**Álgebra Relacional**

Álgebra Relacional é o conjunto básico de operações para o Modelo Relacional. Essas operações permitem a um usuário especificar as solicitações básicas de recuperação.

O resultado de uma recuperação será uma nova relação que pode ter sido formada de uma ou mais relações.

Uma sequência de operações de Álgebra Relacional forma uma expressão de Álgebra Relacional cujos resultados também serão uma relação que representa o resultado de uma consulta de BD (ou solicitação de recuperação).

Importância da Álgebra Relacional:

* Provê um fundamento formal para operações do modelo relacional;
* É usada como uma base para implementar e otimizar as consultas em SGBDRs;
* Alguns de seus conceitos são incorporados na linguagem de consulta padrão SQL para SGBDRs.

**Operadores definidos pelo modelo relacional:**

Seleção União

Projeção Interseção

Junção Diferença

Divisão Produto Cartesiano

OBS: as operações Seleção e Projeção são operações unárias porque atuam em relações únicas.

PROFESSOR ALUNO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Matricula** | **Nome** |  | **Matricula** | **Nome** |
| 101 | Ana | 103 | Marcos |
| 110 | Carlos | 101 | Ana |
| 120 | Jaqueline | 105 | Janete |

**União A**  **B**

União é a operação entre duas relações cujo resultado contém todas as tuplas presentes em uma das relações originais ou em ambas, onde:

* As relações devem ter uma estrutura similar, com o mesmo número de colunas e domínios compatíveis dos atributos correspondentes.
* Tuplas duplicadas são eliminadas PROFESSOR  ALUNO

|  |  |
| --- | --- |
| **Matricula** | **Nome** |
| 101 | Ana |
| 110 | Carlos |
| 120 | Jaqueline |
| 103 | Marcos |
| 105 | Janete |

**Interseção A**  **B**

Interseção é a operação entre duas relações de estrutura similar cujo resultado contém todas as tuplas presentes em ambas.

* Tuplas duplicadas são eliminadas PROFESSOR  ALUNO

|  |  |
| --- | --- |
| **Matricula** | **Nome** |
| 101 | Ana |





**Diferença A - B**

Diferença é a operação entre duas relações de estrutura similar cujo resultado contém todas as tuplas que estão na primeira relação e não aparecem na segunda.

* A - B é diferente de B – A

PROFESSOR - ALUNO ALUNO - PROFESSOR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Matricula** | **Nome** |  | **Matricula** | **Nome** |
| 110 | Carlos | 103 | Marcos |
| 120 | Jaqueline | 105 | Janete |

**Seleção** condição(tabela)

A operação de Seleção é usada para selecionar um subconjunto de tuplas de uma relação que satisfaça uma condição de seleção. Em geral a operação de seleção é indicada por *<condição de seleção>* (R). A operação de seleção opera sobre uma relação, gerando uma relação resultante que contém todas as colunas das tuplas selecionadas da relação original.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

CIDADE

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Cidade** | **Nome\_Cidade** | **Capital** | **UF** |
| 1234 | Porto Alegre | S | RS |
| 2345 | São Paulo | S | SP |
| 1235 | Caxias do Sul | N | RS |

UF=’RS’(CIDADE)

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Cidade** | **Nome\_Cidade** | **Capital** | **UF** |
| 1234 | Porto Alegre | S | RS |
| 1235 | Caxias do Sul | N | RS |

UF=’RS’ and Capital=’S’(CIDADE)

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Cidade** | **Nome\_Cidade** | **Capital** | **UF** |
| 1234 | Porto Alegre | S | RS |

**Projeção** A1, A2,…,An(tabela)

Enquanto a operação de seleção seleciona algumas das tuplas da relação e descarta outras, a operação de projeção seleciona certas colunas da relação e descarta outras.

Se estivermos interessados apenas em certos atributos de uma relação, usamos a operação Projeção.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Nome\_Cidade, UF(CIDADE)

|  |  |
| --- | --- |
| **Nome\_Cidade** | **UF** |
| Porto Alegre | RS |
| São Paulo | SP |
| Caxias do Sul | RS |

UF(CIDADE)

|  |
| --- |
| **UF** |
| RS |
| SP |
| RS |

obs: tuplas (linhas) duplicadas são eliminadas

**Sequências de Operações e Rename**

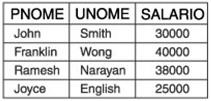
Normalmente é desejável aplicar várias operações de álgebra relacional, uma após a outra. Ex.: Listar o primeiro e o último nome dos empregados do departamento 5.

