INSTITUTO MATEMÁTICA E ESTATÍSTICA

Exercícios de Revisão – Prova 2 – BD I

Seja o esquema relacional da base de dados simplificada de uma Empresa. As chaves primárias respectivas estão sublinhadas.

EMPREGADO (Ident, Nome, Sal, Endereco, Sexo, DataNasc, DepNum, SuperIdent)
DEPARTAMENTO (Num, Nome, IdentGer)
PROJETO (Num, Nome, Local, DepNum)
TRABALHA_NO (IdentEmp, ProjNum, HRS)
DEPENDENTE (DNome, Sexo, DataNasc, Parentesco, IdentEmp)
DEPLOC (DepNum, Local)

As seguintes características são representadas na base de dados. A empresa é organizada em departamentos, cada qual tendo um nome, um número de identificação e um empregado responsável (gerente) pelo mesmo. Guarda-se como informação a data a partir da qual o empregado assumiu a gerência do departamento. Cada departamento pode estar presente em diversas localidades do país.
Um departamento controla certo número de projetos, cada qual tendo um nome, um número de identificação e uma localidade única. Os departamentos têm vários empregados, para os quais guarda-se os respectivos nomes, número de matrícula na empresa, endereço residencial, sexo, data de nascimento e salário. Todo empregado tem um empregado que é seu superior hierárquico (supervisor direto) dentro do departamento, informação que também deve ser mantida no banco de dados.
Um empregado é alocado a um único departamento mas pode estar alocado a mais de um projeto, estes não necessariamente controlados pelo mesmo departamento. Controla-se o total de horas semanais em que um empregado trabalha em cada projeto. Para fim de controle de seguro de saúde de cada empregado, devemos manter informações dos nomes, sexo, data de nascimento e grau de parentesco de seus dependentes.

Utilizar a **Álgebra Relacional** para definir as seguintes consultas:

1. Quais empregados (identidade, nome, endereço e sexo) não trabalham em projeto algum?

$$EmpComProj \leftarrow \Pi_{(ident)}(TRABALHA\_NO \bowtie_{(identEmp = Ident)} EMPREGADO)$$

$$IdentSemProj \leftarrow \Pi_{(ident)}(EMPREGADO) - EmpComProj$$

$$Resposta \leftarrow \Pi_{(ident,nome,endereco,sexo)}(IdentSemProj \bowtie EMPREGADO)$$

2. Apresente os nomes dos projetos onde trabalham empregados com salário superior a R$ 1.000,00.

$$Sal1000 \leftarrow \Pi_{(ident)}(\sigma_{(sal > 1000)}(EMPREGADO))$$

$$TrabSal1000 \leftarrow (TRABALHA\_NO \bowtie_{(identEmp = ident)} Sal1000)$$

$$ProjSal1000 \leftarrow (TrabSal1000 \bowtie_{(projNum = num)} PROJETO)$$

$$Resposta \leftarrow \Pi_{(nome)}(ProjSal1000)$$

Lista de exercícios adaptada do material do prof. Sérgio Lifschitz (Puc-Rio)

3. Qual o valor total pago em salários para os empregados lotados em departamentos localizados em 'Belo Horizonte'?

$$TrabEmp \leftarrow \Pi_{(ident,\ sal,\ projNum)}(TRABALHA\_NO \bowtie_{(ident\ =\ identEmp)} EMPREGADO)$$

$$ProjEmp \leftarrow \Pi_{(ident,\ sal,\ projNum,local)}(PROJETO \bowtie_{(num\ =\ projNum)} TrabEmp)$$

$$ProjBH \leftarrow \sigma_{(local\ =\ 'Belo\ Horizonte')}(ProjEmp)$$

$$Final \leftarrow \mathcal{G}_{(sum(sal))}(ProjBH)$$

4. Liste o nome e a identidade dos empregados que não possuem dependentes do sexo feminino.

$$Dep \leftarrow \Pi_{(sexo,\ identEmp)}(Dependente)$$

$$DepFem \leftarrow \sigma_{(sexo\ =\ 'feminino')}(Dep)$$

$$EMPFem \leftarrow \Pi_{(Nome,\ Ident)}(DepFem \bowtie_{(IdentEmp\ =\ Ident)} EMPREGADO)$$

$$EMP \leftarrow \Pi_{(Nome,\ Ident)}(EMPREGADO)$$

$$Resposta \leftarrow \Pi_{(Nome,\ Ident)}(EMP - EMPFem)$$

5. Quais são as identidades dos empregados que trabalham em projetos coordenados por departamentos distintos daqueles onde estão alocados?

