a participação e obrigatoriedade do relacionamento.

Modelagem de dados (DER)

- Cardinalidade máxima



Notação N pra 1.

Notação tradicional.

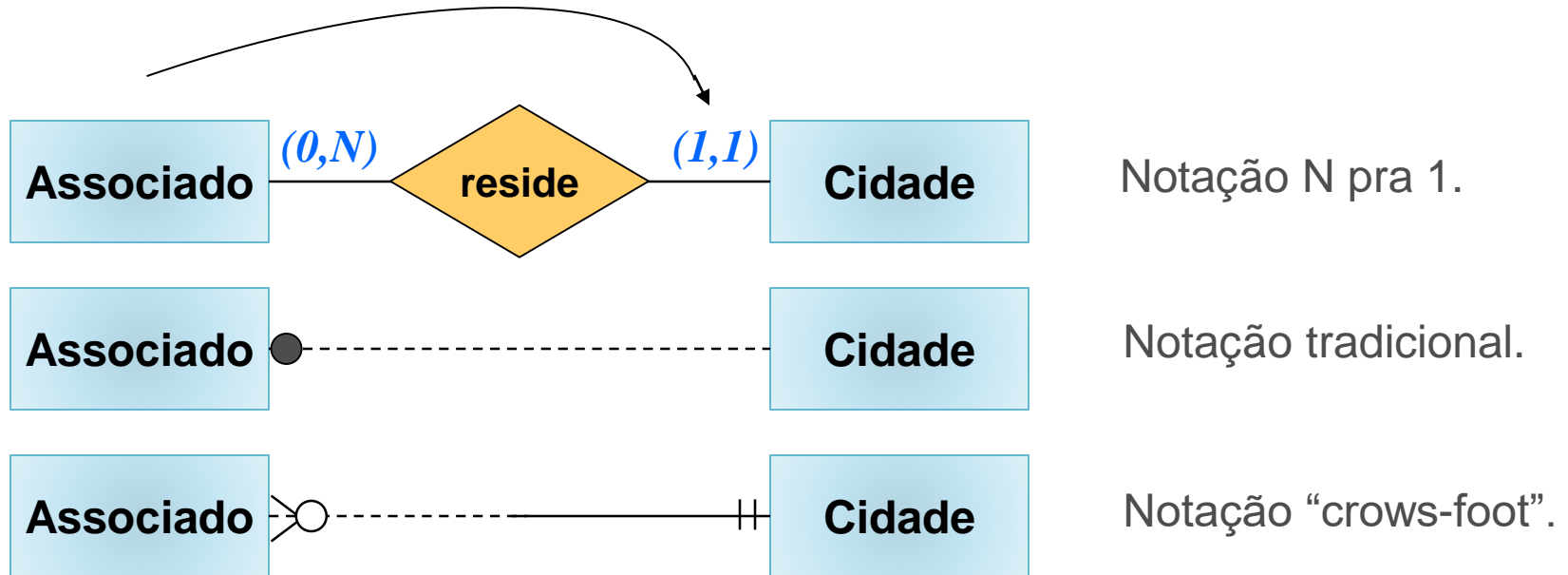
Notação “crows-foot”.

Um associado reside no máximo em 1 cidade.

Uma cidade tem até N associados residindo nela.

Modelagem de dados (DER)

- Cardinalidade mínima

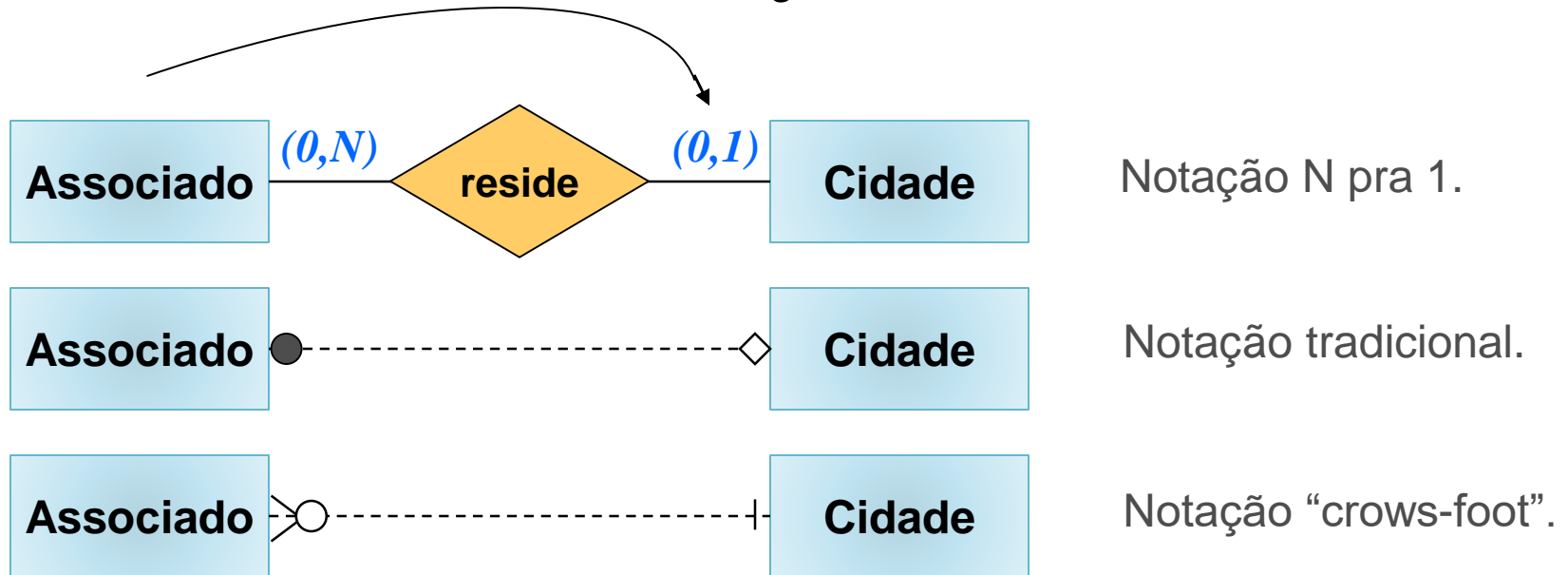


Um associado **obrigatoriamente reside** no máximo em 1 cidade.

Uma cidade **pode ter** até N associados residindo nela.

Modelagem de dados (DER)

- Cardinalidade mínima, alterando obrigatoriedade:

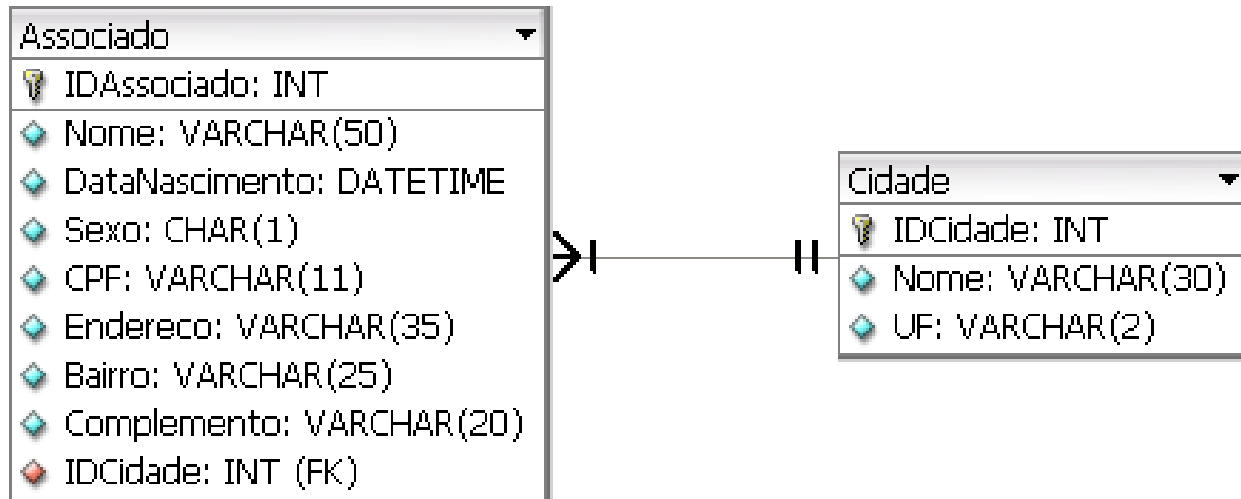


Um associado **opcionalmente reside** no máximo em 1 cidade.

Uma cidade **pode ter** até N associados residindo nela.