- Temos uma seleção seguida de uma projeção.

**1ª opção**: Aninhar as operações:

 PNOME, UNOME, SALARIO ( DNO = 5 (EMPREGADO) )

* Note que a ordem destas operações neste exemplo não pode ser invertida.
* Qual seria a relação de resposta para esta operação? Veja a seguir:



**2ª opção**: criar relações intermediárias:

TEMP  

DNO = 5

(EMPREGADO)

RESULTADO   PNOME, UNOME, SALARIO (TEMP)

- Renomeando atributos durante uma atribuição:

TEMP   DNO = 5 (EMPREGADO)

RESULTADO(PrimeiroNome, UltimoNome, Salario)   PNOME, UNOME, SALARIO(TEMP) O resultado final:

Usando o operador **Rename** para renomear atributos e relações. Suponha que temos uma relação R(A1, A2, ..., An).

*  (B1, B2, ..., Bn) (R): renomeia os atributos de R para B1, B2, ..., Bn.
*  S (R): renomeia a relação R (ela passa a se chamar S).
*  S (B1, B2, ..., Bn) (R): renomeia a relação em si e os atributos.

**Produto Cartesiano** tabela\_1  tabela\_2

A operação Produto Cartesiano é também conhecida como *Cross Product* (Produto Cruzado) ou *Cross Join* (Junção Cruzada) que é indicada por “x”. Essa é uma operação binária que combina as tuplas de duas relações de forma combinatória.

O resultado de R(A1, A2,..., An)  S(B1, B2,..., Bm) é uma relação Q, com grau n + m de atributos Q(A1, A2, ......An, B1, B2, Bm), nessa ordem.

|  |  |
| --- | --- |
| A | X |
| B | X |
| C | X |
| A | Y |
| B | Y |
| C | Y |

|  |
| --- |
| A |
| B |
| C |

=

|  |
| --- |
| X |
| Y |

Normalmente essa operação aplicada sozinha não tem sentido. Ela será útil quando for seguida por uma seleção que casar os valores de determinados atributos provindos das relações componentes.

# Exercícios

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A** | **B** |  | **B** | **C** | **D** |  | **A** | **B** | **B** | **C** | **D** |
| 1 | 2 | 2 | 5 | 6 | 1 | 2 | 2 | 5 | 6 |
| 3 | 4 | 4 | 7 | 8 | 1 | 2 | 4 | 7 | 8 |
|  | | | 9 | 10 | 11 | 1 | 2 | 9 | 10 | 11 |
|  | | | | | | | 3 | 4 | 2 | 5 | 6 |
| 3 | 4 | 4 | 7 | 8 |
| 3 | 4 | 9 | 10 | 11 |

