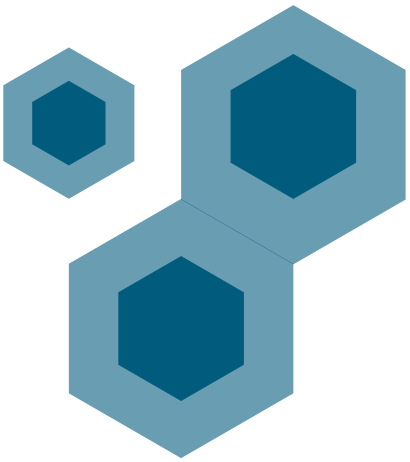


SQL 2016 - Criando Sistemas de Banco de Dados (online)

Gabarito

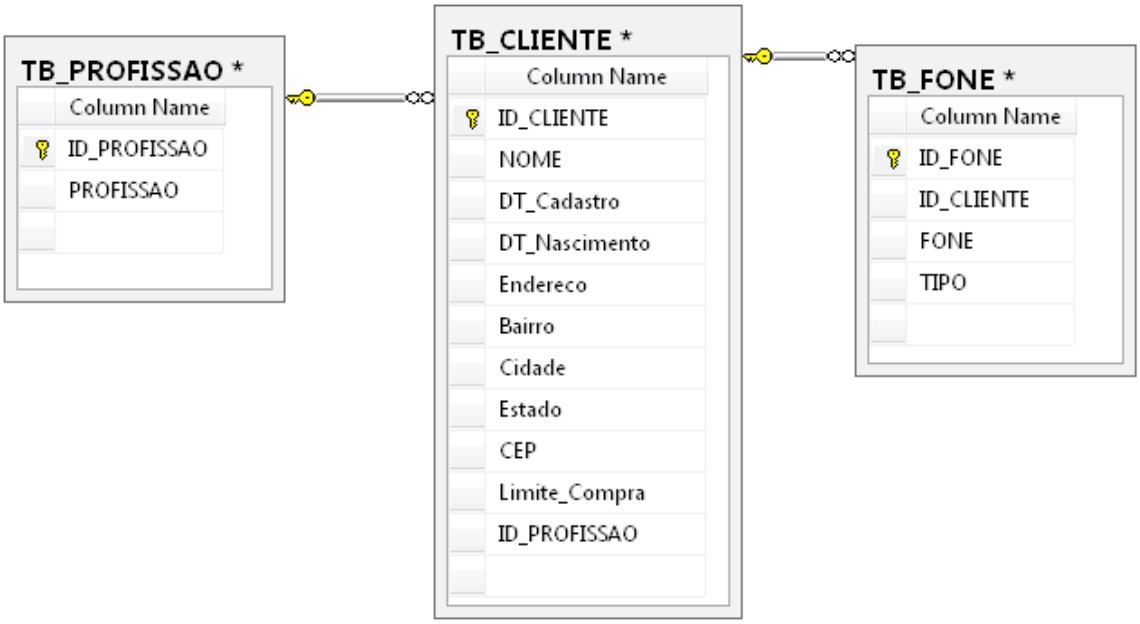
Editora IMPACTA

1827_Gabarito01
Abril/ 2017

Mãos à obra!

Laboratório 1 das Aulas 1 a 5

Passo 2 – Desenho do modelo lógico das tabelas do cadastro de clientes:



Passo 3 – Descrição do dicionário de dados do modelo lógico:

Tabela TB_PROFISSAO

Sequência	Campo	Descrição	Tipo de Dados	NOT NULL	Identity	Bytes	PK	FK	Regras	Default
1	ID_PROFISSAO	Chave da tabela	INT	Não	Sim	4	SIM			Autonumerável
2	PROFISSAO	Descrição da Profissão	Varchar(50)	Não		50				