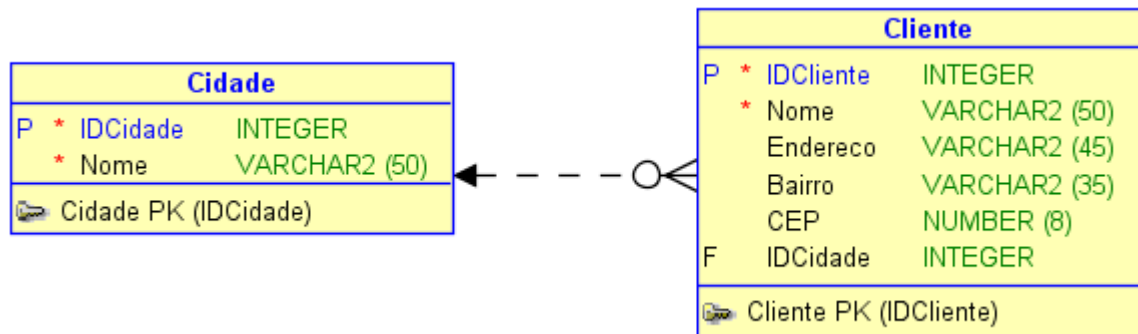
Modelagem de dados (DER)

- Exemplo da relação entre um Associado e Cidade:



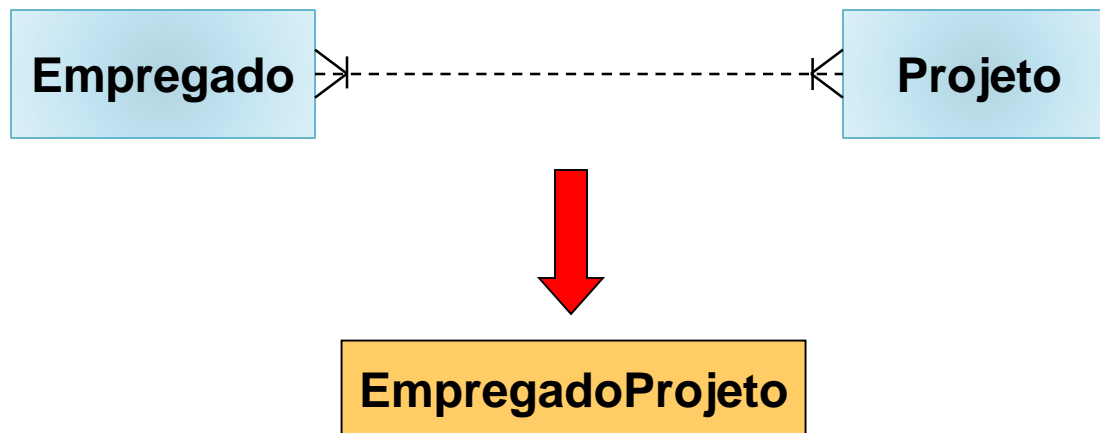
Modelagem de dados (DER)

- Exemplo da relação entre Cidade e Cliente:



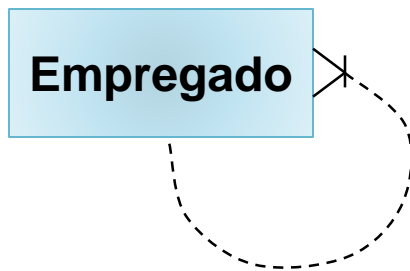
Modelagem de dados (DER)

- Exemplo de um diagrama, utilizando notação "Crows foot", onde o relacionamento irá gerar uma nova tabela.



Modelagem de dados (DER)

- Exemplo de uma tabela com auto-relacionamento:



A tabela terá uma coluna opcional, que fará referência para a própria tabela, porém outra coluna e conseqüentemente outro registro.

Normalização

- A normalização tem a função de analisar tabelas e organizá-las de forma que a sua estrutura seja simples, relacional e estável.
- Permitindo que seu gerenciamento seja simples, seguro e eficiente.
- O objetivo é garantir a integridade, evitar perda de informação e redundância não controlada.

Banco de dados | relacional (SQL)

Relação de conjuntos de dados

Cliente

Cidade

Banco de dados | relacional (SQL)

Relação de conjuntos de dados

Cliente

IDCliente	Nome
1	Pedro
2	Maria
3	Julia
10	Carlos
15	Antônio

Cidade

IDCidade	Nome	UF
1	Porto Alegre	RS
2	São Paulo	SP
3	Sapucaia do Sul	RS

Banco de dados | relacional (SQL)

Relação de conjuntos de dados

Cliente

IDCliente	Nome
1	Pedro
2	Maria
3	Julia
10	Carlos
15	Antônio

Chave primária (PK)

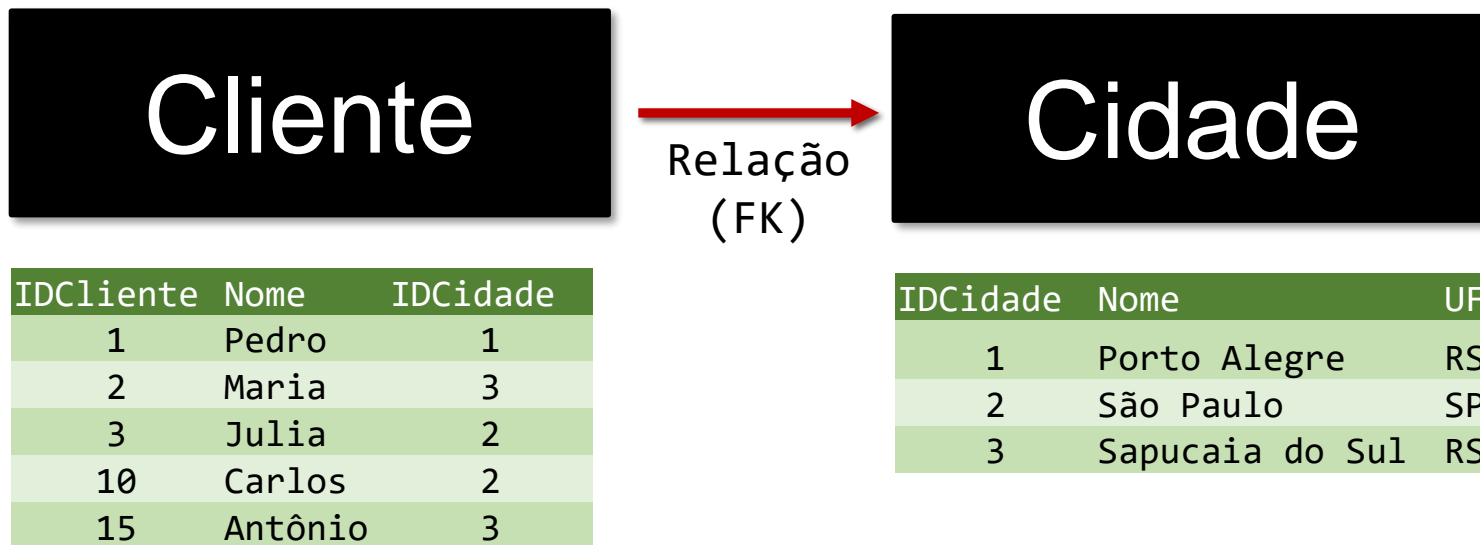
Cidade

IDCidade	Nome	UF
1	Porto Alegre	RS
2	São Paulo	SP
3	Sapucaia do Sul	RS

Chave primária (PK)

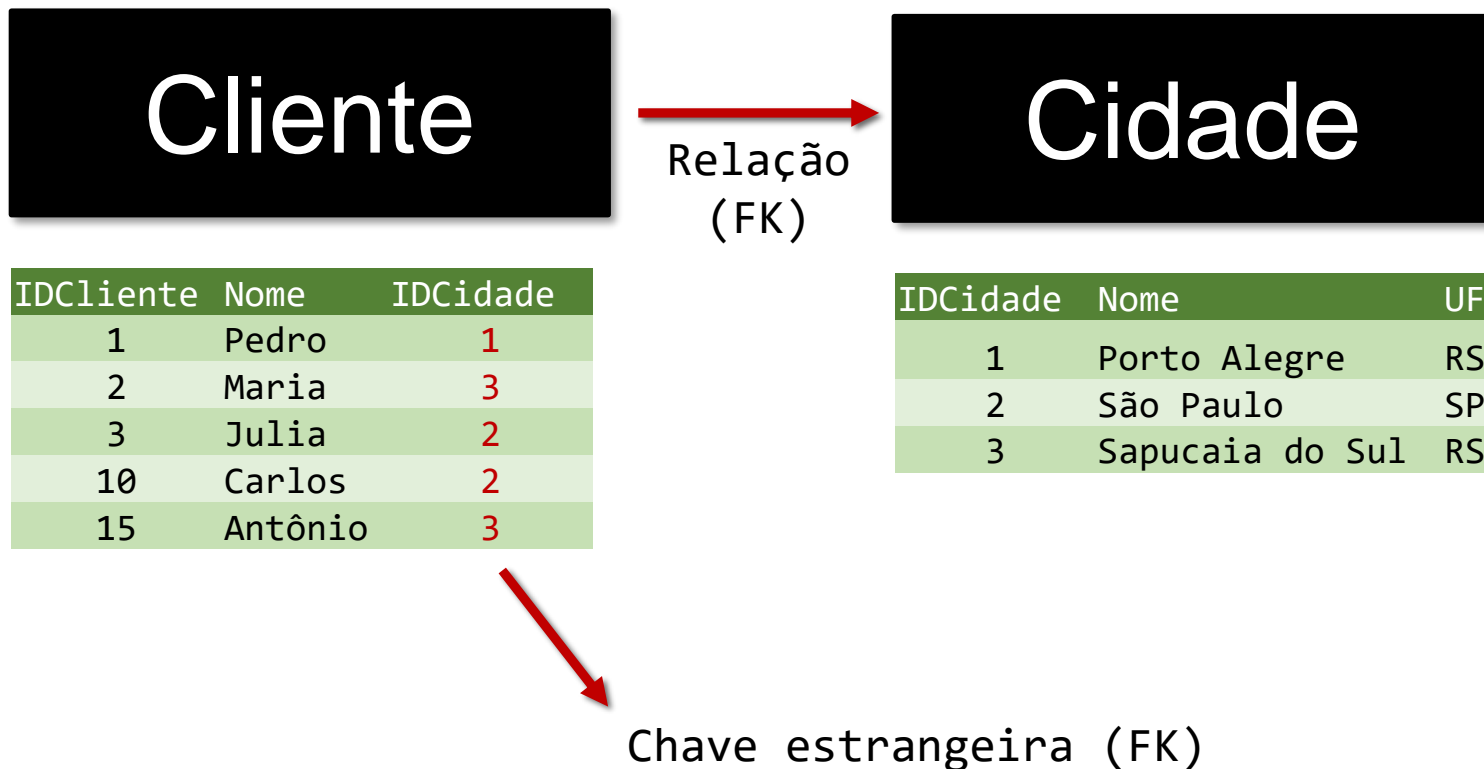
Banco de dados | relacional (SQL)