1. - Descreva a relação resultado do produto cartesiano R  S R S R  S

1. - Descreva a relação resultado do produto cartesiano CIDADE  ESTADO

### CIDADE ESTADO

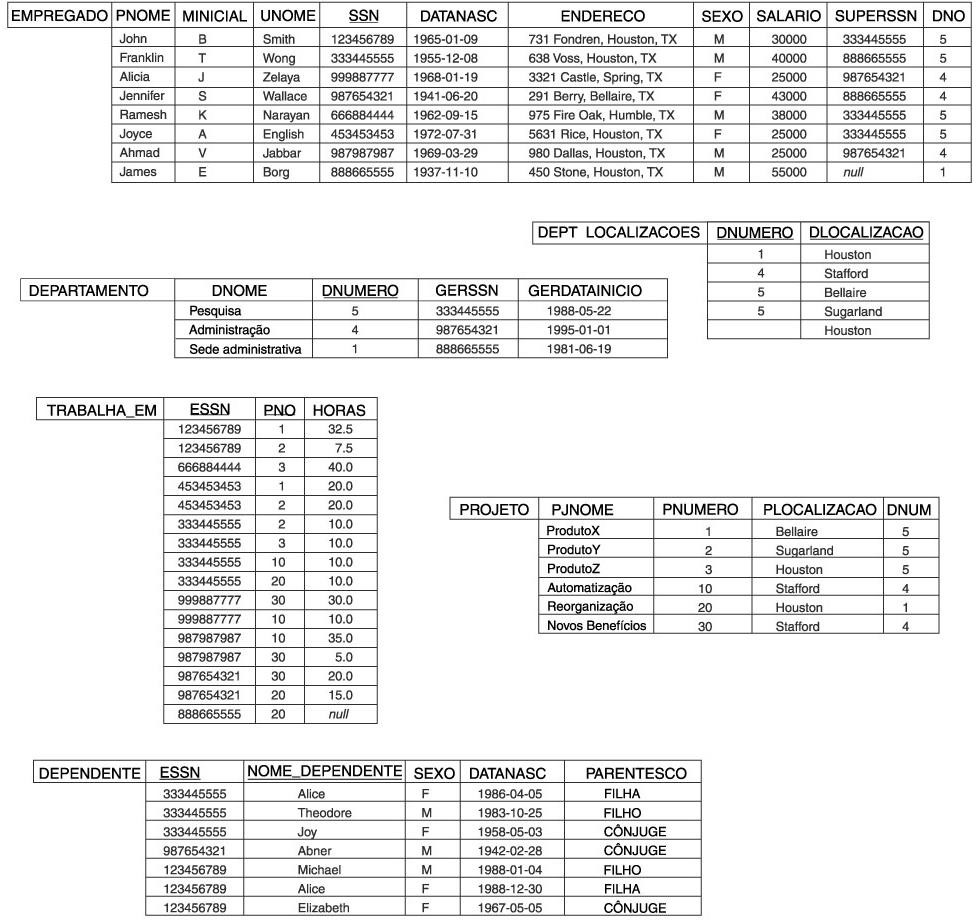
|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Cidade** | **Nome\_Cid.** | **Capital** | **UF** |
| 1234 | Porto Alegre | S | RS |
| 2345 | São Paulo | S | SP |
| 1235 | Caxias do Sul | N | RS |

|  |  |  |
| --- | --- | --- |
| **UF** | **Nome\_Estado** | **Regiao** |
| RS | Rio Grande do Sul | Sul |
| SP | São Paulo | Sudeste |

### CIDADE  ESTADO

### 

1. - Considerando os dados apresentados abaixo, faça:



//2190 + alt + x = ←

//03c0 + alt + x = π

//03c3 + alt + x = σ

//2A1D + alt + x = ⨝

//

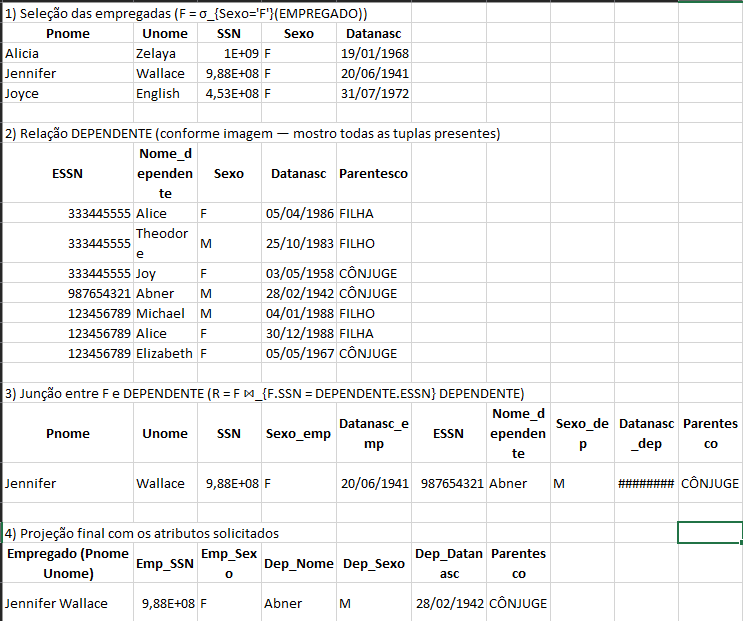
//2A1D + alt

* 1. Com a Álgebra Relacional descreva as seguintes consultas:
     + Montar uma lista com o nome, sexo, data de nascimento e parentesco dos dependentes de cada empregado do sexo feminino, além do nome, SSN e Sexo do empregado.

