

cin.ufpe.br



UNIVERSIDADE FEDERAL DE PERNAMBUCO



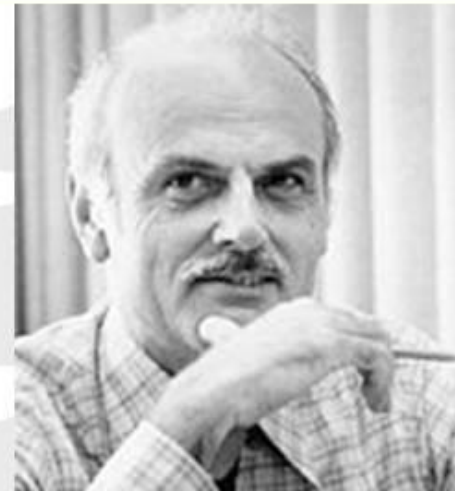
Modelo Relacional



Por: Prof. Robson do Nascimento Fidalgo

Visão Geral

- ▶ Definido em 1970 por Codd, E.F.
- ▶ Modelo lógico
 - ▶ Não é abstrato como o conceitual, tampouco considera aspectos físicos de armazenamento, acesso e desempenho
- ▶ Tem sólida base formal (teoria dos conjuntos) e é baseado em conceitos simples (relações com atributos, tuplas e domínios)
 - ▶ Relações capturam dados de entidades ou relacionamentos
- ▶ Primeiro produto (1979): Oracle
- ▶ Base para a linguagem declarativa (o que fazer) Structured Query Language (SQL)