Relação de conjuntos de dados



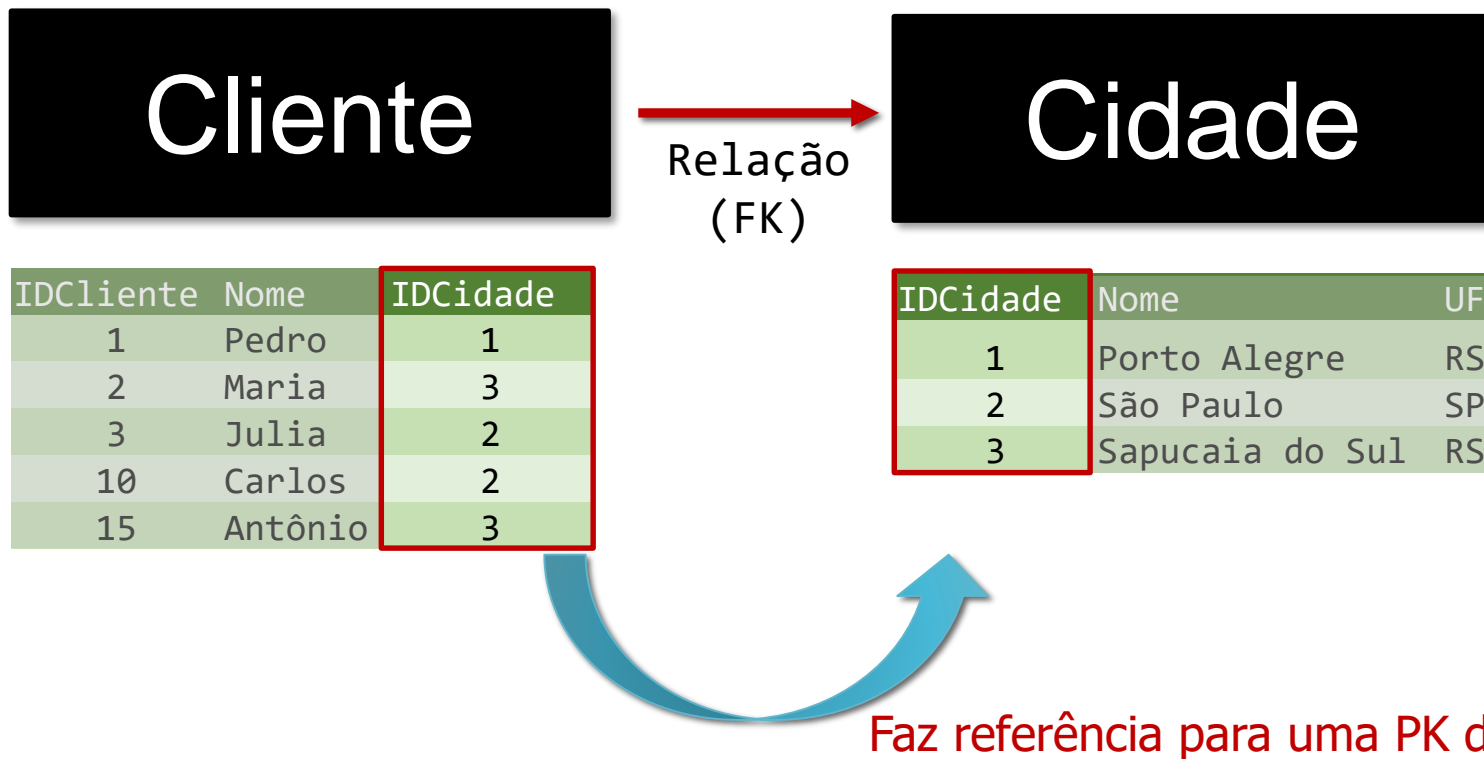
Banco de dados | relacional (SQL)

Relação de conjuntos de dados



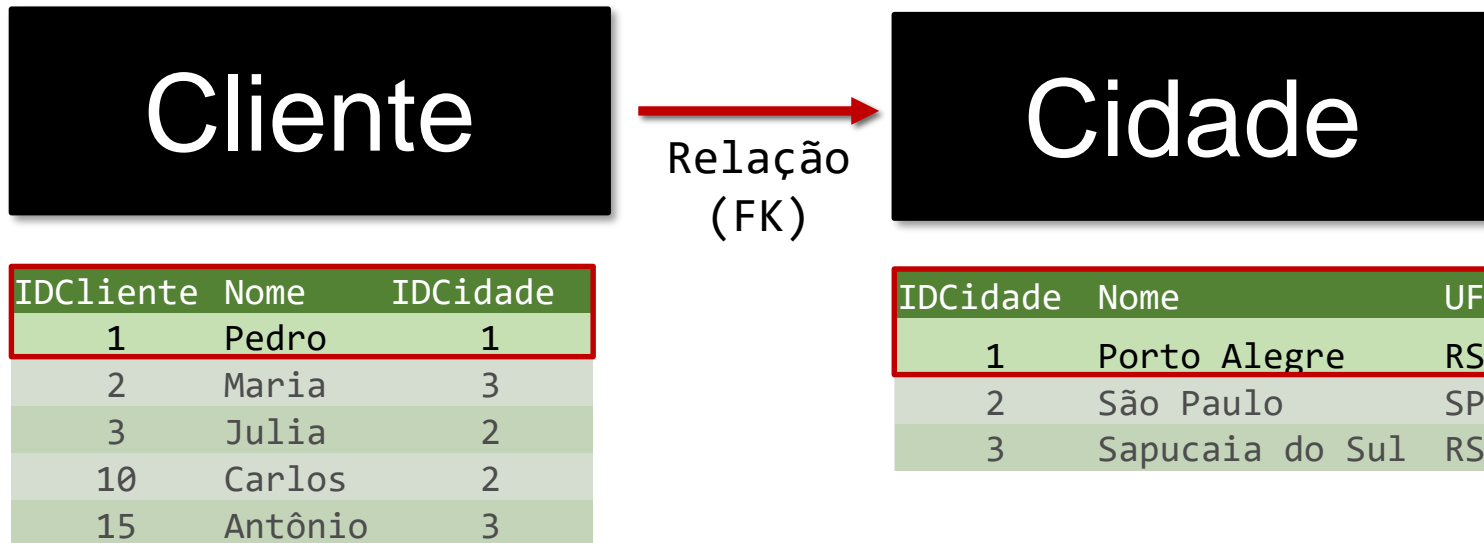
Banco de dados | relacional (SQL)

Relação de conjuntos de dados



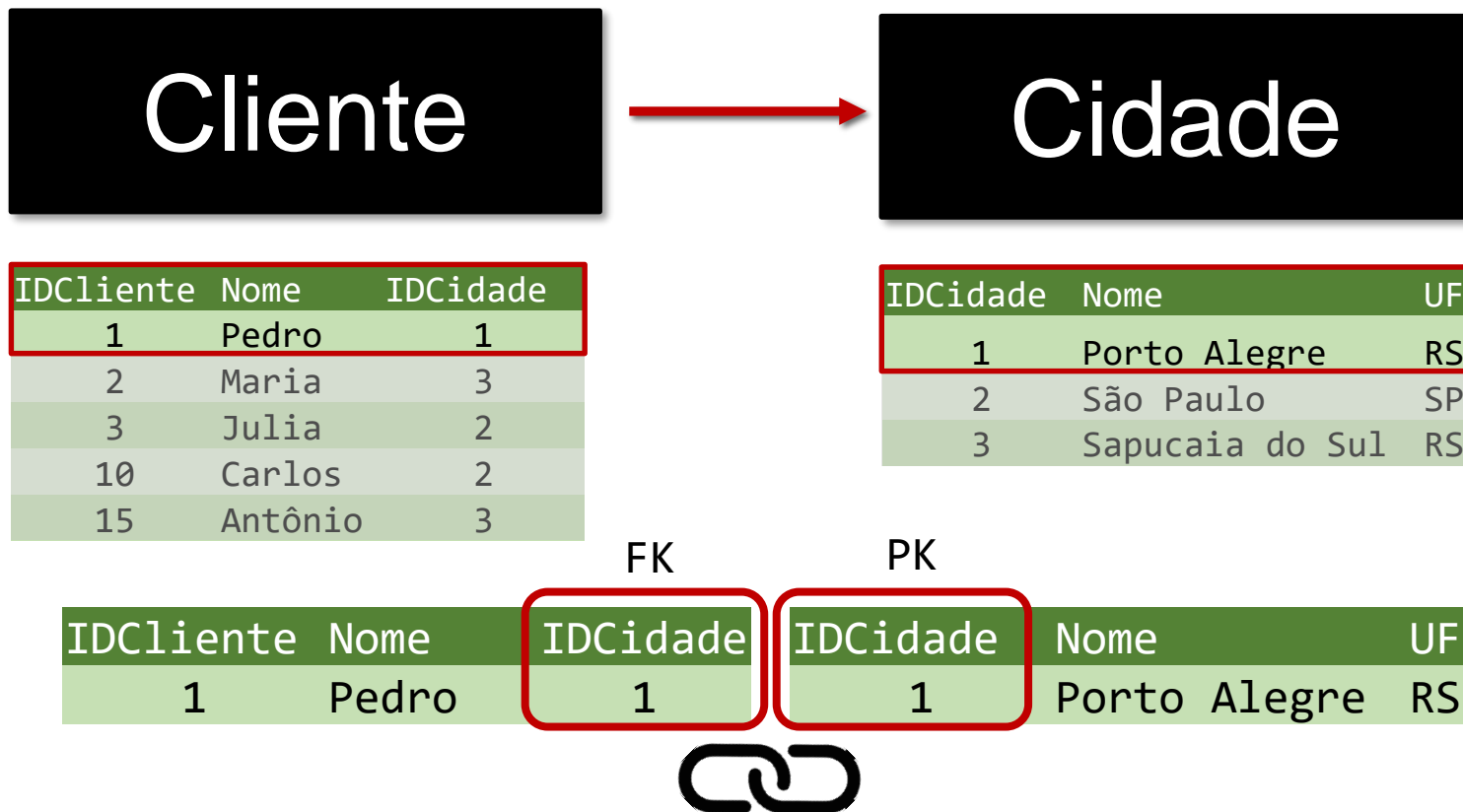
Banco de dados | relacional (SQL)