EMP ← σ sexo=’F’(EMPREGADO)

RESULTADO ← EMP.pnome, EMP.unome, EMP.sexo, EMP.ssn, DEPENDENTE.parentesco(

EMP ⨝ssn=essn DEPENDENTE)

* + - Mostre como ficaram as relações intermediárias.
    - 
  1. Listar a SSN de todos os empregados do departamento 5 e os SSN‟s dos empregados que são supervisores diretos de algum empregado do departamento 5.

**Junção-Theta (Junção)** Tabela\_1 condição Tabela\_2

A operação Junção indicada por  é usada para combinar as tuplas relacionadas em duas relações dentro de uma tupla única.

A operação de Junção pode ser definida por um produto cartesiano seguido por uma seleção.

* Além da *igualdade* outras condições podem utilizadas em uma junção, como ***menor*** (A<D), conforme exercício abaixo.

# Exercícios

1. Mostre o resultado da junção-theta aplicada sobre as relações U e V.
   1. U V U  A < D V

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A | B | C |  | B | C | D |
| 1 | 2 | 3 | 2 | 3 | 4 |
| 6 | 7 | 8 | 2 | 3 | 5 |
| 9 | 7 | 8 | 7 | 8 | 10 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C | B | C | D |
| 1 | 2 | 3 | 2 | 3 | 4 |
| 1 | 2 | 3 | 2 | 3 | 5 |
| 1 | 2 | 3 | 7 | 8 | 10 |
| 6 | 7 | 8 | 7 | 8 | 10 |
| 9 | 7 | 8 | 7 | 8 | 10 |
|  |  |  |  |  |  |

* 1. U  A < D **AND** U.B  V.B V

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | B | C | B | C | D |
| 1 | 2 | 3 | 7 | 8 | 10 |
|  |  |  |  |  |  |

Para ilustrar a Junção, suponha que queiramos recuperar o nome do gerente de cada departamento. Para obter o nome do gerente precisaremos combinar cada tupla departamento com a tupla empregado na qual o valor do SSN case com o valor GERSSN da tupla departamento:

**DEPT\_GER**  **DEPARTAMENTO ** **GERSSN = SSN EMPREGADO RESULTADO**  **DNOME, UNOME, PNOME(DEPTO\_GER)**

***As variações de Junção:***

**Equijoin:**

* Quando a condição de uma junção é a igualdade, a junção é chamada de **Equijoin** (observar que a tabela resultante terá sempre duas colunas idênticas).

**CIDADE ** **Cidade.UF = Estado.UF ESTADO**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Cod\_Cidade** | **Nome\_Cidade** | **Capital** | **Cidade**  **.UF** | **Estado**  **.UF** | **Nome\_Estado** | **Regiao** |
| 1234 | Porto Alegre | S | RS | RS | Rio Grande do Sul | Sul |
| 2345 | São Paulo | S | SP | SP | São Paulo | Sudeste |
| 1235 | Caxias do Sul | N | RS | RS | Rio Grande do Sul | Sul |

colunas idênticas

**Junção Natural** Tabela\_1 Tabela\_2

Uma **Junção Natural** é um **Equijoin** em que uma das colunas idênticas é eliminada.

* O operador de junção natural combina as tuplas de duas relações que tem atributos comuns (mesmo nome), resultando numa relação que contém apenas as tuplas onde todos os atributos comuns apresentam o mesmo valor. A definição padrão de **Junção Natural** exige que os dois atributos de Junção tenham o mesmo nome em ambas as relações. Se esse não for o caso uma operação de **Rename** deve ser aplicada.
* Uma das colunas idênticas é eliminada, evitando a duplicidade:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tabela\_1** | A | B | C | D | E |  | | |
|  | | | | | | | | |
| **Tabela\_2** | | | | D | E | F | G | H |
|  | | | | | | | | |
| **Junção Natural** | A | B | C | D | E | F | G | H |

## CIDADE ESTADO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cód\_Cidade** | **Nome\_Cidade** | **Capital** | **UF** | **Nome\_Estado** | **Regiao** |
| 1234 | Porto Alegre | S | RS | Rio Grande do Sul | Sul |
| 2345 | São Paulo | S | SP | São Paulo | Sudeste |
| 1235 | Caxias do Sul | N | RS | Rio Grande do Sul | Sul |

***Exercícios***

1- Mostre o resultado da junção natural das relações U e V.