$$TrabEmp \leftarrow (TRABALHA\_NO \bowtie_{(identEmp\ =\ ident)} EMPREGADO)$$

$$Proj \leftarrow \Pi_{(num,depNum)}(PROJETO)$$

$$\delta_{depNum->depProj}(Proj)$$

$$ProjEmp \leftarrow (Proj \bowtie_{(num\ =\ projNum)} TrabEmp)$$

$$ProjEmp \leftarrow \sigma_{(depProj\ !=\ depNum)}(ProjEmp)$$

$$Resposta \leftarrow \Pi_{(ident)}(ProjEmp)$$

6. Sabendo que o pai de 'Gregor Samsa' trabalha em algum departamento da empresa, quais seus possíveis locais de trabalho?

$$Gregor \leftarrow \sigma_{(Nome = 'Gregor\ Samsa')}(DEPENDENTE)$$

$$Pai \leftarrow \Pi_{(Ident)}(EMPREGADO \bowtie_{(Ident = IdentEmp)} Gregor)$$

$$PaiTrab \leftarrow (Pai \bowtie_{(Ident = IdentEmp)} TRABALHA\_NO)$$

$$Resposta \leftarrow \Pi_{(Local)}(PaiTrab \bowtie_{(ProjNum = Num)} PROJETO)$$

7. Listar o nome e o salário de todos os funcionários do projeto 'BEPiD'.

$$Trab \leftarrow \Pi_{(Ident, emp, projnum)}(TRABALHA\_NO)$$

$$Proj \leftarrow \Pi_{(Num, nome)}(PROJETO)$$

$$ProjTrab \leftarrow (Proj \bowtie_{(Num = ProjNum)} Trab)$$

$$ProjBepid \leftarrow \Pi_{(IdentEmp)}(\sigma_{(Nome = 'BEPiD')}(ProjTrab))$$

$$Resposta \leftarrow \Pi_{(Nome, Salário)}(ProjBepid \bowtie_{(IdentEmp = Ident)} EMPREGADO)$$

Utilizando **SQL**, solucione as questões a seguir:

8. Quais nomes dos dependentes que têm o nome igual ao do empregado do qual são dependentes?

```
SELECT DNOME
FROM EMPREGADO E, DEPENDENTE D
WHERE E.IDENT = D.IDENTEMP
      AND E.NOME = D.DNOME
```

9. Para cada empregado, mostrar seu nome e o nome do seu superior imediato.

```
SELECT   E.NOME as NomeSuper, S.NOME as NomeSub
FROM     EMPREGADO E, EMPREGADO S
WHERE    E.SUPERIDENT=S.IDENT
```

10. Listar os valores de salários pagos aos empregados da empresa.

```
SELECT   SAL
FROM     EMPREGADO
```

11. Apresentar os nomes de todos os empregados que não tem superintendente.

```
SELECT   NOME
FROM     EMPREGADO
WHERE    SUPERIDENT IS NULL
```

12. Listar todos os empregados que moram em endereços cujo nome contém "Salvador".

```
SELECT   *
FROM     EMPREGADO
WHERE    ENDERECO LIKE '%Salvador%'
```

13. Apresentar o resultado dos salários dos empregados que trabalham no projeto "Reengenharia" caso fosse dado um aumento de 10%.

```
SELECT   EMPREGADO.NOME, 1.1*SAL AS SAL
FROM     EMPREGADO, TRABALHANO, PROJETO
WHERE EMPREGADO.IDENT = TRABALHANO.IDENTEMP AND
      TRABALHANO.PROJNUM = PROJETO.NUM AND
      PROJETO.NOME = 'Reengenharia';
```