Relação de conjuntos de dados



Banco de dados | relacional (SQL)

Relação de conjuntos de dados

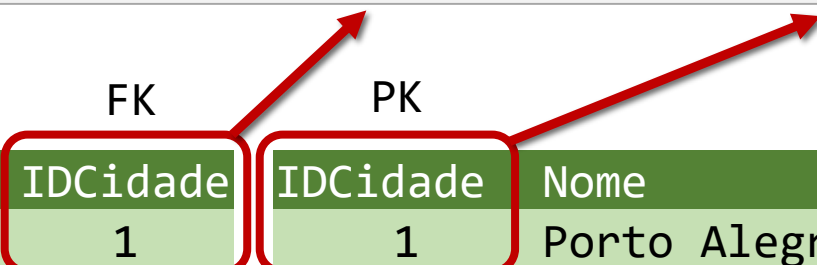


Banco de dados | relacional (SQL)

Relacionando 2 tabelas (joins)

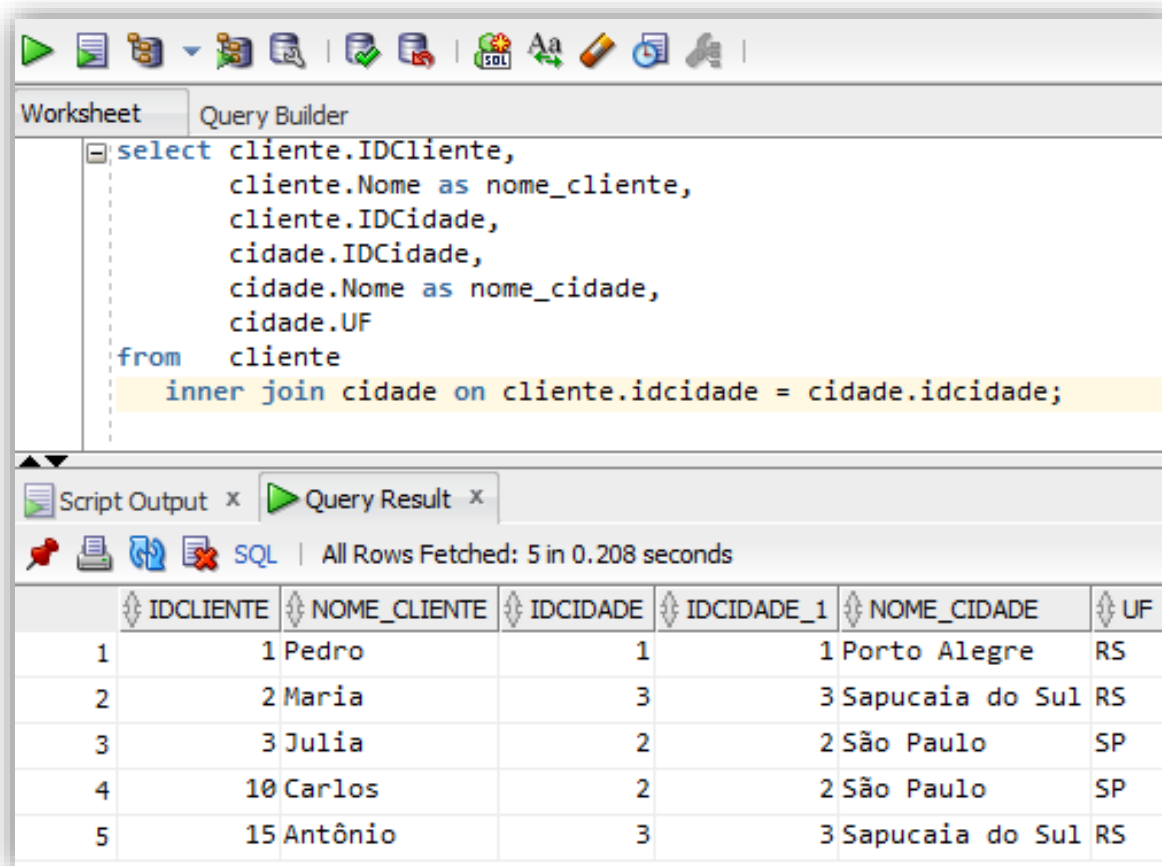
```
select cliente.IDCliente,  
       cliente.Nome as nome_cliente,  
       cliente.IDCidade,  
       cidade.IDCidade,  
       cidade.Nome as nome_cidade,  
       cidade.UF  
from   cliente  
       inner join cidade on cliente.idcidade = cidade.idcidade;
```

IDCliente	Nome	IDCidade	IDCidade	Nome	UF
1	Pedro	1	1	Porto Alegre	RS



Banco de dados | relacional (SQL)

Relação de conjuntos de dados



The screenshot displays a SQL Query Builder window. The top toolbar includes icons for running queries, saving, and editing. The main area shows a SQL query:

```
select cliente.IDCliente,
       cliente.Nome as nome_cliente,
       cidade.IDCidade,
       cidade.IDCidade,
       cidade.Nome as nome_cidade,
       cidade.UF
from   cliente
       inner join cidade on cliente.idcidade = cidade.idcidade;
```

Below the query editor, the 'Query Result' tab is active, showing the results of the query. The status bar indicates 'All Rows Fetched: 5 in 0.208 seconds'.

	IDCLIENTE	NOME_CLIENTE	IDCIDADE	IDCIDADE_1	NOME_CIDADE	UF
1	1	Pedro	1	1	Porto Alegre	RS
2	2	Maria	3	3	Sapucaia do Sul	RS
3	3	Julia	2	2	São Paulo	SP
4	10	Carlos	2	2	São Paulo	SP
5	15	Antônio	3	3	Sapucaia do Sul	RS

Banco de dados | relacional (SQL)

Join (junção)

Cliente		
IDCliente	Nome	IDCidade
1	Pedro	1
2	Maria	3
3	Julia	2
10	Carlos	2
15	Antônio	3

Cidade		
IDCidade	Nome	UF
1	Porto Alegre	RS
2	São Paulo	SP
3	Sapucaia do Sul	RS

Para cada registro da tabela "Cliente" é feita uma busca na tabela "Cidade", conforme o IDCidade.

Banco de dados | relacional (SQL)

Join (junção)

Cliente		
IDCliente	Nome	IDCidade
1	Pedro	1
2	Maria	3
3	Julia	2
10	Carlos	2
15	Antônio	3

Cidade		
IDCidade	Nome	UF
1	Porto Alegre	RS
2	São Paulo	SP
3	Sapucaia do Sul	RS

Para cada registro da tabela "Cliente" é feita uma busca na tabela "Cidade", conforme o IDCidade.

Banco de dados | relacional (SQL)

Join (junção)

Cliente		
IDCliente	Nome	IDCidade
1	Pedro	1
2	Maria	3
3	Julia	2
10	Carlos	2
15	Antônio	3

Cidade		
IDCidade	Nome	UF
1	Porto Alegre	RS
2	São Paulo	SP
3	Sapucaia do Sul	RS

Para cada registro da tabela "Cliente" é feita uma busca na tabela "Cidade", conforme o IDCidade.

Banco de dados | modelagem

- 1) Número de colunas por tabela;
- 2) Todo campo deve ser obrigatório;
- 3) Toda chave estrangeira deve ter um índice;
- 4) A aplicação deve trabalhar “offline”.
- 5) Evite chaves primárias compostas;

Modelagem | número de colunas

1) Evite criar tabela com muitas colunas.

Não existe um número mágico (sugestão 25);

Difícilmente todas as colunas serão projetadas sempre na aplicação.

O que fazer?

Crie uma tabela específica para determinado grupo de informações (relação 1-1).

Modelagem | obrigatoriedade

2) Toda coluna deve ser obrigatória, até que se prove o contrário.

Isso evita situações que precisam tratar valores nulos.

Quando um número muito grande de colunas é opcional, devido alguma de regra do negócio recomenda-se avaliar o modelo.

Modelagem | obrigatoriedade

2) Toda coluna deve ser obrigatória, até que se prove o contrário.