Definições

▶ Domínio

- ▶ Conjunto de valores atômicos

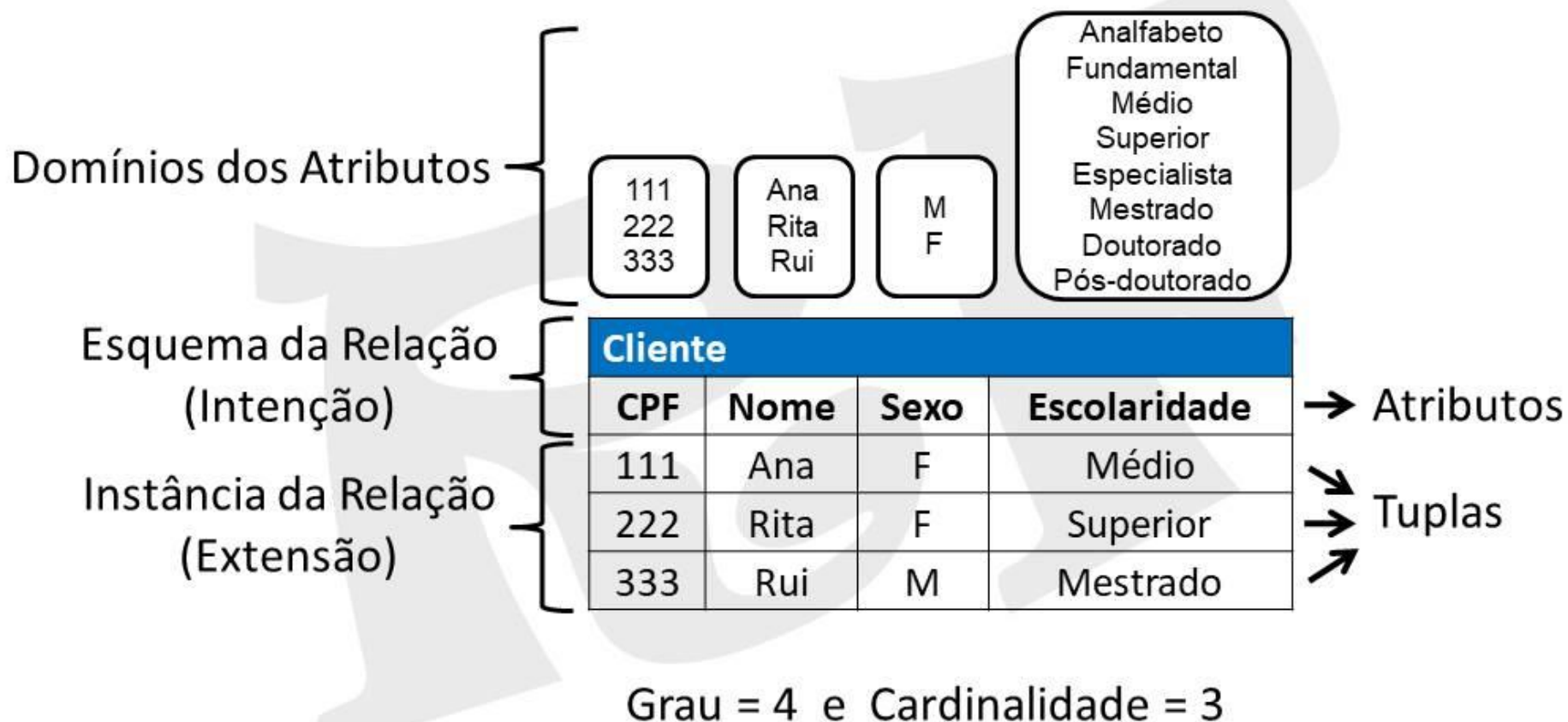
▶ Relação

- ▶ Dados os conjuntos D_1, D_2, \dots, D_n (Domínios não necessariamente distintos), R é uma relação nestes n conjuntos se esta for um conjunto de tuplas $\langle v_1, v_2, \dots, v_n \rangle$ onde $v_1 \in D_1, v_2 \in D_2, \dots$ e $v_n \in D_n$, formando um subconjunto do produto cartesiano $D_1 \times D_2 \times \dots \times D_n$

▶ Exemplo

- ▶ Nomes = {Ana, Rita, Rui} e Sexos = {M, F}
- ▶ Tupla = (Ana, F)
- ▶ Nomes \times Sexos = {(Ana, M), (Ana, F), (Rita, M), (Rita, F), (Rui, M), (Rui, F)}
- ▶ Clientes = {(Ana, F), (Rita, F), (Rui, M)}

Terminologia (Relação X Tabela)



Propriedades de uma Relação

- ▶ Toda relação tem um número fixo de atributos distintos
- ▶ O valor null é usado quando um atributo não tem valor ou este é desconhecido
- ▶ A ordem dos atributos e das tuplas é irrelevante
 - ▶ Não há ordenação entre tuplas e valores podem ser associados aos atributos independentemente de uma ordem

Chaves

- ▶ Conceito usado para identificar/referenciar tuplas
- ▶ Quatro tipos:
 - ▶ chave candidata
 - ▶ chave primária
 - ▶ chave estrangeira
 - ▶ chave alternativa

Chave Candidata

- ▶ É um atributo (chave simples) ou uma concatenação de atributos (chave composta) cujos valores distinguem uma tupla das demais tuplas de uma relação
- ▶ Deve ser mínima (ex. Matrícula ou CPF)
 - ▶ Chaves compostas não mínimas podem gerar inconsistência
 - Ex. Matrícula+CPF permite várias matrículas por CPF e vice-versa

Chaves Candidatas

simples

Cliente		
Matrícula	CPF	Nome
1111	101.010.101-01	Rita
2222	202.020.202-02	Ana
3333	303.030.303-03	Pedro
4444	404.040.404-04	José

composta

Dependente		
Matrícula	Num	Nome
1111	1	Pedro
1111	2	Ruth
2222	1	Rosa
3333	1	João

Chave Primária

- ▶ É a chave candidata escolhida para identificar uma tupla
- ▶ É frequentemente utilizada para selecionar as tuplas de uma relação
- ▶ Não admite valor null

Chaves Primárias

simples

Cliente		
<u>Matrícula</u>	CPF	Nome
1111	101.010.101-01	Rita
2222	202.020.202-02	Ana
3333	303.030.303-03	Pedro
4444	404.040.404-04	José

composta

Dependente		
<u>Matrícula</u>	<u>Num</u>	Nome
1111	1	Pedro
1111	2	Ruth
2222	1	Rosa
3333	1	João

Chave Estrangeira

- ▶ Um atributo ou uma concatenação de atributos que faz referência a uma chave primária
- ▶ É utilizada para relacionar tuplas de relações
- ▶ Admite valor null (participação opcional)