14. Crie as tabelas "Empregado" e "Projeto", definindo suas respectivas chaves, durante a sua criação e considerando que a tabela Departamento já está criada. Todas as colunas devem ser obrigatórias.

```
CREATE TABLE EMPREGADO
(
      IDENT            NUMERIC(10)      NOT NULL      PRIMARY KEY,
      NOME             VARCHAR(50)      NOT NULL,
      SAL              NUMERIC(11)      NOT NULL,
      ENDERECO         VARCHAR(50)      NOT NULL,
      SEXO             CHAR(1)          NOT NULL,
      DATANASC         DATE             NOT NULL,
      DEPNUM           NUMERIC(5)       NOT NULL      REFERENCES DEPARTAMENTO(NUM),
      SUPERIDENT       NUMERIC(10)      NOT NULL
);
CREATE TABLE PROJETO
(
      NUM              NUMERIC(5)       NOT NULL      PRIMARY KEY,
      NOME             VARCHAR(50)      NOT NULL,
      LOCAL            VARCHAR(50)      NOT NULL,
      DEPNUM           NUMERIC(5)       NOT NULL      REFERENCES DEPARTAMENTO(NUM)
);
```

15. Crie a tabela "TRABALHA_NO", definido as suas chaves primária e estrangeira.

```
CREATE TABLE TRABALHA_NO
(
      IDENTEMP         NUMERIC(10)      NOT NULL      REFERENCES EMPREGADO(IDENT) ,
      PROJNUM          NUMERIC(5)       NOT NULL      REFERENCES PROJETO(NUM),
      HRS              VARCHAR(5)       NOT NULL,
      CONSTRAINT TRABALHA_PK PRIMARY KEY (IDENTEMP, PROJNUM)
);
```

16. Altere a tabela "EMPREGADO" para contemplar a restrição de que a data de nascimento não pode ser maior do que a data atual – 18 (menores de 18 anos). A função que retorna a data atual no PostgreSQL é a NOW() e para subtrair os anos, o postgresql usa a função INTERVAL '<valor> <YEARS>'.

```
ALTER TABLE EMPREGADO
ADD CONSTRAINT CK_DTNASC
CHECK (DATANASC < (NOW() – INTERVAL '18 YEARS'));
```

17. Altere a tabela "PROJETO" para que o seu local passe a ser uma coluna opcional.

```
ALTER TABLE PROJETO
ALTER COLUMN LOCAL DROP NOT NULL;
```

18. Crie um índice sobre o número do departamento de projeto.

```
CREATE INDEX IN_DEPNUMPROJ
ON PROJETO (DEPNUM);
```

19. Agora remova o índice.

```
DROP INDEX IN_DEPNUMPROJ;
```

20. Quais os nomes dos empregados e os números de departamento dos quais eles são gerentes, se o forem?

```
SELECT   Distinct E.nome, D.num
FROM     Departamento D RIGHT OUTER JOIN Empregado E
ON       E.Ident = D.IdentGer;
```

21. Listar os nomes dos empregados, assim como os departamentos onde trabalham, que ganham mais do que qualquer empregado do departamento de "Pesquisa"

```
SELECT   E.nome AS NomeEmp, D.nome AS NomeDepto
FROM     empregado E, departamento D
WHERE    E.depnum = D.num AND
                 sal >ALL (SELECT sal
                           FROM empregado E, departamento D
                           WHERE E.depnum = D.num AND
                           D.nome = 'Pesquisa');
```

22. Listar os nomes dos empregados que trabalham o mesmo total de horas em algum projeto em que o empregado Caetano Veloso trabalha.

```
SELECT   DISTINCT Nome
FROM     Trabalhano T1, Empregado E1
WHERE    T1.IdentEmp = E1.Ident
AND      E1.Nome <>  'Caetano Veloso'
AND EXISTS
(SELECT ProjNum, HRS
                  FROM Trabalhano T2, Empregado E2
                  WHERE T2.IdentEmp = E2.Ident
                  AND E2.Nome = 'Caetano Veloso'
                  AND T2.ProjNum = T1.ProjNum
                  AND T2.HRS = T1.HRS);
```

23. Quais os nomes dos empregados que ganham os 3 maiores salários da empresa?

```
SELECT ident, nome, sal
FROM empregado e1
WHERE 3 > (SELECT count (distinct SAL)
                    FROM   empregado e2
                    WHERE  e2.sal > e1.sal);
```

Obs.: Pode-se substituir 3 por N caso sejam os N maiores salários!

