Universidade Federal da Fronteira Sul - UFFS
Campus Chapecó
Ciência da Computação
Banco de Dados I
Prof.: Denio Duarte

Solução A3

1. Dado o seguinte esquema relacional:
pessoa *(cpf, nome, dtnasc, cpfpai(pessoa))* ***tipoitens****(cod, descr)*
IRPF*(cpf (pessoa), ano, vlpg, vlrest)*
itensIRPF*(cpf(IRPF), ano(IRPF) tpitem(tipoitens), vldecl, tribut)*

   Resolva as consultas abaixo em álgebra relacional:

   (a) Retorne os nomes das pessoas e nomes dos itens que foram declarados em 2017.
   É necessário fazer todo o caminho para encontrar os itens declarados por um pessoa, ou seja, partir de pessoa, fazer junção com *IRPF*, *itensIRPF* e *tipoitens*.
   $\pi_{pessoa.nome,tipoitens.nome}(pessoa \bowtie \sigma_{ano=2017}(IRPF) \bowtie itensIRPF \bowtie_{tpitem=cod} tipoitens)$
   OU $\pi_{pessoa.nome,tipoitens.nome}(\sigma_{ano=2017}(pessoa \bowtie IRPF \bowtie itensIRPF \bowtie_{tpitem=cod} tipoitens))$

   (b) Retorne os nomes dos declarantes nascidos em 1971 com os nomes, caso existirem, dos seus respectivos pais.
   Nesta consulta, a tabela *pessoa* tem que ser utilizada duas vezes: uma para o declarante e outra para encontrarmos o pai do declarante. Para evitar renomear a tabela dentro da mesma consulta, criei uma tabela temporária chama *pai* para auxiliar. Perceba que que o atributo de *join* (*cpfpai*) é opcional, ou seja, nem todo contribuinte declarou um pai, assim a consulta deve ser do tipo *outer left*. O *outer join* permite que todos os contribuintes nascidos em 1971 sejam retornados e, caso o atributo *cpfpai* não seja nulo, retorna os nomes dos pais.
   $pai \leftarrow \pi_{nome,cpf}(pessoa)$
   $\pi_{pessoa.nome,pai.nome}(\sigma_{dtnasc>=01/01/1971 \wedge dtnasc<=31/12/2017}(pessoa)) \rtimes pai)$

   (c) Retorne o nome da pessoa que teve o maior valor restituído (*vlrest*) em 2012.
   Opção 1:
   $\pi_{nome}(\sigma_{vlrest=(\mathcal{G}max(vlrest)(\sigma_{ano=2012}(IRPF)))}(pessoal \bowtie \sigma_{ano=2012}(IRPF)))$
   Opção 2: Cria uma tabela que armazena apenas o maior valor e depois faz um *join* pelo valor.
   $Maior \leftarrow \rho_{(maxval)}(\mathcal{G}max(vlrest)(\sigma_{ano=2012}(IRPF)))$
   $\pi_{nome}(pessoa \bowtie \sigma_{ano=2012}(IRPF) \bowtie_{vlrest=maxval} Maior)$

   (d) Retorne o ano que teve o maior número de declarações de imposto de renda.
   Conta o número de declarações agrupando pelo ano. Cria a tabela *Qtdecl*.
   $Qtdecl \leftarrow \rho_{(ano,qtd)}(ano\mathcal{G}count(*)(IRPF))$
   Retorna o maior valor da tabela *Qtdecl*, criando a tabela *Mxdecl* com este valor.
   $Mxdecl \leftarrow \rho_{(max)}(\mathcal{G}max(qtd)(Qtdecl))$
   Faz *join* entre *Mxdecl* e *Qtdecl* pelos valores e depois, com o *ano* em *Qtdecl*, faz o *join* com *IRPF*.
   $\pi_{ano}(Mxdecl \bowtie_{max=qtd} Qtdecl \bowtie IRPF)$