Exemplos: `obrigatoriedade.sql`

Tabelas: "Pessoa" e "Salario".

Modelagem | obrigatoriedade

2) Toda coluna deve ser obrigatória, até que se prove o contrário.

Consultado apenas o cadastro de pessoas:

```
select * from Pessoa;
```

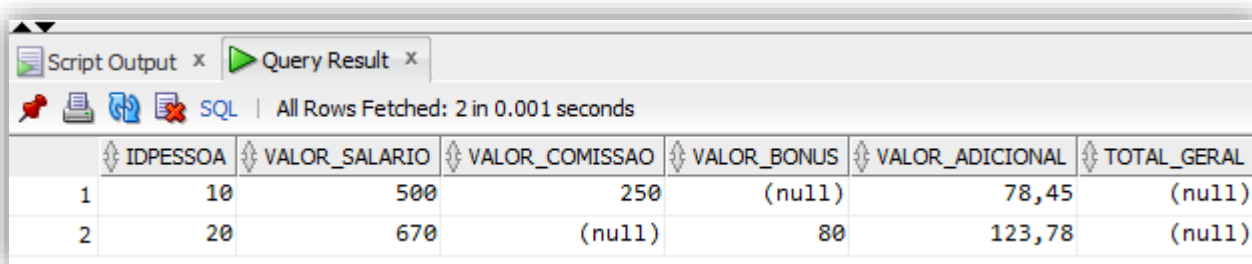
IDPESSOA	NOME	DATA_NASCIMENTO	CPF	RG	RAZAO_SOCIAL	CNPJ	INSCR_ESTAD
1	10 Pedro de Alcantra	22/04/1922	12345678911	1020304050	(null)	(null)	(null)
2	20 João Soares	23/05/1965	23456678911	9020304050	(null)	(null)	(null)
3	30 Armazem do Almeida	(null)	(null)	(null)	Comercio de Secos e Molhados do Almeida	78956789110001	isento
4	40 Bolicho do Salim	(null)	(null)	(null)	Cia de Varejo do Salim Ltda	78956789110001	isento

Observe no resultado que ocorre um “OR” em algumas colunas, algumas são preenchidas para pessoas físicas, outras para pessoas jurídicas.

Modelagem | obrigatoriedade

2) Toda coluna deve ser obrigatória, até que se prove o contrário.
Aplicando cálculos:

```
select IDPessoa,  
       Valor_Salario ,  
       Valor_Comissao,  
       Valor_Bonus ,  
       Valor_Adicional,  
       (Valor_Salario+Valor_Comissao+Valor_Bonus+Valor_Adicional) as Total_Geral  
from Salario;
```



	IDPESSOA	VALOR_SALARIO	VALOR_COMISSAO	VALOR_BONUS	VALOR_ADICIONAL	TOTAL_GERAL
1	10	500	250	(null)	78,45	(null)
2	20	670	(null)	80	123,78	(null)

Observe no resultado o total não é a soma correta das colunas, como esperado.

Modelagem | obrigatoriedade

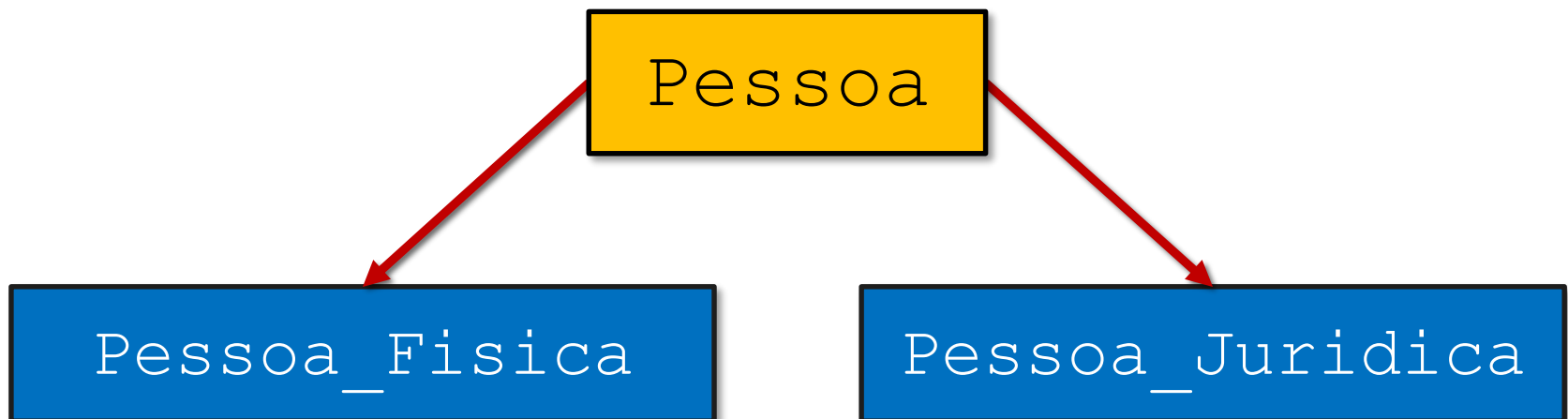
2) Toda coluna deve ser obrigatória, até que se prove o contrário.

Alternativas?

Modelagem | obrigatoriedade

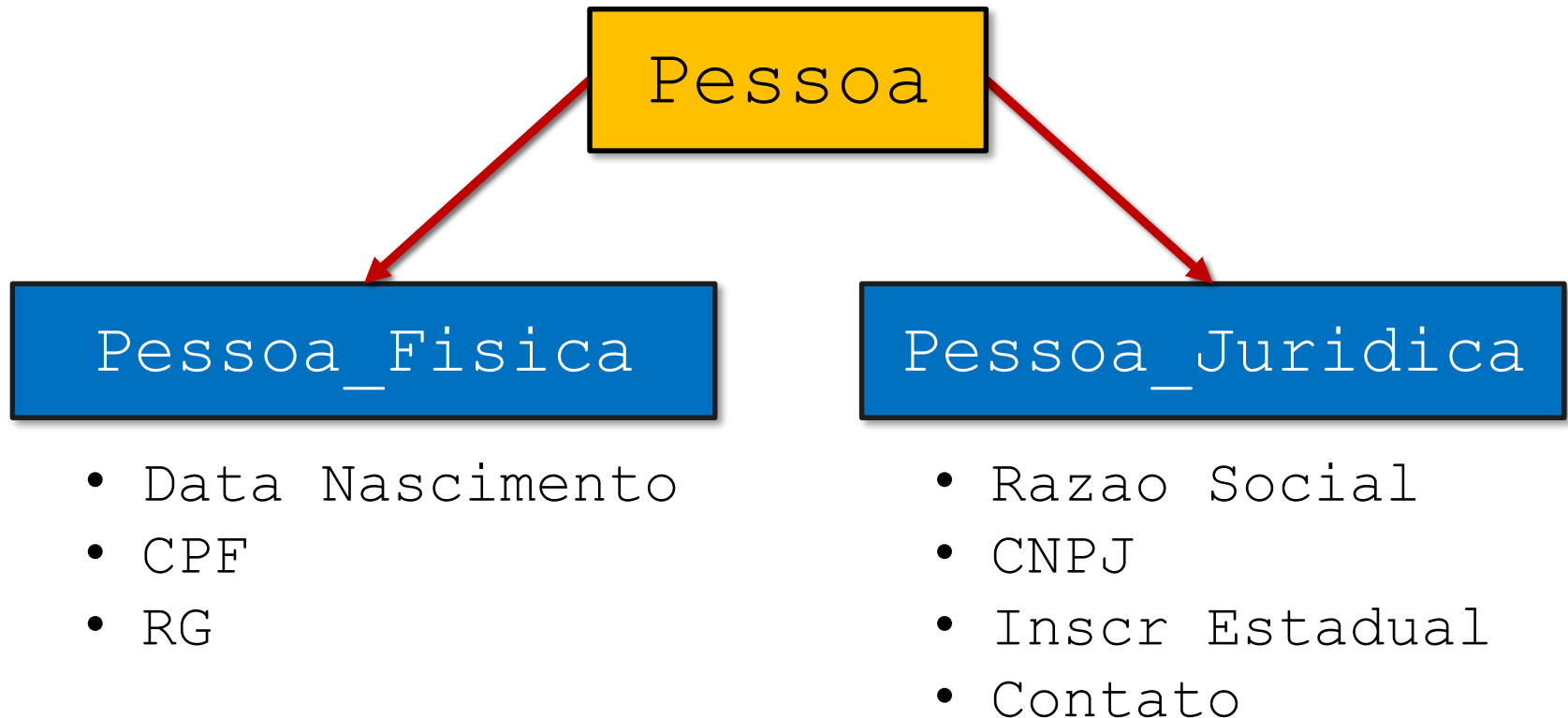
2) Toda coluna deve ser obrigatória, até que se prove o contrário.

Tabela Pessoa: criar outras 2 tabelas, especializando o modelo.



Modelagem | obrigatoriedade

2) Toda coluna deve ser obrigatória, até que se prove o contrário.



Modelagem | obrigatoriedade

2) Toda coluna deve ser obrigatória, até que se prove o contrário.

Tabela Salario: torne os campos obrigatórios!

```
create table Salario (  
    IDSalario          integer      not null,  
    IDPessoa           integer      not null,  
    Valor_Salario      number(7,2) not null,  
    Valor_Comissao     number(7,2) DEFAULT 0 not null,  
    Valor_Bonus        number(7,2) DEFAULT 0 not null,  
    Valor_Adicional    number(7,2) DEFAULT 0 not null,  
    constraint PK_Salario primary key (IDSalario),  
    constraint FK_Salario_Pessoa foreign key (IDPessoa)  
        references Pessoa (IDPessoa)  
);
```

Modelagem | índice para FK

3) Toda chave estrangeira deve ter índice!