Chave Estrangeira

simples

Cliente		
Matrícula	CPF	Nome
1111	101.010.101-01	Rita
2222	202.020.202-02	Ana
3333	303.030.303-03	Pedro
4444	404.040.404-04	José

Dependente		
Matrícula	Num	Nome
1111	1	Pedro
1111	2	Ruth
2222	1	Rosa
3333	1	João

Chave Estrangeira (auto-relacionamento)

Chave Primária
simples

Chave Estrangeira
simples

Empregado		
<u>CodEmpregado</u>	Nome	<u>CodChefe</u>
1010	Ana	
2020	Paulo	1010
3030	Fred	1010
4040	Silvia	2020

Auto-Relacionamento

Chave Alternativa (ou secundária)

- ▶ É a chave candidata que não foi escolhida como chave primária

Chave Alternativa

simples

Cliente		
Matrícula	CPF	Nome
1111	101.010.101-01	Rita
2222	202.020.202-02	Ana
3333	303.030.303-03	Pedro
4444	404.040.404-04	José

Dependente		
Matrícula	Num	Nome
1111	1	Pedro
1111	2	Ruth
2222	1	Rosa
3333	1	João

Chave Alternativa (ou secundária)

ATENÇÃO:
Chave Alternativa não faz
relacionamento com
Chave Estrangeira !

Restrições de integridade

- ▶ Regras sobre os valores armazenados nas relações
- ▶ Têm por objetivo garantir a consistência das relações
- ▶ Quatro tipos:
 - ▶ Restrições de Domínio
 - ▶ Restrições de Chave
 - ▶ Integridade da Entidade
 - ▶ Integridade Referencial

Restrições de integridade

- ▶ Restrições de Domínio
 - ▶ Todo valor de um atributo deve ser atômico (simples e mono valorado) e pertencer ao domínio do atributo
- ▶ Restrições de Chave
 - ▶ Todo valor de chave primária deve ser mínimo e único na relação
- ▶ Integridade da Entidade
 - ▶ Chaves primárias não podem ter valor null
- ▶ Integridade referencial
 - ▶ Especifica que os valores de uma chave estrangeira devem aparecer na chave primária da tabela referenciada

Restrições de integridade (Exemplos)

- ▶ Inserções ou atualizações
- ▶ Podem violar as restrições de:
 - ▶ Domínio - valor não atômico ou diferente do permitido
 - ▶ Chave - valor duplicado de chave primária
 - ▶ Integridade de entidade - valor null para chave primária
 - ▶ Integridade referencial - o valor da chave estrangeira não existe na chave primária referenciada
- ▶ Solução: rejeitar e lançar mensagem de erro

Restrições de integridade (Exemplos)

- ▶ Exclusões
 - ▶ Só podem violar restrições de Integridade referencial - o valor da chave primaria existe em um chave estrangeira que a referencia
- ▶ Soluções:
 - ▶ Rejeitar e lançar mensagem de erro,
 - ▶ Excluir em cascata ou
 - ▶ Definir o valor null ou um valor padrão

Restrições Semânticas

- ▶ São restrições para impor regras de negócio
- ▶ Estas devem ser implementadas pelos programadores (dentro ou fora do SGBD), pois não são automaticamente garantidas
- ▶ Ex: Um empregado não pode ter um salário maior que seu superior imediato

Notação Simplificada

- ▶ Esquema Relacional = definição das tabelas
- ▶ Representação básica (incompleta mas compacta)

→ = Chave Estrangeira

Sublinhado = Chave Primária

▶ Ex:

FUNCIONARIO (FUNC_PK, nome, ..., DEPTO_FK!)

DEPTO_FK → DEPARTAMENTO (COD)

DEPARTAMENTO (COD, nome, ..., [CHEFE_FK]!)