Tabela TB_CLIENTE

Sequência	Campo	Descrição	Tipo de Dados	NOT NULL	Identity	Bytes	PK	FK	Regras	Default
1	ID_CLIENTE	Chave da tabela	INT	Não	Sim	4	Sim			Autonumerável
2	Nome	Nome do Cliente	Varchar(50)	Não		50		Sim		
3	DT_CADASTRO	Data de Cadastro	Datetime	Não		8				GETDATE()
4	DT_NASCIMENTO	Data de Nascimento	Datetime	Não		8				
5	ENDERECO	Endereço	Varchar(70)	Não		70				
6	BAIRRO	Bairro	Varchar(50)	Não		50				
7	CIDADE	Cidade	Varchar(30)	Não		30				
8	ESTADO	Estado	CHAR(2)	Não		2				
9	CEP	CEP	CHAR(8)	Não		8				
10	LIMITE_COMPRA	Limite de compra	Decimal(8,2)	Não		8				1000
11	ID_PROFISSAO	Campo Chave para profissão	INT	Não		4				

Tabela TB_FONE

Sequência	Campo	Descrição	Tipo de Dados	NOT NULL	Identity	Bytes	PK	FK	Regras	Default
1	ID_FONE	Chave da tabela	INT	Não	SIM	4	SIM			Autonumerável
2	ID_CLIENTE	Campo chave para Cliente	INT	Não		4		Sim		
3	FONE	Telefone	Varchar(11)	Não		11				
4	TIPO	Tipo do Telefone	TYNINT	Não		2				1 - Residencial 2 - Comercial 3 - Celular 4 - Recado

Passo 5 – Formas normais utilizadas:

- 1ª Forma normal ao criar a tabela de telefone;
- 2ª Forma normal ao utilizar uma tabela de profissões;
- 3ª Forma normal por não criar o campo Idade na tabela de clientes.

