

Para os exercícios abaixo, utilize a base de dados de venda fornecida pela professora (está no Moodle). Não se esqueça de postar as respostas no Moodle.

```
cliente (codigo, nome, email, telefone)
funcionario (codigo, nome, email, dtaNasc, salario)
venda (numero, data, hora, codclie#, codFun#, tipo)
      codClie REFERENCES cliente (codigo)
      codFun REFERENCES funcionario (codigo)
produto (codigo, nome, preco, qtdEstoque)
produtoVendido (numero#, codProd#, qtd, valor)
      numero REFERENCES venda (numero),
      codProd REFERENCES produto (codigo)
```

Os exercícios englobam consultas usando todos as possíveis construções vistas em aula: aninhamentos com IN, ANY ou ALL, operadores de conjunto UNION, INTERSECT ou EXCEPT, uso do EXISTS. No entanto, sempre que possível, procure usar o EXISTS para exercitar a sintaxe e o raciocínio em cima do operador.

1. Nome do produto de maior preço de custo, desde que não tenha sido vendido ainda.
 - a. Usando NOT EXISTS

```
SELECT nome
FROM produto
WHERE preco = (SELECT max(a.preco)
               FROM (SELECT p1.preco
                     FROM produto p1
                     WHERE NOT EXISTS (SELECT pv.codprod
                                       FROM produtovendido pv
                                       WHERE pv.codprod = p1.codigo) ) a )
```

- b. Usando operador de conjunto

```
SELECT nome
FROM produto
WHERE preco = (SELECT max(a.preco)
               FROM ( SELECT p1.codigo, p1.preco
                     FROM produto p1
                     EXCEPT
                     SELECT pv.codprod, p.preco
                     FROM produto p join produtovendido pv on p.codigo =pv.codprod) a )
```

2. Data e hora das vendas do tipo 'A VISTA ', e nomes dos clientes.

```
SELECT data, hora, nome  
FROM venda JOIN cliente on codclie=codigo  
WHERE tipo = 'A VISTA'
```

Não é possível executar esta consulta de forma aninhada (usando EXISTS, por exemplo) porque a cláusula SELECT mais externa projeta colunas existentes nas duas tabelas envolvidas na consulta.

3. A consulta abaixo retorna uma tupla do banco:
"Leinaura"
O que quer dizer este resultado? Explique:

```
SELECT f.nome  
FROM funcionario f  
WHERE NOT EXISTS (SELECT *  
                  FROM cliente c  
                  WHERE NOT EXISTS (SELECT *  
                                    FROM venda v  
                                    WHERE v.codclie =c.codigo AND v.codfun = f.codigo))
```

4. Retornar uma tabela com duas colunas: nomes e e-mails, de clientes e funcionários. A lista deve ter uma terceira coluna que indique se ele é 'cliente' ou 'funcionário'. Os funcionários devem ser apenas os vendedores, ou seja, funcionário que aparece na tabela venda é um vendedor.

```
SELECT nome, email, 'cliente' as categoria  
FROM cliente  
UNION  
SELECT nome, email, 'funcionario'  
FROM funcionario JOIN venda ON codigo = codfun
```

5. Data e hora das vendas, e nomes dos clientes. As vendas efetuadas para clientes sem cadastro também devem ser listadas.

```
SELECT data, hora, nome  
FROM venda LEFT JOIN cliente on codclie=codigo
```

6. Nomes dos produtos que não foram vendidos no período de: início de 2004 até final de 2007
a. Usando NOT EXISTS

```
SELECT p.nome
FROM produto p
WHERE NOT EXISTS ( SELECT *
                   FROM produtovendido pv JOIN venda v ON v.numero=pv.numero
                   WHERE v.data BETWEEN '01/01/2004' AND '12/31/2007'
                   AND p.codigo = pv.codprod )
```

- b. Usando operador de conjunto

```
SELECT p.nome
FROM produto p JOIN (SELECT codigo
                    FROM produto
                    EXCEPT
                    SELECT codprod
                    FROM produtovendido pv JOIN venda v ON v.numero=pv.numero
                    WHERE v.data BETWEEN '01/01/2004' AND '12/31/2007') nv
ON nv.codigo = p.codigo
```

7. Nomes dos produtos e o preço médio de venda dele no período de: início de 2004 até final de 2009.

```
SELECT p.nome, avg(pv.valor)
FROM produto p JOIN produtoVendido pv ON p.codigo = pv.codprod
                JOIN venda v ON v.numero =pv.numero
WHERE v.data BETWEEN '01/01/2004' AND '12/31/2009'
GROUP BY p.nome
```

8. Nomes dos produtos cujo preço de venda seja inferior ao seu preço de custo.

Somente Gravata nunca foi vendida por um valor menor do que seu preço de custo (a única que não deve aparecer no resultado)

```
SELECT nome
FROM produto
WHERE preco > ANY (SELECT valor
                   FROM produtovendido
                   WHERE codprod=codigo)
```

9. Retornar o nome do funcionário que também já foi cliente. Neste caso, uma mesma pessoa é identificada pelo nome + e-mail, ou seja, cliente e funcionário que têm o mesmo nome e o mesmo e-mail são consideradas a mesma pessoa.

a. Usando EXISTS

```
SELECT f.nome  
FROM funcionario f  
WHERE EXISTS (SELECT *  
              FROM cliente c  
              WHERE c.nome = f.nome AND f.email = c.email)
```

b. Usando JOIN

```
SELECT f.nome  
FROM funcionario f JOIN cliente c ON c.nome = f.nome AND f.email = c.email
```

10. Data de venda, nomes dos produtos e total vendido no período de: início de 2003 até final de 2004, desde que este total seja superior a 100. Ordene por data de venda.

```
SELECT v.data, p.nome, count(*)  
FROM venda v JOIN produtoVendido pv on v.numero = pv.numero JOIN produto p ON p.codigo =  
pv.codprod  
WHERE v.data BETWEEN '01/01/2003' AND '12/31/2004'  
GROUP BY v.data, p.nome HAVING count(*) > 100  
ORDER BY v.data
```

Ps.: como eu não especifiquei se é total em unidade (quantidade total) ou total em valores (valor total) , aceitaria como correto qualquer um dos dois.

11. Data da venda, nome do funcionário que efetuou a venda e total vendido para o cliente 'Monira Rosa'. Ordene por data de venda.

```
SELECT v.data, f.nome, count(*)  
FROM venda v JOIN funcionario f ON f.codigo = v.codfun JOIN cliente c ON c.codigo = v.codclie  
WHERE c.nome = 'Monira Rosa'  
GROUP BY v.data, f.nome  
ORDER BY v.data
```

Ps.: como eu não especifiquei se é total em unidade (quantidade total) ou total em valores (valor total) , aceitaria como correto qualquer um dos dois.



12. Selecionar o nome do funcionário cujo salário é maior do que o funcionário mais velho da empresa.

```
SELECT nome
FROM funcionario
WHERE salario > ( SELECT salario
                  FROM funcionario
                  WHERE dtaNasc = (SELECT min(dtaNasc)
                                   FROM funcionario))
```