U V U  V

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A | B | C |  | B | C | D |
| 1 | 2 | 3 | 2 | 3 | 4 |
| 6 | 7 | 8 | 2 | 3 | 5 |
| 9 | 7 | 8 | 7 | 8 | 10 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D |  |  |  |  |
| 1 | 2 | 3 | 4 |  |  |  |  |
| 1 | 2 | 3 | 5 |  |  |  |  |
| 6 | 7 | 8 | 10 |  |  |  |  |
| 9 | 7 | 8 | 10 |  |  |  |  |
| 6 | 7 | 8 |  |  |  |  |  |
| 6 | 7 | 8 |  |  |  |  |  |
| 9 | 7 | 8 |  |  |  |  |  |
| 9 | 7 | 8 |  |  |  |  |  |
| 9 | 7 | 8 |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

No exemplo seguinte, primeiro nós renomearemos o atributo DNUMERO de DEPARTAMENTO para DNUM -- Assim ele terá o mesmo nome do atributo DNUM em PROJETO – então aplicaremos a Junção Natural:

DEP\_PROJ  PROJETO \* (DNOME, DNUM, GERSSN, GERDATAINICIO) (DEPARTAMENTO)

A mesma consulta em dois passos:

DEPT  (DNOME, DNUM, GERSSN, GERDATAINICIO) (DEPARTAMENTO) DEPT\_PROJ  PROJETO \* DEPT

Obs.: o atributo DNUM é chamado de atributo de Junção.

**Inner Join**

Originalmente o modelo relacional definiu o operador de junção apenas com as características apresentadas até aqui. Posteriormente, esse tipo de operador ficou conhecido como *Inner Join* e a operação de junção foi estendida para a definição de *Outer Join*.

Em um *Inner Join* existe a possibilidade de que algumas das linhas de uma ou ambas as tabelas de uma junção não façam parte da tabela resultante.

**Exemplo:**

### FUNCIONARIO PROJETO

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Cod\_Func** | **Nome** | **Cod\_Proj** |  | **Cod\_Proj** | **Projeto** |
| 301 | João | 01 | 01 | Alfa Centauro |
| 482 | Miguel | 02 | 02 | Betelgeuse |
| 127 | Roberto | 04 | 03 | Canopus |
| 185 | Dora | 02 | 04 | Orion |
| 079 | Esmeralda | 01 | 05 | Argos |
| 246 | Jaqueline | 03 | 06 | Vênus |
| 128 | Leandro | -- |  | | |
| 248 | Marcos | -- |

**FUNCIONARIO ** **PROJETO**

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Func** | **Nome** | **Cod\_Proj** | **Projeto** |
| 301 | João | 01 | Alfa Centauro |
| 482 | Miguel | 02 | Betelgeuse |
| 127 | Roberto | 04 | Orion |
| 185 | Dora | 02 | Betelgeuse |
| 079 | Esmeralda | 01 | Alfa Centauro |
| 246 | Jaqueline | 03 | Canopus |

De acordo com as regras de um ***Inner Join***, as informações sobre os funcionários Leandro, Marcos e os projetos Argos e Vênus não foram incluídas na tabela resultante.

**Outer Join**

O *Outer Join* é uma extensão do operador de junção utilizado para tratamento de ausência de informações.

A operação *Outer Join* mantém na tabela resultante TODAS as linhas de uma ou mais tabelas participantes da junção.

* + Utilizada quando é necessário incluir linhas de uma tabela onde os valores não têm correspondência com a outra.

***Outer Join*** = Resultado do ***Inner Join*** +

Linhas da tabela sem correspondência (na outra tabela do *join*);

* + As linhas da tabela que não tem correspondência são completadas com nulos.

**Exemplo:**

Linhas de FUNCIONÁRIO sem correspondência em PROJETO

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Func** | **Nome** | **Cod\_Proj** | **Projeto** |
| 128 | Leandro | -- | -- |
| 248 | Marcos | -- | -- |

Linhas de PROJETO sem correspondência em FUNCIONÁRIO

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Func** | **Nome** | **Cod\_Proj** | **Projeto** |
| -- | -- | 05 | Argos |
| -- | -- | 06 | Vênus |

Foram definidos três tipos de *Outer Join* entre duas tabelas:

* **Left Outer Join **
* **Right Outer Join **
* **Full Outer Join **

***Left Outer Join*** = *Inner Join* + linhas não participantes da tabela à esquerda

### FUNCIONARIO PROJETO

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Func** | **Nome** | **Cod\_Proj** | **Projeto** |
| 301 | João | 01 | Alfa Centauro |
| 482 | Miguel | 02 | Betelgeuse |
| 127 | Roberto | 04 | Orion |
| 185 | Dora | 02 | Betelgeuse |
| 079 | Esmeralda | 01 | Alfa Centauro |
| 246 | Jaqueline | 03 | Canopus |
| 128 | Leandro | -- | -- |
| 248 | Marcos | -- | -- |

Todas as linhas de FUNCIONARIO participam da tabela resultante

***Right Outer Join* =** *Inner Join* + linhas não participantes da tabela à direita

### FUNCIONARIO PROJETO

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Func** | **Nome** | **Cod\_Proj** | **Projeto** |
| 301 | João | 01 | Alfa Centauro |
| 482 | Miguel | 02 | Betelgeuse |
| 127 | Roberto | 04 | Orion |
| 185 | Dora | 02 | Betelgeuse |
| 079 | Esmeralda | 01 | Alfa Centauro |
| 246 | Jaqueline | 03 | Canopus |
| -- | -- | 05 | Argos |
| -- | -- | 06 | Vênus |

Todas as linhas de PROJETO participam da tabela resultante

***Full Outer Join*** = *Inner Join* + linhas não participantes de ambas tabelas

### FUNCIONARIO PROJETO

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod\_Func** | **Nome** | **Cod\_Proj** | **Projeto** |
| 301 | João | 01 | Alfa Centauro |
| 482 | Miguel | 02 | Betelgeuse |
| 127 | Roberto | 04 | Orion |
| 185 | Dora | 02 | Betelgeuse |
| 079 | Esmeralda | 01 | Alfa Centauro |
| 246 | Jaqueline | 03 | Canopus |
| 128 | Leandro | -- | -- |
| 248 | Marcos | -- | -- |
| -- | -- | 05 | Argos |
| -- | -- | 06 | Vênus |

Todas as linhas de FUNCIONARIO e PROJETO participam da tabela resultante

**Divisão A**  **B**

Divisão é a operação entre duas tabelas onde:

* Uma é o dividendo e outra o divisor
* As tabelas devem possuir uma ou mais colunas em comum
* A tabela dividendo (A) deve possuir colunas adicionais
* A tabela resultante é formada pelas colunas da tabela dividendo (A) que não fazem parte da tabela divisora (B).
* A tabela resultante contém apenas linhas da tabela dividendo (A) que satisfazem a comparação com todas as linhas da tabela divisora (B)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **DIVIDENDO** |  |  | **DIVISOR** |  | **RESULTADO** |
| **Cod\_Emp** | **Habilidade** |  | **Habilidade** |  | **Cod\_Emp** |
| 301 | 73 | 02 | 301 |
| 301 | 02 | 11 | 482 |
| 301 | 11 |  | | | |
| 482 | 02 |
| 482 | 11 |
| 127 | 73 |
| 127 | 43 |
| 127 | 02 |

Resultado: Lista de todos os empregados que possuem as habilidades „02‟ e „11‟

## Operações Aninhadas

Podemos construir expressões complexas aplicando operadores sobre o resultado da aplicação de outros operadores e assim por diante

Exemplo: Nome\_Cidade, UF(regiao= ‘SUL’ ( CIDADE  ESTADO) )

Nome\_Cidade, UF(CIDADE  (regiao= ‘SUL’ (ESTADO) ) )

**Renomeando uma relação**

 S(A1, A2, …,An) (R)

 S (R) Renomeia a relação “R” para “S”, sem renomear os atributos.

 S(A1, A2, …,An) (R) Renomeia a relação “R” para “S”, renomeando também os atributos resultantes para A1,A2,…,An na ordem da esquerda para a direita

A operação de renomear é útil quando necessitamos controlar os nomes dos atributos como, por exemplo, no produto cartesiano de uma relação com ela mesma.

# Exercícios

Considerando as relações a seguir, escreva as expressões da álgebra relacional para responder as consultas, mostrando ainda o resultado das mesmas.