Importante em 2 momentos:

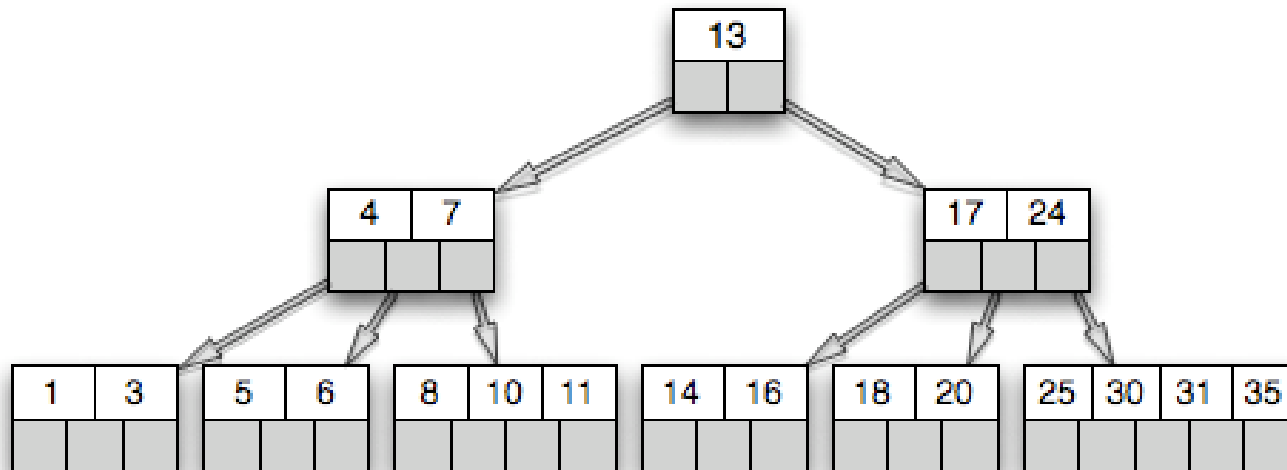
(1) consulta relacionando com a tabela pai;

(2) alterações (*update*) ou exclusões (*delete*) na tabela pai.

Modelagem | índice para FK

3) Toda chave estrangeira deve ter índice!

Índice ?



Modelagem | aplicação offline

4) Aplicação “offline”

Defina seu modelo de dados e aplicação de forma que seja independente de qualquer outra aplicação;

Se necessário, crie uma redundância controlada.

Modelagem | aplicação offline

4) Aplicação “offline”

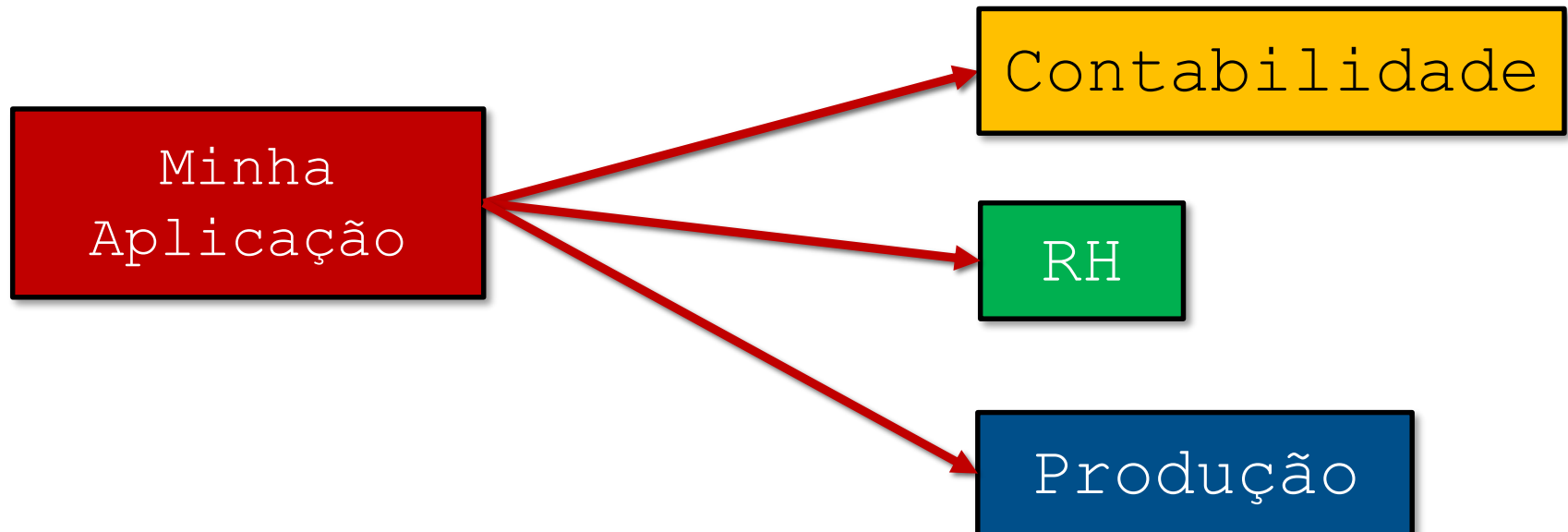
A aplicação deve continuar operacional, mesmo que outras aplicações à qual ela busca dados deixem de funcionar.

- ⇒ Crie serviços para buscar os dados;
- ⇒ Busque apenas os dados realmente necessários;
 - ⇒ A redundância controlada não é problema.
- ⇒ Garanta supervisão (visibilidade) sobre a execução desses serviços;

Modelagem | aplicação offline

4) Aplicação "offline"

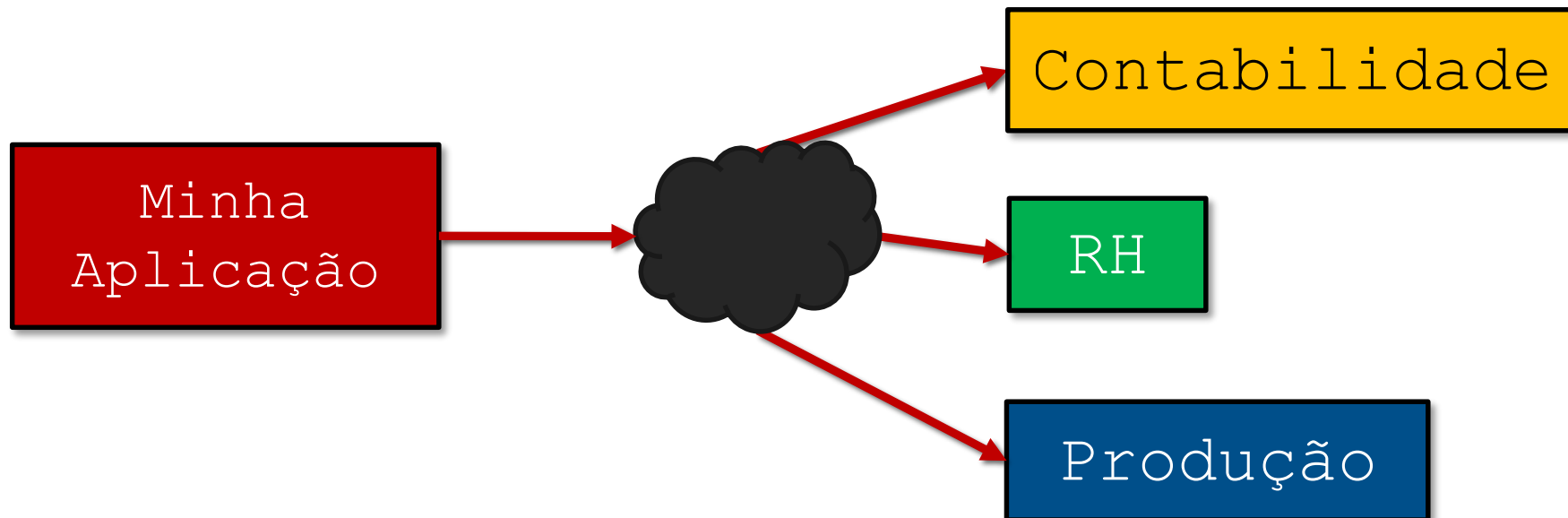
Pense em performance SEMPRE!