CHEFE_FK → (FUNC_PK)

[] = Valor Único (unique)

! = Valor Obrigatório (not null)



Álgebra Relacional



Por: Prof. Robson do Nascimento Fidalgo

Linguagens de Consulta Relacionais

- ▶ Linguagens de consulta
 - ▶ Permitem recuperar e manipular dados
 - ▶ Modelo relacional dá suporte à linguagens de consultas simples, poderosas e com forte embasamento formal
 - ▶ São linguagens declarativas
 - ▶ Focam no “o que fazer” e não no “como fazer”
- ▶ Duas propostas bem aceitas:
 - ▶ **Álgebra Relacional** e **Cálculo Relacional**

Álgebra Relacional

- ▶ Desenvolvida para descrever operações sobre um BDR
 - ▶ Ajudam a entender SQL
- ▶ Principais operações:
 - União
 - Interseção
 - Diferença
 - Produto Cartesiano
 - União Exclusiva

Operações sobre Conjuntos

 - Seleção
 - Projeção

Operações Relacionais Unárias

 - Junção
 - Divisão

Operações Relacionais Binárias

Compatibilidade de domínio

- ▶ Duas relações $A(a_1, a_2, \dots, a_n)$ e $B(b_1, b_2, \dots, b_n)$ são ditas compatíveis em domínio se ambas têm o mesmo grau n e se $\text{Dom}(a_i) = \text{Dom}(b_i)$, $1 \leq i \leq n$

- ▶ Exemplo:

- ▶ Aluno (nome, idade, curso)
- ▶ Professor (nm, idd, crs)
- ▶ Funcionario (nome, curso, idade)

Dom(nome) = char(30)
Dom(nm) = char(30)
Dom(idade) = int
Dom(idd) = int
Dom(curso) = char(10)
Dom(depto) = char(10)
Dom(crs) = char(10)

Aluno é compatível com Professor, mas não é com Funcionario.

- ▶ Note que :
 - ▶ A estrutura de uma relação (tabela) é mais importante do que sua semântica
 - ▶ A ordem dos atributos prevalece

Operações sobre conjuntos

- ▶ Tais operações são as usuais da teoria dos conjuntos:
 - ▶ **União:** $(A \cup B) \rightarrow$ Une as tuplas das relações A e B.
 - ▶ **Interseção:** $(A \cap B) \rightarrow$ Retorna as tuplas cujos valores sejam comuns às A e B
 - ▶ **Diferença:** $(A - B) \rightarrow$ Retorna as tuplas de A cujos valores não estão em B
 - ▶ **Produto cartesiano:** $(A \times B) \rightarrow$ Combina todas as tuplas das relações A e B
 - ▶ **União exclusiva:** $(A \cup | B) \rightarrow$ Retorna todas as tuplas de A ou a B que não estão em ambas, ou seja, $A \cup | B = A \cup B - A \cap B$
- ▶ Com exceção do produto cartesiano, as demais operações exigem relações compatíveis em domínio

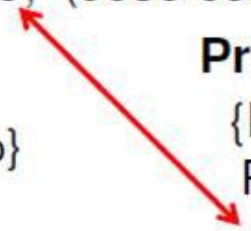
União

Aluno (nome, idade, curso)

{José, 25, Computação; (José como aluno de Doutorado)
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação}

Professor (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química;
José, 25, Computação}



EX: Retornar todos os alunos e professores da Universidade

Aluno \cup Professor = (nome, idade, curso)

{José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação;
Ruth, 35, Computação;
Rosa, 32, Química}

Convenciona-se usar os nomes dos atributos da relação a esquerda, quando não especificado.

Note que José só aparece uma vez !

Interseção

Aluno (nome,idade,curso)

{José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação}

Professor (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química;
José, 25, Computação}

EX: Retornar todos que ao mesmo tempo sejam alunos e professores da Universidade

Aluno \cap Professor = (nome,idade,curso)

{José, 25, Computação}

Diferença

Aluno (nome,idade,curso)

{José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação}

Professor (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química;
José, 25, Computação}

EX1: Retornar todos os alunos que não são professores

EX2: Retornar todos os professores que não são alunos

EX1:Aluno - Professor = (nome,idade,curso)