**PRODUTO PC**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **fabricante** | **modelo** | **Tipo** |  | **modelo** | **velocidade** | **memoria** | **hd** | **cd** | **preco** |
| A | 1001 | Pc | 1001 | 133 | 16 | 1.6 | 6x | 1595 |
| A | 1002 | Pc | 1002 | 120 | 16 | 1.6 | 6x | 1399 |
| A | 1003 | Pc | 1003 | 166 | 24 | 2.5 | 6x | 1899 |
| B | 1004 | Pc | 1004 | 166 | 32 | 2.5 | 8x | 1999 |
| B | 1006 | Pc | 1005 | 166 | 16 | 2.0 | 8x | 1999 |
| B | 3002 | impressora | 1006 | 200 | 32 | 3.1 | 8x | 2099 |
| B | 3004 | impressora | 1007 | 200 | 32 | 3.2 | 8x | 2349 |
| C | 1005 | Pc | 1008 | 180 | 32 | 2.0 | 8x | 3699 |
| C | 1007 | Pc | 1009 | 200 | 32 | 2.5 | 8x | 2599 |
| D | 1008 | Pc | 1010 | 160 | 16 | 1.2 | 8x | 1495 |
| D | 1009 | Pc | **LAPTOP** | | | | | | |
| D | 1010 | Pc |
| D | 2001 | Laptop |  | **modelo** | **velocidade** | **memoria** | **hd** | **tela** | **preco** |
| D | 2002 | Laptop | 2001 | 100 | 20 | 1.10 | 9.5 | 1999 |
| D | 2003 | Laptop | 2002 | 117 | 12 | 0.75 | 11.3 | 2499 |
| D | 3001 | impressora | 2003 | 117 | 32 | 1.00 | 10.4 | 3599 |
| D | 3003 | impressora | 2004 | 133 | 16 | 1.10 | 11.2 | 3499 |
| E | 2004 | Laptop | 2005 | 133 | 16 | 1.00 | 11.3 | 2599 |
| E | 2008 | Laptop | 2006 | 120 | 8 | 0.81 | 12.1 | 1999 |
| F | 2005 | Laptop | 2007 | 150 | 16 | 1.35 | 12.1 | 4799 |
| G | 2006 | Laptop | 2008 | 120 | 16 | 1.10 | 12.1 | 2099 |
| G | 2007 | Laptop | **IMPRESSORA** | | | | | | |
| H | 3005 | impressora |
| I | 3006 | impressora |  | **modelo** | **cor** | **tipo** | **preço** |  |  |
|  | | | | 3001 | true | ink-jet | 275 |  |  |
| 3002 | true | ink-jet | 269 |  |  |
| 3003 | false | laser | 829 |  |  |
| 3004 | false | laser | 879 |  |  |
| 3005 | false | ink-jet | 180 |  |  |
| 3006 | true | dry | 470 |  |  |

*(fonte: A First Course in Database Systems. Ullmann & Widom. Prentice-Hall)*

//2190 + alt + x = ←

//03c0 + alt + x = π

//03c3 + alt + x = σ

//2A1D + alt + x = ⨝

//27D7 + alt + x =⟗

*//2260 + alt + x = ≠*

1. Quais os modelos de PC que têm velocidade maior ou igual a 150 ?

Π modelo (σ velocidade >= 150 (PC))

1. Que fabricantes fazem laptops com disco rígido maior ou igual a 1 GB ?

L ← σ hd >= 1.0 (LAPTOP)

Π fabricante (PRODUTO ⨝ PRODUTO.modelo = L.modelo L)

1. Encontre os números dos modelos e preços de todos os produtos (de qualquer tipo) produzidos pelo fabricante B.

PRODUTO\_PC = PRODUTO ⨝ PRODUTO.modelo = PC.modelo PC

PRODUTO\_LAPTOP = PRODUTO ⨝ PRODUTO.modelo = LAPTOP.modelo LAPTOP

PRODUTO\_IMPRESSORA = PRODUTO ⨝ PRODUTO.modelo = IMPRESSORA.modelo IMPRESSORA

Π modelo, preco (σ fabricante = ‘B’ (PRODUTO\_PC ∪ PRODUTO\_LAPTOP ∪ PRODUTO\_IMPRESSORA)

1. Encontre os números dos modelos de todas as impressoras laser colorida.

Π modelo (σ tipo = ‘laser’ AND cor = ‘true’ (PRODUTO ⨝ PRODUTO.modelo = IMPRESSORA.modelo IMPRESSORA))

1. Encontre todos os fabricantes que vendem laptops, mas não vendem PCs.

F1 ← σ tipo = ‘Laptop’ (PRODUTO)

F2 ← σ tipo = ‘Pc’ (PRODUTO)

Π fabricante (F1 – F2)

1. Encontre os tamanhos de discos rígidos que ocorrem em dois ou mais PCs.

P1 ← PC

P2 ← PC

Π P1.hd (σ P1.hd = P2.hd (P1 x P2)

1. Encontre os pares de modelos de PCs que apresentam a mesma velocidade e memória (um par somente deve aparecer uma vez, i.e., se deve listar (i,j) mas não (j,i)).