Modelagem | aplicação offline

4) Aplicação "offline"

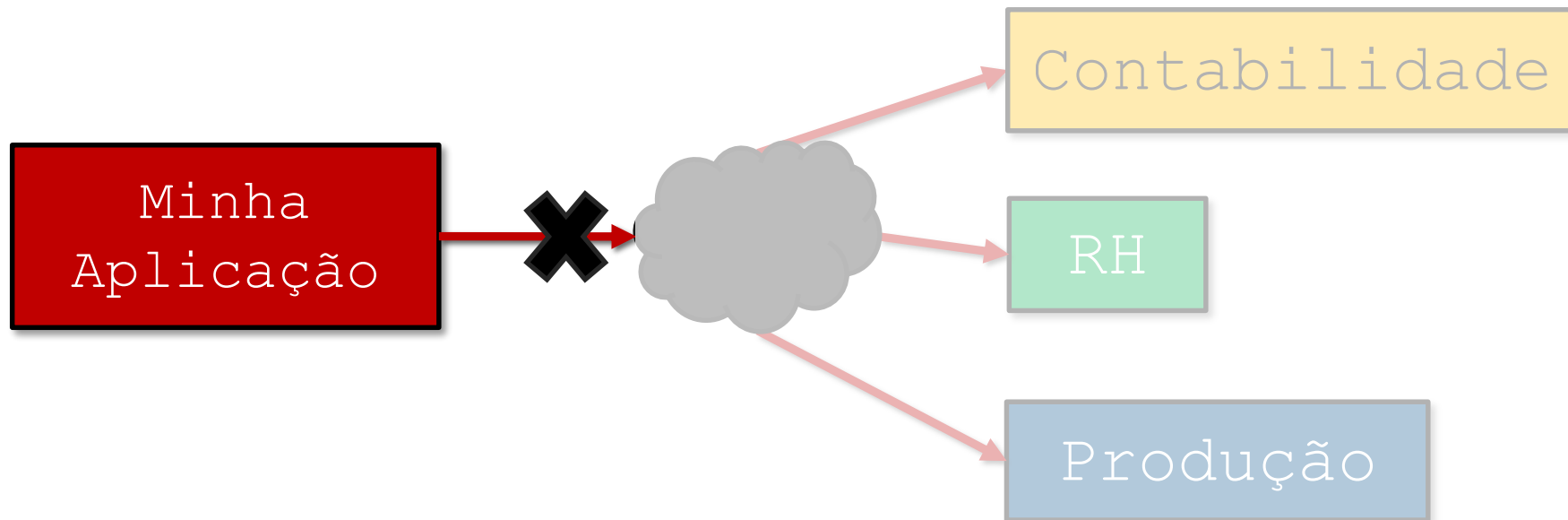
Pense em performance SEMPRE!



Modelagem | aplicação offline

4) Aplicação "offline"

Pense em performance SEMPRE!



Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)

Não é proibido utilizar chave primária composta!

Porém, é sugerido que se evite!

Prefira definir UMA coluna apenas, preferencialmente sequencial (sequence).

Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)

O maior problema da chave primária composta (mais de uma coluna) é quando outra tabela faz referência para esta:

Tabela A	
PK	Campo1
PK	Campo2
PK	Campo3
	Campo4
	Campo5

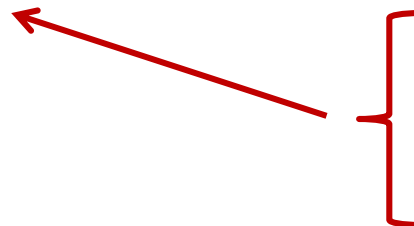


Tabela B	
PK	Campo1
	Campo2
FK	Campo1_tbA
FK	Campo2_tbA
FK	Campo3_tbA

Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)

Executando um estudo de caso.

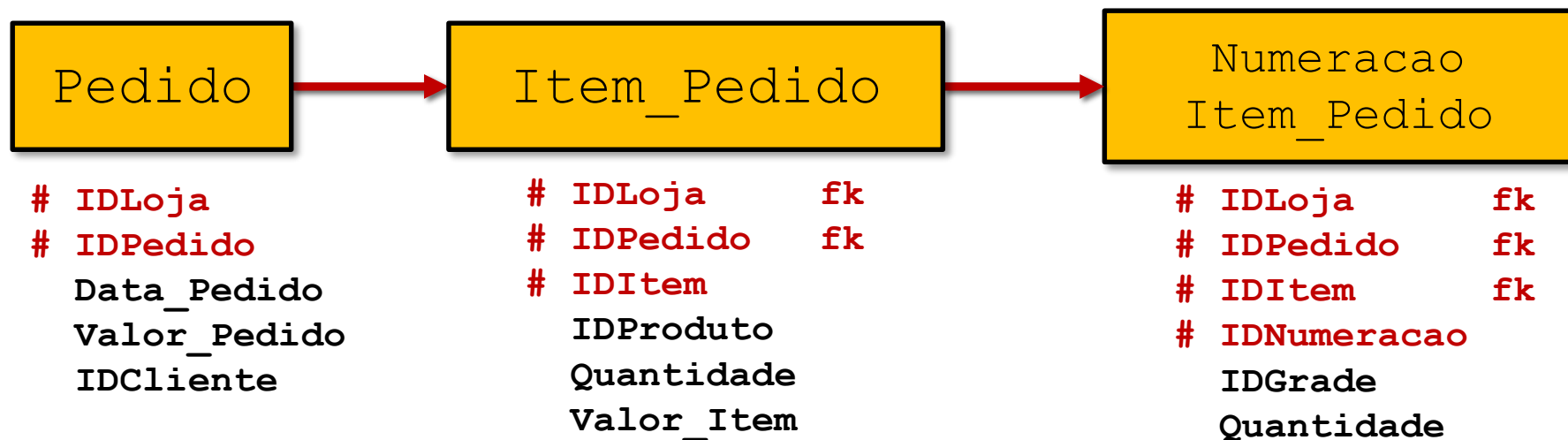
Situação A: tabelas com chaves primárias compostas.

Situação B: tabelas com chaves primárias simples.



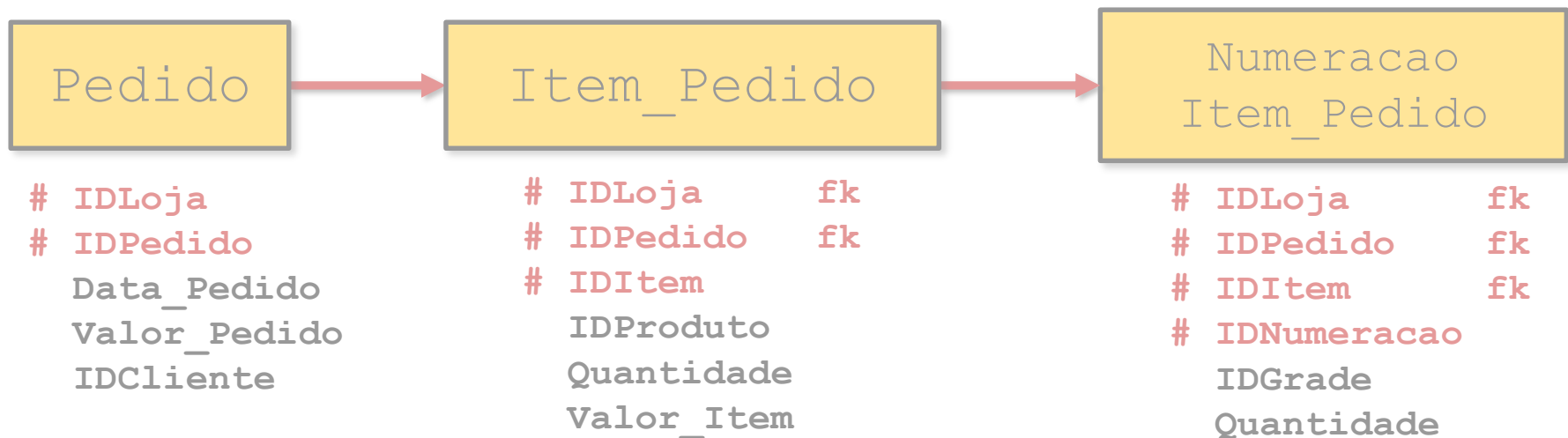
Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)



Modelagem | chave primária composta

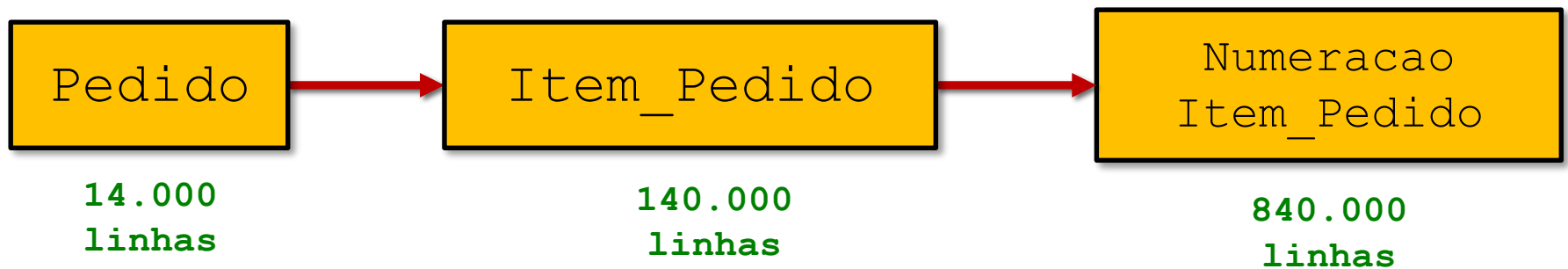
5) Evite chave primária composta (a polêmica)



Execute o arquivo "Create-Tables-PK-Composta.sql"

Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)