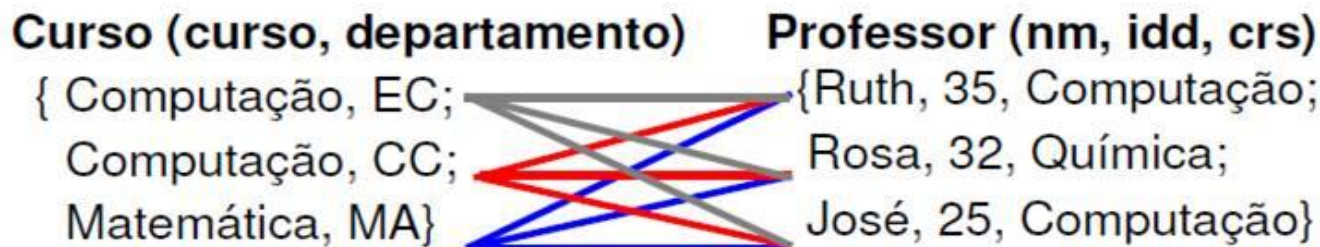
{ Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação}

EX2:Professor - Aluno= (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química}

Note que a Diferença não é comutativa, ou seja, $A-B \neq B-A$

Produto Cartesiano



EX: Retornar todas as combinações entre os cursos e os professores da Universidade

Curso X Professor = (curso, departamento, nm, idd, crs)

{ Computação, EC, Ruth, 35, Computação;
 Computação, EC, Rosa, 32, Química;
 Computação, EC, José, 25, Computação;
 Computação, CC, Ruth, 35, Computação;
 Computação, CC, Rosa, 32, Química;
 Computação, CC, José, 25, Computação;
 Matemática, MA, Ruth, 35, Computação;
 Matemática, MA, Rosa, 32, Química;
 Matemática, MA, José, 25, Computação}

União Exclusiva

Aluno (nome,idade,curso)

{José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação}

Professor (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química;
José, 25, Computação}

EX: Retornar todos que ao mesmo tempo não são aluno e professor da Universidade

Aluno \cup Professor = (nome, idade, curso)

{ Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação;
Ruth, 35, Computação;
Rosa, 32, Química}



Operações Relacionais Unárias

- ▶ Produzem como resultado uma nova relação que é um subconjunto (horizontal ou vertical) da relação origem
- ▶ São elas:

Seleção: ($\sigma_{\langle \text{condição} \rangle}(\text{Relação})$) \Rightarrow seleciona tuplas de uma relação que satisfazem um dada condição.

- Onde (Relação) é uma tabela ou uma expressão de álgebra relacional, e $\langle \text{condição} \rangle$, uma expressão booleana (and, or, not, =, \neq , < , \leq , > , \geq) envolvendo atributos da tabela
- Produz um subconjunto horizontal de uma relação

Projeção: ($\pi_{\langle \text{atributos} \rangle}(\text{Relação})$) \Rightarrow seleciona de uma relação os atributos de interesse

- Onde (Relação) é uma tabela ou uma expressão de álgebra relacional, e $\langle \text{atributos} \rangle$, uma lista de colunas da tabela operando
- Produz um subconjunto vertical de uma relação

Seleção

Aluno (nome,idade,curso)

{José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação}

Professor (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química;
José, 25, Computação}

EX: Retornar todos os alunos maiores de 20 anos do curso de computação da Universidade

$\sigma_{\langle \text{idade} > 20 \text{ and curso} = \text{"Computação"} \rangle}(\text{Aluno}) = (\text{nome, idade, curso})$
{José, 25, Computação}

Outra solução:

$\sigma_{\langle \text{idade} > 20 \rangle}(\sigma_{\langle \text{curso} = \text{"Computação"} \rangle}(\text{Aluno})) = (\text{nome, idade, curso})$

{José, 25, Computação}  {José, 25, Computação;
Ana, 19, Computação}

Projeção

Aluno (nome, idade, curso)

{José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação}

Professor (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química;
José, 25, Computação}

EX: Retornar todos os alunos e seus cursos

$\pi_{\langle \text{nome, curso} \rangle}(\text{Aluno}) = (\text{nome, curso})$

{José, Computação;
Pedro, Química;
Paulo, Física;
Ana, Computação}

OB: Na Projeção pode haver eliminação de linhas

EX: Retornar todos os cursos que têm alunos matriculados na Universidade

$\pi_{\langle \text{curso} \rangle}(\text{Aluno}) = (\text{curso})$

{ Computação;
Química;
Física}

Projeção + Seleção

Aluno (nome, idade, curso)