Laboratório 1 das Aulas 6 e 7

```
USE PEDIDOS_VAZIO;
--============================================================
--3 - Criar chaves estrangeiras para TB_PEDIDO
--============================================================
-- Com TB_CLIENTE
ALTER TABLE TB_PEDIDO ADD CONSTRAINT FK_TB_PEDIDO_TB_CLIENTE
FOREIGN KEY (CODCLI) REFERENCES TB_CLIENTE(CODCLI);
-- Com TB_VENDEDOR
ALTER TABLE TB_PEDIDO ADD CONSTRAINT FK_TB_PEDIDO_VENDEDORES
FOREIGN KEY (CODVEN) REFERENCES TB_VENDEDOR (CODVEN);
--============================================================
-- 4 - Criar chaves estrangeiras para TB_PRODUTO
--============================================================
-- Com TB_TIPOPRODUTO
ALTER TABLE TB_PRODUTO ADD CONSTRAINT FK_TB_PRODUTO_TIPOPRODUTO
FOREIGN KEY (COD_TIPO) REFERENCES TB_TIPOPRODUTO(COD_TIPO);

-- Com TB_UNIDADE
ALTER TABLE TB_PRODUTO ADD CONSTRAINT FK_TB_PRODUTO_TB_UNIDADE
FOREIGN KEY (COD_UNIDADE) REFERENCES TB_UNIDADE(COD_UNIDADE);

--============================================================
-- 5 -Criar chaves estrangeiras para TB_ITENSPEDIDO
--============================================================
-- Com TB_PEDIDO
ALTER TABLE TB_ITENSPEDIDO ADD CONSTRAINT FK_TB_ITENSPEDIDO_PEDIDOS
FOREIGN KEY (NUM_PEDIDO) REFERENCES TB_PEDIDO(NUM_PEDIDO);
-- Com TB_PRODUTO
ALTER TABLE TB_ITENSPEDIDO ADD CONSTRAINT FK_TB_ITENSPEDIDO_PRODUTOS
FOREIGN KEY (ID_PRODUTO) REFERENCES TB_PRODUTO(ID_PRODUTO);
-- Com TB_COR
ALTER TABLE TB_ITENSPEDIDO ADD CONSTRAINT FK_TB_ITENSPEDIDO_TB_COR
FOREIGN KEY (CODCOR) REFERENCES TB_COR(CODCOR);

--============================================================
-- 6 - Criar chave única para TB_UNIDADE (campo UNIDADE)
--============================================================
ALTER TABLE TB_UNIDADE ADD CONSTRAINT UNQ_TB_UNIDADE_UNIDADE
UNIQUE ( UNIDADE );
--============================================================
-- 7 - Criar chave única para TB_TIPOPRODUTO (campo TIPO)
--============================================================
ALTER TABLE TB_TIPOPRODUTO ADD CONSTRAINT UNQ_TB_TIPOPRODUTO_TIPO
UNIQUE ( TIPO );
--============================================================
-- 8 - Criar check constraints para TB_PRODUTO
--============================================================
-- Preço de venda (PRECO_VENDA) tem que ser maior que preço de
-- custo (PRECO_CUSTO)
ALTER TABLE TB_PRODUTO
ADD CONSTRAINT CHK_TB_PRODUTO_VENDA_MAIOR_QUE_CUSTO
CHECK( PRECO_VENDA > PRECO_CUSTO );
-- Preço mínimo precisa ser maior que zero
ALTER TABLE TB_PRODUTO
ADD CONSTRAINT CHK_TB_PRODUTO_PR_CUSTO_POSITIVO
CHECK( PRECO_CUSTO > 0 );
-- O campo QTD_REAL não pode ser menor que zero
ALTER TABLE TB_PRODUTO
ADD CONSTRAINT CHK_TB_PRODUTO_REAL_MAIOR_QUE_ZERO
CHECK( QTD_REAL > = 0 );

--============================================================
-- 9 - Criar check constraints para TB_ITENSPEDIDO
--============================================================
-- QUANTIDADE tem que ser >= 1
ALTER TABLE TB_ITENSPEDIDO ADD
```

```
CONSTRAINT CHK_TB_ITENSPEDIDO_QTD_MAIOR_QUE_ZERO
CHECK( QUANTIDADE >= 1 );

-- PR_UNITARIO tem que ser maior que zero
ALTER TABLE TB_ITENSPEDIDO ADD
CONSTRAINT CHK_TB_ITENSPEDIDO_PR_UNIT_MAIOR_QUE_ZERO
CHECK( PR_UNITARIO > 0 );

-- DESCONTO não pode ser maior que 10 e nem menor que zero
ALTER TABLE TB_ITENSPEDIDO ADD
CONSTRAINT CHK_TB_ITENSPEDIDO_DESCONTO_ENTRE_0_10
CHECK (DESCONTO BETWEEN 0 AND 10);

--=========================================================
-- 10 - Criar valores default para TB_PRODUTO
--=========================================================
-- PRECO_VENDA e PRECO_CUSTO valor ZERO
ALTER TABLE TB_PRODUTO ADD CONSTRAINT DEF_TB_PRODUTO_PR_VENDA
DEFAULT( 0 ) FOR PRECO_VENDA;

ALTER TABLE TB_PRODUTO
ADD CONSTRAINT DEF_TB_PRODUTO_PR_CUSTO
DEFAULT( 0 ) FOR PRECO_CUSTO;
-- QTD_REAL, QTD_ESTIMADA e QTD_MINIMA igual a zero
-- 1 comando ALTER TABLE para cada campo
-- COD_TIPO e COD_UNIDADE igual a zero
ALTER TABLE TB_PRODUTO ADD
   CONSTRAINT DEF_TB_PRODUTO_QTD_REAL
   DEFAULT(0) FOR QTD_REAL,
   CONSTRAINT DEF_TB_PRODUTO_QTD_ESTIMADA
   DEFAULT(0) FOR QTD_ESTIMADA,
   CONSTRAINT DEF_TB_PRODUTO_QTD_MINIMA
   DEFAULT(0) FOR QTD_MINIMA,
   CONSTRAINT DEF_TB_PRODUTO_COD_TIPO
   DEFAULT(0) FOR COD_TIPO,
   CONSTRAINT DEF_TB_PRODUTO_COD_UNIDADE
   DEFAULT(0) FOR COD_UNIDADE;
```