Execute o arquivo **"Create-Proc-Carga-Composta.sql"**

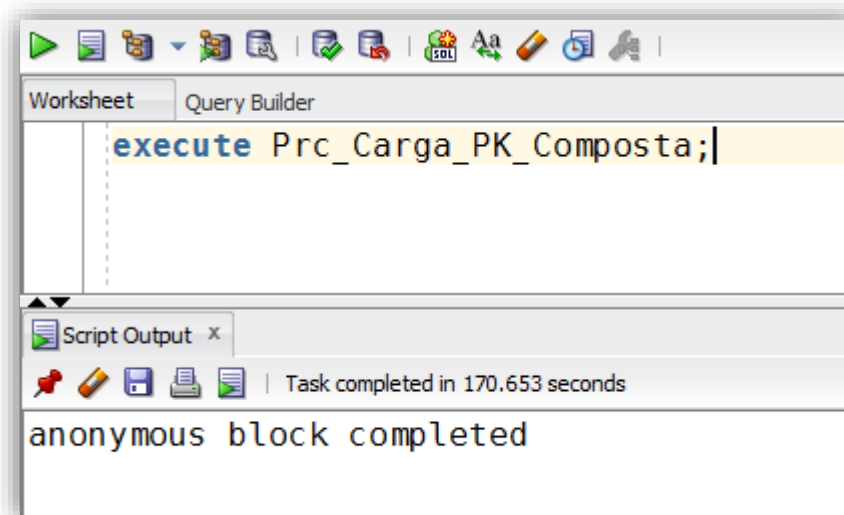
Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)

Passo 1: criação da estrutura de tabelas com **PK composta**;

Passo 2: criação do procedimento que fará a carga;

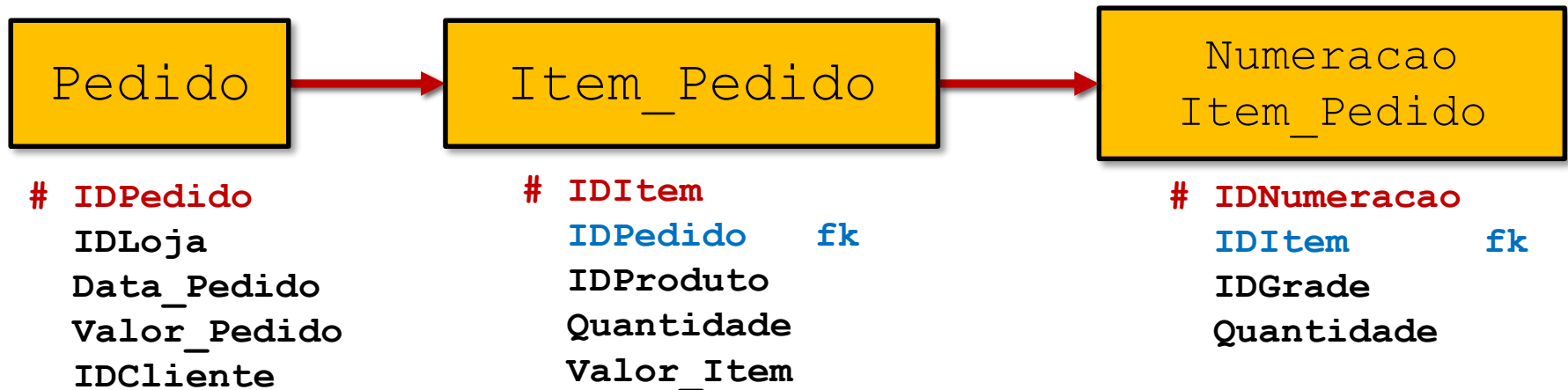
Passo 3: executar o procedimento de carga.



170 segundos

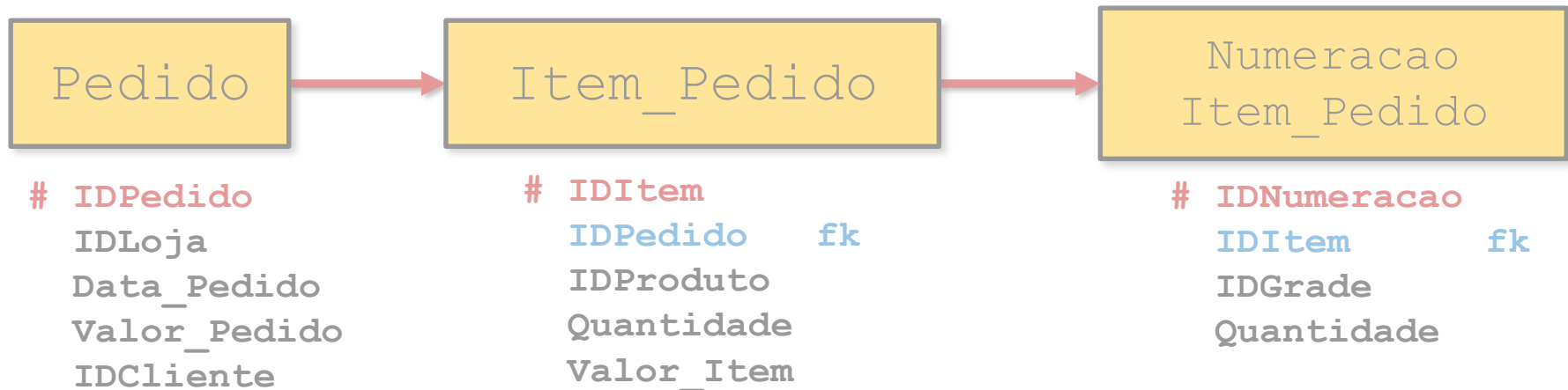
Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)



Modelagem | chave primária composta

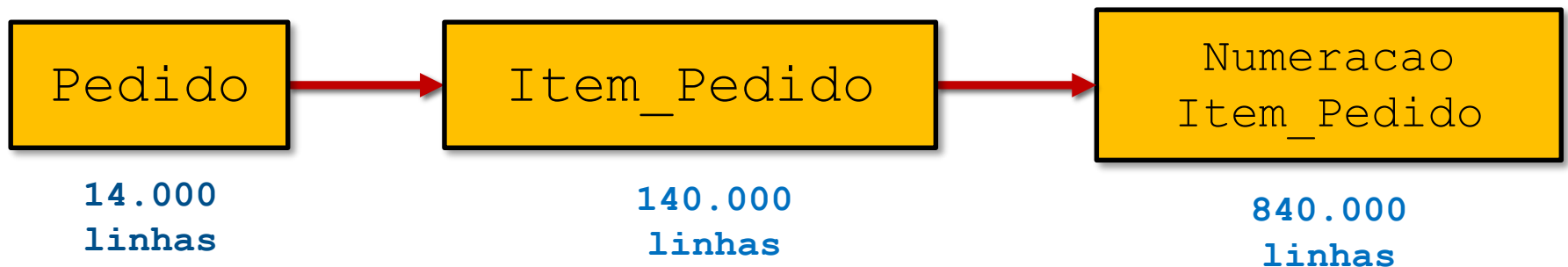
5) Evite chave primária composta (a polêmica)



Execute o arquivo "Create-Tables-PK-Simples.sql"

Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)



Execute o arquivo **"Create-Proc-Carga-Simples.sql"**

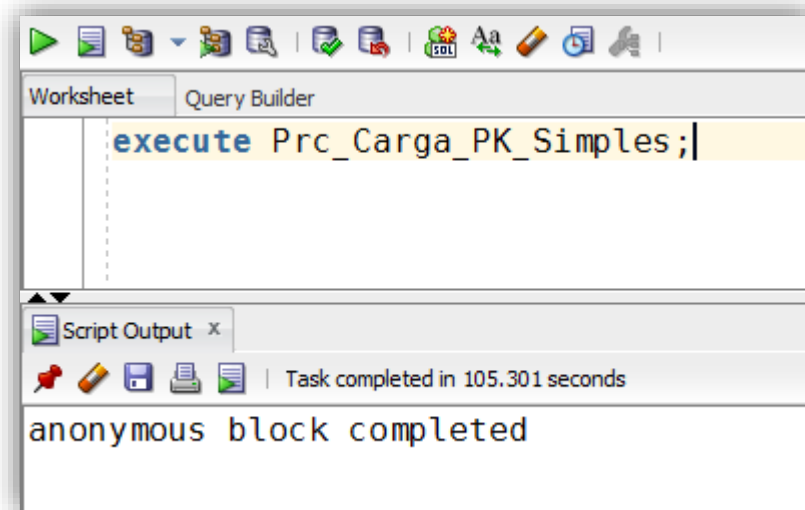
Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)

Passo 1: criação da estrutura de tabelas com **PK simples**;

Passo 2: criação do procedimento que fará a carga;

Passo 3: executar o procedimento de carga.



105 segundos

Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)

A diferença de tempo não foi grande.

Difícilmente seria notado dentro de uma aplicação pequena ou média.

PK composta

170 segundos

PK simples

105 segundos

61,8%



Modelagem | chave primária composta

5) Evite chave primária composta (a polêmica)

Estatísticas geradas:

PK	Tabela	Registros	Blocks	Dados (MB)	Tam. Médio Linha (bytes)	Índices (MB)
Simples	ITEM_PEDIDO	140.000	496	4	20	6,00
	NUMERACAO_ITEM_PEDIDO	840.000	2512	20	16	27,00
	PEDIDO	14.000	58	0,5	23	0,69
58,19	MB total		3066	24,5		33,69
Composta	ITEM_PEDIDO	140.000	622	5	21	12,00
	NUMERACAO_ITEM_PEDIDO	840.000	2890	23	19	73,00
	PEDIDO	14.000	58	0,5	23	0,94
114,44	MB total		3570	28,5		85,94

Modelagem | chave primária composta

6) Outras dicas

Utilize de 5 colunas adicionais para cada tabela:

- * Data e hora da criação (created)
- * Data e hora da última alteração (last_changed)
- * Usuário criador (created_by)
- * Último usuário que alterou (changed_by)
- * Situação do registro (status = Active / Inativo)

Modelagem | chave primária composta

6) Outras dicas

Situações que que serão úteis?

- Controle de alterações, rastreabilidade ou replicação;
- Exclusão lógica.



André Luís Nunes

andre.nunes@cw.com.br