   (e) Retorne o nome e o CPF do contribuinte que declarou todos os itens de imposto de renda cadastrados.
   Tenho que fazer duas relações: uma com o nome, CPF e código dos itens declarados pelo contribuinte; e outra com os códigos de todos os itens cadastrados. É necessário se preocupar com os nomes dos atributos para que a divisão funcione.
   $\pi_{nome,cpf,tpitem}(pessoa \bowtie IRPF \bowtie itensIRPF) \div \rho_{(tpitem)}(\pi_{cod}(tipoitens))$
   Assim, o *tpitem* da primeira consulta é utilizado para comparar com todos da segunda consulta. Aquelas duplas *nome* e *cpf* que possuem todos *tpitem* são retornadas.

2. Considere a relação $Orders(Order, Product, Customer, Address, Qty, UnitPrice)$ e as seguintes DFs:
   $Order \rightarrow Customer\ Address\ Qty$
   $Customer \rightarrow Address$

$Product \rightarrow UnitPrice$
Os atributos *Order* e *Product* têm que fazer parte da chave, pois não aparecem à direita de nenhum dependência funcional (DF) e portanto, o algoritmo de fechamento não consegue colocá-los no conjunto.
A primeira DF é decomposta para o algoritmo de fechamento, e na primeira passada, os atributos *Customer Address Qty* são adicionados ao conjunto inicial {Order, Product}, resultando em {Order, Product, Customer, Address, Qty}, como Address de Customer já está no conjunto, ele não é adicionado, finalmente, UnitPrice é inserido por conta da DF de Product. O conjunto fechamento de $\{Order, Product\}^+$=$\{Order, Product, Customer, Address, Qty, UnitPrice\}$, ou seja, todos os atributos da tabela *Order*. Então uma das chaves é Order Product. Não é possível inferir nenhuma outra chave que não contenha ambos os atributos. Então as DF são:

$Order \rightarrow Customer\ Address\ Qty$
$Customer \rightarrow Address$
$Product \rightarrow UnitPrice$
$Order\ Product \rightarrow Customer\ Address\ Qty\ UnitPrice$

Baseado no novo conjunto de dependências, vamos normalizar o tabela.
1FN: não existem atributos multivalorados assim, Order atende esta forma normal.
2FN: UnitPrice (com a dependência $Product \rightarrow UnitPrice$) e Customer Address e Qty com a dependência ($Order \rightarrow Customer\ Address\ Qty$) dependem parcialmente da chave da tabela, ou seja, Order Product.
Vamos eliminar a DF $Product \rightarrow UnitPrice$ criando a tabela Products: Products(Product, UnitPrice) respeitando $Product \rightarrow UnitPrice$.
Agora, vamos eliminar $Order \rightarrow Customer\ Address\ Qty$. Para tal, uma nova tabela é criada:
OrderCust(Order, Customer, Address, Qty)
Após a aplicação da 2FN, o esquema do banco e DF ficou assim:

Orders(Order, Product) e DF: $Order\ Product \rightarrow Order\ Product$
Products(Product, UnitPrice) e DF: $Product \rightarrow UnitPrice$
OrderCust(Order, Customer, Address, Qty) e DF: $Order \rightarrow Customer\ Address\ Qty$ e $Customer \rightarrow Address$

BCNF: a DF $Customer \rightarrow Address$ é válida, mas Customer não é chave. Criamos a tabela Customers e transferimos a DF para lá: Customers(Customer, Address).
O esquema resultante é:

Orders(Order, Product) e DF: $Order\ Product \rightarrow Order\ Product$
Products(Product, UnitPrice) e DF: $Product \rightarrow UnitPrice$
OrderCust(Order, Customer, Qty) e DF: $Order \rightarrow Customer\ Qty$
Customers(Customer, Address) e DF $Customer \rightarrow Address$

Comentários: o BD resultante ficou meio estranho, mas seguindo a risca as regras de normalização, essa é a solução.