24. Para cada projeto em que trabalham mais de 2 empregados, listar número e nome dos projetos, e o total de empregados que trabalham no projeto.

```
SELECT p.num, p.nome, count(t.identemp)
FROM projeto p, trabalhano t
WHERE p.num = t.projNum
GROUP BY p.num, p.nome
HAVING count(t.identemp) > 2
```

25. Para cada departamento que tenha mais de 2 empregados, listar o número do departamento e a quantidade de empregados que ganham mais de R$ 30.000,00

```
SELECT d.num, count(e.ident)
FROM departamento d, empregado e
WHERE e.depnum = d.num
AND e.sal > 30000
GROUP BY d.num
HAVING count(e.ident) > 2
```

26. Listar os nomes dos empregados que trabalham em TODOS os projetos.

```
select nome from empregado e
where not exists
(select * from projeto p
where not exists
(select * from trabalhano t
where e.ident = t.identemp
and t.projnum = p.num))
OU
select nome
        from empregado e
        where e.ident IN
                (select t.identemp
                from trabalhano t
                group by identemp
                having count(*) =(select count(*) from projeto));
```

27. Para cada projeto em que trabalham mais de 2 empregados, listar número e nome dos projetos, e o total de empregados que trabalham no projeto.

```
SELECT p.num, p.nome, count(*)
FROM projeto p, trabalhano t
WHERE p.num = t.projnum
GROUP BY p.num, p.nome
HAVING count(*) > 2;
```

28. Realize inserções em cada uma das tabelas, respeitando as integridades. Atenção para a ordem dos comandos a serem executados.

```
INSERT INTO EMPREGADO (Ident, Nome, Sal, End, Sexo, DataNasc, DepNum, SuperIdent) VALUES (12345678, 'Joao da Silva', 1000.00, 'Rua da Quitanda, 1', 'Masculino', '01/01/2000', null, null);
INSERT INTO DEPARTAMENTO (Num, Nome, IdentGer, DataIni) VALUES (1, 'Financeiro', 12345678, '01/01/2015');
INSERT INTO PROJETO (Num, Nome, Local, DepNum) VALUES (1, 'Projeto Universitário', 'São Paulo', 1);
INSERT INTO TRABALHANO (IdentEmp, ProjNum, HRS) VALUES (12345678, 1, 20);
INSERT INTO DEPENDENTE (IdentEmp, Nome, Sexo, DataNasc, Parentesco) VALUES (12345678, 'Paulo José', 'Masculino', '01/01/2014', 'Filho');
INSERT INTO DEPLOC (DepNum, Local) VALUES (1, 'São Paulo');
```

29. Altere o local de todos os departamentos para Rio de Janeiro.

```
UPDATE DEPLOC
SET local = 'Rio de Janeiro';
```

30. Altere o valor do atributo Sal para mais 5% de todos os empregados que trabalham em mais de 2 projetos.

```
UPDATE EMPREGADO
SET sal = sal * 1.05
WHERE (Ident) IN
    (SELECT Ident FROM empregado e, trabalhaNo t
    WHERE e.Ident = t.IdentEmp
    GROUP BY Ident
    HAVING count(*) > 2);
```

31. Exclua todos os dependentes.

```
DELETE FROM dependente;
```

32. Exclua todos os empregados que não trabalham em nenhum projeto.

```
DELETE FROM empregado
WHERE (Ident) NOT IN
    (SELECT Ident FROM trabalhaNo);
```

Seja a base de dados CARROS com as seguintes relações (chaves primárias sublinhadas e chaves estrangeiras em itálico referenciando atributos de mesmo nome em diferentes tabelas):

```
AUTOMOVEIS (Codigo, Fabricante, Modelo, Ano, País,
Preco_tabela)
REVENDEDORAS (CGC, Nome, Proprietário, Cidade, Estado)
CONSUMIDORES (CPF, Nome, Sobrenome, Cidade, Estado)
NEGOCIOS (Comprador, Revenda, CodAuto, Ano, Data, Preço)
GARAGENS (CGCRevenda, CodAuto, AnoAuto, Quantidade)
```

Utilizar a **Álgebra Relacional** para definir as seguintes consultas:

33. Listar os modelos dos carros que podem ser adquiridos na revendedora 'Alpha'.

$$GarRev \leftarrow (REVENDEDORAS \bowtie_{(GGC = CGCRevenda)} GARAGENS)$$

$$GarRev \leftarrow \sigma_{(nome = 'Alpha')} (GarRev)$$

$$Final \leftarrow (AUTOMOVEIS \bowtie_{(Codigo = CODAuto)} GARAGENS)$$

$$Final \leftarrow \Pi_{(Modelo)} (Final)$$

34. Listar os nomes dos consumidores que compraram algum carro com o preço de tabela superior a R$ 30.000, independente do valor que foi efetivamente pago.

$$ConsNegoc \leftarrow (CONSUMIDORES \bowtie_{(CPF = Comprador)} NEGOCIOS)$$

$$ValorNegoc \leftarrow AUTOMOVEIS \bowtie ConsNegoc$$

$$Sup30000 \leftarrow \sigma_{(Preco\_Tabela > 30000)} (ValorNegoc)$$

$$Final \leftarrow \Pi_{(Nome)} (Sup30000)$$

35. Quais os nomes das revendedoras que possuem carros do modelo 'Tucson'?

$$AutoGar \leftarrow (AUTOMOVEIS \bowtie_{(Codigo = CODAuto)} GARAGENS)$$

$$Tucson \leftarrow \sigma_{(Modelo = 'Tucson')} (AutoGar)$$

$$Tucson \leftarrow \Pi_{(CGCRevenda)} (Tucson)$$

$$Final \leftarrow (REVENDEDORAS \bowtie_{(CGC = CGCRevenda)} Tucson)$$

$$Resposta \leftarrow \Pi_{(nome)} (Final)$$

36. Listar os nomes de todas as revendedoras que vendam carros de todos os anos que algum dos carros que estão à venda na revendedora 'Venda à Jato'

$$RevJato \leftarrow \sigma_{(nome = 'Venda a Jato')} (Revendedoras)$$

$$RevJato \leftarrow (RevJato \bowtie_{(CGC = CGCRevenda)} GARAGENS)$$

$$AnoAuto \leftarrow \Pi_{(anoauto)} (GARAGENS)$$

$$GarRev \leftarrow (REVENDEDORAS \bowtie_{(CGC = CGCRevenda)} GARAGENS)$$

$$Final \leftarrow \Pi_{(nome)} (GarRev \div AnoAuto)$$

37. Quais os nomes das revendedoras que vendem todos os modelos existentes do fabricante 'Renault'?

$$AutoGar \leftarrow (AUTOMOVEIS \bowtie_{(Codigo = CodAuto)} GARAGENS)$$

$$Renault \leftarrow \sigma_{(Fabricante = 'Renault')} (AutoGar)$$

$$RevRen \leftarrow (REVENDEDORAS \bowtie_{(CGC = CGCRevenda)} Renault)$$

$$ModelosRenault \leftarrow \Pi_{(Modelo)} (\sigma_{(Fabricante = 'Renault')} (AUTOMOVEIS))$$

$$Tucson \leftarrow \Pi_{(Nome)} (RevRen \div ModelosRenault)$$

38. Listar o nome de todos os consumidores que compraram um 'Peugeot 206' na revendedora 'Ago Barra I'.

$$Peu206 \leftarrow \sigma_{(Fabricante = 'Peugeot',\ Modelo='206')} (AUTOMOVEIS)$$

$$RevNeg \leftarrow (REVENDEDORAS \bowtie_{(CGC = Revenda)} NEGOCIOS)$$

$$AgoBarra \leftarrow \sigma_{(nome='Ago Barra I')} (RevNeg)$$

$$PeuAgo \leftarrow (AgoBarra \bowtie_{(Codigo = CodAuto)} Peu206)$$

$$Cons \leftarrow \delta_{nome->nomeCon}(Consumidores)$$

$$Resposta \leftarrow \Pi_{(nomeCon)} (PeuAgo \bowtie_{(Comprador = CPF)} Cons)$$

Utilizar a linguagem **SQL** para definir as seguintes consultas:

39. Listar os nomes dos fabricantes dos automóveis na base de dados e os respectivos países de fabricação originalmente.

```
SELECT   distinct Fabricante, Pais
FROM     Automoveis;
```