Laboratório 1 das Aulas 10 a 13

```
-- 1.  Criar um banco de dados chamado PEDIDOS_VENDA:

CREATE DATABASE PEDIDOS_VENDA;

-- Colocar em uso o banco de dados PEDIDOS_VENDA:

USE PEDIDOS_VENDA;

-- 2.  Criar tabela chamada TB_PRODUTO com os campos:
/*
       Código do produto           inteiro, autonumeração e chave primária
       Nome do produto                    alfanumérico
       Código da unid. de medida   inteiro
       Código da categoria         inteiro
       Quantidade em estoque              numérico
       Quantidade mínima           numérico
       Preço de custo                     numérico
       Preço de venda                     numérico
       Características técnicas    texto longo
       Fotografia                  binário longo
*/

CREATE TABLE TB_PRODUTO
(      COD_PRODUTO          INT    IDENTITY      PRIMARY KEY,
       DESCRICAO            VARCHAR(50),
       COD_UNIDADE          INT,
       COD_CATEGORIA INT,
       QTD_ESTOQUE          NUMERIC(12,2),
       QTD_MINIMA           NUMERIC(12,2),
       PRECO_CUSTO          NUMERIC(12,2),
       PRECO_VENDA          NUMERIC(12,2),
       CARACTERISTICAS      VARCHAR(max),
       FOTO                 VARBINARY(max)       );

-- 3.  Criar a tabela TB_UNIDADE para armazenar as unidades de medida:
/*
       Código da unidade           inteiro, autonumeração e chave primária
       Nome da unidade                    alfanumérico
*/

CREATE TABLE TB_UNIDADE
(      COD_UNIDADE          INT    IDENTITY      PRIMARY KEY,
       UNIDADE                     VARCHAR(30) );
```

```
-- 4. Inserir na tabela as seguintes unidades:
/*
PEÇAS, METROS, QUILOGRAMAS, DÚZIAS, PACOTE, CAIXA
*/

INSERT INTO TB_UNIDADE (UNIDADE)
VALUES ('PEÇAS'), ('METROS'), ('QUILOGRAMAS'), ('DÚZIAS'), ('PACOTE'), ('CAIXA');


-- 5.  Criar a tabela TB_CATEGORIA para armazenar as categorias de produto:
/*
      Código da categoria        inteiro, autonumeração e chave primária
      Nome da categoria          alfanumérico
*/

CREATE TABLE TB_CATEGORIA
(     COD_CATEGORIA INT    IDENTITY     PRIMARY KEY,
      CATEGORIA            VARCHAR(30) );

-- 6. Inserir na tabela as seguintes categorias:
/*
MOUSE, PEN-DRIVE, MONITOR DE VIDEO, TECLADO, CPU, CABO DE REDE
*/

INSERT INTO TB_CATEGORIA
VALUES ('MOUSE'), ('PEN-DRIVE'), ('MONITOR DE VIDEO'), ('TECLADO'), ('CPU'), ('CABO
DE REDE');

-- 7.Inserir os produtos a seguir utilizando a cláusula OUTPUT:

INSERT INTO TB_PRODUTO
(DESCRICAO    ,COD_UNIDADE,COD_CATEGORIA,QTD_ESTOQUE,QTD_MINIMA,PRECO_CUSTO,PRECO_
VENDA )
OUTPUT INSERTED.*
VALUES
('Caneta Azul'       ,      1 ,    1 ,    150    ,40    , 0.50 , 0.75),
('Caneta Verde'      ,      1 ,    1 ,    50     ,40    , 0.50 , 0.75),
('Caneta Vermelha'   ,      1 ,    1 ,    80     ,35    , 0.50 , 0.75),
('Lápis'             ,      1 ,    1 ,    400    ,80    , 0.50 , 0.80),
('Régua'             ,      1 ,    1 ,    40     ,10    , 1.00 , 1.50)
```