{ José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação }

Professor (nm, idd, crs)

{ Ruth, 35, Coputação;
Rosa, 32, Química;
José, 25, Computação }

EX: Retornar os nomes e cursos dos alunos maiores de 20 anos do curso de Computação da Universidade

$\Pi_{\langle \text{nome}, \text{curso} \rangle} \sigma_{\langle \text{idade} > 20 \text{ and } \text{curso} = \text{"Computação"} \rangle} \text{Aluno}$

Operações Relacionais Binárias

- ▶ Produz como resultado uma nova relação que é um subconjunto (seleção) do produto cartesiano das relações envolvidas
 - ▶ Em geral, após o produto cartesiano, é necessário comparar um grupo de atributos (compatíveis em domínio) para selecionar as tuplas do resultado final
- ▶ São elas:
 - ▶ Junção:
 - ▶ Divisão:

Operações Relacionais Binárias - Junção

- ▶ Retorna apenas as tuplas do produto cartesiano de seus argumentos que satisfaçam uma dada condição

Sintaxe: $((R1) \bowtie_{\langle \text{condição} \rangle} (R2)) = (\sigma_{\langle \text{condição} \rangle} (R1 \times R2))$

- ▶ Onde (R1) e (R2) são relações ou expressões de álgebra relacional, e $\langle \text{condição} \rangle$ é uma expressão booleana envolvendo atributos das duas relações

Operações Relacionais Binárias - Junção

Aluno (nome,idade,curso)

{José, 25, Computação;
Pedro, 21, Química;
Paulo, 19, Física;
Ana, 19, Computação;
João, 34, Computação}

Professor (nm, idd, crs)

{Ruth, 35, Computação;
Rosa, 32, Química;
José, 25, Computação}

- ▶ EX: Retornar todos os alunos mais velhos do que qualquer professor da Universidade

((Aluno) ⋈_{<Aluno.idade > Professor.idade>} (Professor))
=
(nome, idade, curso, nm, idd, crs)

{João, 34, Computação, Rosa, 32, Química
João, 34, Computação, José, 25, Computação}

Operações Relacionais Binárias - Divisão

- ▶ Produz uma relação $R(X)$ com as tuplas de $R1(A)$ que estão combinadas com todas as tuplas de $R2(B)$.
- ▶ Sintaxe: $R(X) = R1(A) \div R2(B)$,
- ▶ Onde:
 - ▶ $B \subseteq A$ e
 - ▶ $X = A - B$
 - ▶ $R1$ e $R2$ são relações ou expressões de álgebra relacional

Operações Relacionais Binárias - Divisão

Matricula (nome-a, discipl, nota)

{José, IF111, 9,0;
Pedro, IF333, 3,5;
Paulo, IF111, 7,5;
Paulo, IF333, 6,5;
José, IF333, 10,0;
José, IF222, 6,5;
Ana, IF222, 7,0 }

Aulas (nome-p, discipl)

{Lopes, IF111;
Joana, IF222;
Lopes, IF333}

EX:Retornar os alunos que cursam todas as disciplinas ministradas pelo Prof. Lopes?

$\pi_{\langle \text{nome-a, discipl} \rangle}(\text{Matricula}) \div \pi_{\langle \text{discipl} \rangle}(\sigma_{\text{nome-p}=\text{"Lopes"}}(\text{Aulas}))$

{ José , Paulo }

Note: Todas as operações podem ser combinadas entre si

Operações Relacionais Binárias - Divisão

Piloto (nome, avião)

{Pedro, 101;
Pedro, 105;
Bruno, 101;
Bruno, 104;
Bruno, 105;
Bruno, 103;
Paulo, 103;
Paulo, 104}

Avião (identificação)

{101;
104;
105;
103}

EX:Retornar os pilotos que estão habilitados para conduzir todos os aviões da companhia.

(Piloto) \div (Avião) = (nome)

{ Bruno }

cin.ufpe.br



UNIVERSIDADE FEDERAL DE PERNAMBUCO