P1 ← PC

P2 ← PC

π P1.modelo, P2.modelo (σ P1.modelo < P2.modelo ( P1 ⨝ P1.velocidade = P2.velocidade AND P1.memoria = P2.memoria P2))

1. Encontre os fabricantes de ao menos dois computadores diferentes (PCs ou laptops) com velocidade maior ou igual a 133.

PRODUTO\_PC = PRODUTO ⨝ Produto.modelo = Pc.modelo PC

PRODUTO\_LAPTOP = PRODUTO ⨝ Produto.modelo = Laptop.modelo LAPTOP

T ← σ velocidade >= 133 (PRODUTO\_PC)

L ← σ velocidade >= 133 (PRODUTO\_LAPTOP)

Π fabricante (σ COUNT(modelo) >= 2 (T ∪ L)

T←(σvelocidade≥133​(PRODUTO⨝PRODUTO.modelo=PC.modelo​PC))∪(σvelocidade≥133​(PRODUTO⨝PRODUTO.modelo=LAPTOP.modelo​LAPTOP))

T1←ρT1​(T) T2←ρT2(T)

T2 ← ρ\_{T2}(T)T2←ρT2​(T)

R←σT1.fabricante=T2.fabricante∧T1.modelo=T2.modelo​(T1×T2)

πT1.fabricante​(R)

1. Encontre o(s) fabricante(s) do computador com maior velocidade disponível.

V ← max(velocidade) (PC ∪ LAPTOP)

Π fabricante (σ velocidade = V (PRODUTO ⨝ (PC ∪ LAPTOP)))

C←(PC∪LAPTOP)

C1←ρC1​(C),C2←ρC2​(C)

Menores←πC1.modelo​(σC1.velocidade<C2.velocidade​(C1×C2))

Maiores←C−(σmodelo∈Menores​(C))

πfabricante​(PRODUTO⨝Maiores)

1. Encontre os fabricantes de PCs com ao menos três velocidades diferentes.

P ← PRODUTO ⨝ PRODUTO.modelo = PC.modelo PC

Π fabricante (σ COUNT(velocidade) >= 3 (P))

P←PRODUTO⨝PRODUTO.modelo=PC.modelo​PC

P1←ρP1​(P),

P2←ρP2​(P),

P3←ρP3​(P)

R←σP1.fabricante=P2.fabricante∧P1.fabricante=P3.fabricante∧P1.modelo=P2.modelo∧P1.modelo=P3.modelo∧P2.modelo=P3.modelo∧P1.velocidade=P2.velocidade∧P1.velocidade=P3.velocidade∧P2.velocidade=P3.velocidade​(P1×P2×P3)

πP1.fabricante​(R)

1. Encontre os fabricantes que vendem exatamente três modelos diferentes de PCs.

V ← PRODUTO ⨝ PRODUTO.modelo = PC.modelo PC

Π fabricante (σ COUNT(modelo) = 3 (V))

P←PRODUTO⨝PRODUTO.modelo=PC.modelo​PC

P1,P2,P3,P4

F3←πP1.fabricante​(σP1.fabricante=P2.fabricante∧P1.fabricante=P3.fabricante∧P1.modelo=P2.modelo∧P1.modelo=P3.modelo∧P2.modelo=P3.modelo​(P1×P2×P3))

F4←πP1.fabricante​(σP1.fabricante=P2.fabricante∧P1.fabricante=P3.fabricante∧P1.fabricante=P4.fabricante∧P1.modelo=P2.modelo∧P1.modelo=P3.modelo∧P1.modelo=P4.modelo∧P2.modelo=P3.modelo∧P2.modelo=P4.modelo∧P3.modelo=P4.modelo​(P1×P2×P3×P

F3−F4