Laboratório 2 das Aulas 10 a 13

```
-- 1. Colocar o banco de dados PEDIDOS em uso
USE PEDIDOS;
-- 2. Aumentar o preço de custo de todos os produtos do tipo 2 em 15%
--    Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
-- Abrir transação
BEGIN TRAN
-- Executar o UPDATE
UPDATE TB_PRODUTO SET PRECO_CUSTO = PRECO_CUSTO * 1.15
OUTPUT INSERTED.COD_PRODUTO, INSERTED.DESCRICAO, INSERTED.COD_TIPO,
       DELETED.PRECO_CUSTO AS PRECO_ANTIGO, INSERTED.PRECO_CUSTO AS PRECO_NOVO,
          INSERTED.PRECO_CUSTO / DELETED.PRECO_CUSTO AS FATOR
WHERE COD_TIPO = 2;
-- Verificar se deu certo analisando o resultado do comando

-- Obs.: Como o campo PRECO_CUSTO tem 4 casas depois da vírgula,
--       a coluna calculada FATOR não será exatamente 1.15, mas próximo disso
-- Ok. Alteração correta
COMMIT

-- 3. Fazer com que os preços de venda dos produtos do
-- tipo 2 fiquem 30% acima do preço de custo
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_PRODUTO SET PRECO_VENDA = PRECO_CUSTO * 1.3
OUTPUT INSERTED.COD_PRODUTO, INSERTED.DESCRICAO, INSERTED.COD_TIPO,
       INSERTED.PRECO_VENDA / INSERTED.PRECO_CUSTO AS FATOR
WHERE COD_TIPO = 2;

COMMIT
-- 4. Alterar o campo IPI de todos os produtos com
--    COD_TIPO = 3 para 5%
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_PRODUTO SET IPI = 5
OUTPUT INSERTED.COD_PRODUTO, inserted.COD_TIPO,
       DELETED.IPI AS IPI_ANTIGO, INSERTED.IPI AS IPI_NOVO
WHERE COD_TIPO = 3;

COMMIT
-- 5. Reduzir em 10% (multiplicar QTD_MINIMA por 0.9) o campo
-- QTD_MINIMA de todos os produtos
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_PRODUTO SET QTD_MINIMA *= 0.9
OUTPUT INSERTED.COD_PRODUTO, DELETED.QTD_MINIMA AS QTD_ANTIGA,
       INSERTED.QTD_MINIMA AS QTD_NOVA;

COMMIT
```

```
-- 6. Alterar os seguintes campos do cliente de código 11
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
/*
       ENDEREÇO: AV. CELSO GARCIA, 1234
       BAIRRO:   TATUAPE
       CIDADE:   SAO PAULO
       ESTADO:   SP
       CEP   :   03407080
*/
-- Resposta:
BEGIN TRAN

UPDATE TB_CLIENTE
SET    ENDERECO = 'AV. CELSO GARCIA, 1234',
       BAIRRO =   'TATUAPE',
       CIDADE =   'SAO PAULO',
       ESTADO =   'SP' ,
       CEP    =   '03407080'
OUTPUT INSERTED.*
WHERE CODCLI = 11;

COMMIT
-- 7. Copiar ENDERECO, BAIRRO, CIDADE, ESTADO e CEP do
-- cliente código 13 para os campos
--  END_COB, BAI_COB, CID_COB, EST_COB e CEP_COB (do mesmo cliente)
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_CLIENTE
SET    END_COB = ENDERECO,
       BAI_COB = BAIRRO  ,
       CID_COB = CIDADE  ,
       EST_COB = ESTADO  ,
       CEP_COB = CEP
OUTPUT INSERTED.*
WHERE CODCLI = 13;

COMMIT
-- 8. Alterar o campo ICMS para 12 da tabela TB_CLIENTE para os clientes dos
--    estados RJ, RO, AC, RR, MG, PR, SC, RS, MS, MT
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_CLIENTE SET ICMS = 12
OUTPUT INSERTED.*
WHERE ESTADO IN ('RJ', 'RO', 'AC', 'RR', 'MG', 'PR', 'SC', 'RS', 'MS', 'MT');

COMMIT
-- 9. Alterar o campos ICMS para 18 de todos os clientes de SP