40. Listar os estados onde se vende o modelo *Elba*, do fabricante *Fiat*.

```
SELECT   distinct Estado
FROM     Automoveis A, Revendedoras R, Garagens G
WHERE    A.Codigo = G.CodAuto
         AND      A.Ano = G.AnoAuto
         AND      G.CGCRevenda = R.CGC
         AND      A.Modelo = 'Elba'
         AND      A.Fabricante = 'Fiat';
```

41. Quais revendedoras não vendem automóveis de origem *francesa*?

```
SELECT   CGC, Nome, Cidade, Estado
FROM     Revendedoras
WHERE    CGC not in
         (SELECT   G.CGCRevenda
                   FROM     Automoveis A, Garagens G
                   WHERE    A.Codigo = G.CodAuto
                            A.ano = G.AnoAuto
                            AND A.Pais = 'Franca');
```

42. Listar código, fabricante, modelo e ano dos carros que são colocados a venda em pelo menos uma revendedora.

Aqui pode ser solução direta por INNER JOIN envolvendo Automoveis e Garagens ou para exemplificar uso do EXISTS:

```
SELECT   Codigo, Fabricante, Modelo, Ano
FROM     Automoveis A
WHERE    exists
         (SELECT   *
         FROM     Garagens G
         WHERE    G.CodAuto = A.Codigo
                  AND G.AnoAuto = A.Ano)
```

43. Quais são os nomes e sobrenomes dos consumidores que ainda não compraram carro algum?

```
SELECT   nome, sobrenome
FROM     consumidores
WHERE    cpf NOT IN
         (SELECT cpf
                   FROM     negocios)
```

44. Quais são as revendedoras de mesmo proprietário presentes em mais de um estado?

```
SELECT   cgc, nome, proprietario
FROM     revendedoras R1
WHERE    EXISTS
         (SELECT   *
         FROM     revendedoras R2
         WHERE    R1.proprietario = R2.proprietario
                  AND R1.estado <> R2.estado )
```

OU neste caso também ok:

```
(SELECT   cgc, nome, proprietario
FROM      revendedoras R1, Revendedoras R2
WHERE     R1.proprietario = R2.proprietario
      AND R1.estado <> R2.estado )

OU

SELECT cgc, nome, proprietario
FROM revendedoras
WHERE proprietario IN
          (SELECT proprietario
          FROM revendedoras
          GROUP BY proprietario
          HAVING COUNT(DISTINCT estado) > 1);
```

45. Listar o CGC e o nome das revendedoras que venderam carros japoneses ou americanos, fabricados em 1995, por valor até 30% acima do preço de tabela.

```
SELECT    DISTINCT r.cgc, r.nome
FROM      revendedoras R, negocios N, automoveis A
WHERE     R.cgc = N.Revenda
          AND A.codigo = N.CodAuto
          AND A.ano = N.Ano
          AND A.pais IN ('Japao', 'EUA')
          AND A.ano = 95
          AND N.preco <= (1.3 * A.preco_tabela)
```

46. Qual o automóvel (fabricante, modelo e ano) mais barato à venda nas revendedoras?

```
SELECT  fabricante, modelo, a.ano
FROM  Automoveis A
   INNER JOIN Garagens G on A.codigo = G.codAuto  and A.ano = G.anoAuto
WHERE Preco_tabela in
(SELECT min(Preco_tabela)
 FROM Automoveis A2
 INNER JOIN Garagens G2 on A2.codigo = G2.codAuto  and A2.ano = G2.anoAuto)

OU

SELECT  fabricante, modelo, ano
FROM  Automoveis A1
WHERE  Preco_tabela in
  (SELECT min(Preco_tabela)
   FROM Automoveis A2
     INNER JOIN Garagens G on A2.codigo = G.codAuto  and A2.ano = G.anoAuto)
AND exists
   (SELECT *
   FROM Garagens G
   WHERE A1.codigo = G.codAuto  and A1.ano = G.anoAuto)
```

47. Para os consumidores que compraram dois ou mais carros, listar nome, sobrenome, quantidade de carros comprados e valor total pago.

```
SELECT c.cpf, c.nome, c.sobrenome, COUNT(*) AS NumCarros, SUM(preco) AS TotalPago
FROM consumidores c, negocios n
WHERE c.cpf = n.comprador
GROUP BY c.cpf, c.nome, c.sobrenome
HAVING COUNT(*) > 1;
```