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_CLIENTE SET ICMS = 18
OUTPUT INSERTED.*
WHERE ESTADO = 'SP';

COMMIT
```

```
-- 10. Alterar o campo ICMS para 7 da tabela TB_CLIENTE para clientes que
--    NÃO SEJAM dos estados RJ, RO, AC, RR, MG, PR, SC, RS, MS, MT, SP
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_CLIENTE SET ICMS = 7
OUTPUT INSERTED.*
WHERE ESTADO NOT IN ('RJ', 'RO', 'AC', 'RR', 'MG', 'PR', 'SC',
                     'RS', 'MS', 'MT', 'SP');

SELECT * FROM TB_CLIENTE
WHERE ESTADO NOT IN ('RJ', 'RO', 'AC', 'RR', 'MG', 'PR', 'SC',
                     'RS', 'MS', 'MT', 'SP');

COMMIT
-- 11. Alterar para 7 o campo DESCONTO da tabela TB_ITENSPEDIDO,
-- mas somente dos itens do produto com ID_PRODUTO = 8 com
-- data de entrega em janeiro de 2014 e com
-- QUANTIDADE acima de 1000.
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_ITENSPEDIDO SET DESCONTO = 7
OUTPUT INSERTED.*
WHERE ID_PRODUTO = 8 AND
      DATA_ENTREGA BETWEEN '2014.1.1' AND '2014.1.31' AND
      QUANTIDADE > 1000;

COMMIT
-- 12. Zerar o campo DESCONTO de todos os itens de pedido
-- com quantidade abaixo de 1000, com data de entrega
-- posterior a 1-Junho-2014 e que tenham desconto acima de zero.
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

UPDATE TB_ITENSPEDIDO SET DESCONTO = 0
OUTPUT INSERTED.*
WHERE DATA_ENTREGA > '2014.6.1' AND QUANTIDADE < 1000 AND
      DESCONTO > 0;

COMMIT
-- 13. Usando SELECT INTO gere uma cópia da tabela VENDEDORES
SELECT * INTO VENDEDORES_TMP FROM TB_VENDEDOR;
```

```
-- 14. Exclua de VENDEDORES_TMP os registros com CODVEN acima de 5
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

DELETE FROM VENDEDORES_TMP
OUTPUT DELETED.*
WHERE CODVEN > 5;

COMMIT
-- 15. Utilizando o comando SELECT...INTO, faça uma cópia da tabela TB_PEDIDO
--     chamada COPIA_PEDIDOS
-- Resposta:
SELECT * INTO COPIA_PEDIDOS FROM TB_PEDIDO;

-- 16. Exclua os registros da tabela COPIA_PEDIDOS que sejam  do vendedor código 2
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

DELETE FROM COPIA_PEDIDOS
OUTPUT DELETED.*
WHERE CODVEN = 2;

COMMIT
-- 17. Exclua os registros da tabela COPIA_PEDIDOS que sejam
-- do primeiro semestre de 2014
-- Utilize transação e a cláusula OUTPUT para conferir o resultado
-- Resposta:
BEGIN TRAN

DELETE FROM COPIA_PEDIDOS
OUTPUT DELETED.*
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.6.30';

COMMIT
-- 18. Exclua todos os registros restantes da tabela COPIA_PEDIDOS
-- Resposta:
DELETE FROM COPIA_PEDIDOS;

-- 19. Exclua a tabela COPIA_PEDIDOS do banco de dados
-- Resposta:
DROP TABLE COPIA_PEDIDOS;
```

Laboratório 1 das Aulas 14 a 19

```
/* 1.Usar o banco de dados PEDIDOS */
USE PEDIDOS;

-- Mostrar as tabelas do banco
EXEC SP_TABLES;

-- Mostrar a estrutura de uma tabela
EXEC SP_HELP TB_PRODUTO;

/* 2.Listar TB_PRODUTO mostrando COD_PRODUTO, DESCRICAO,
     PRECO_CUSTO, PRECO_VENDA e calculando
     o lucro unitário */
SELECT
   COD_PRODUTO, DESCRICAO, PRECO_CUSTO, PRECO_VENDA,
   PRECO_VENDA - PRECO_CUSTO AS LUCRO_UNITARIO
FROM TB_PRODUTO;

/* 3.Listar COD_PRODUTO, DESCRICAO e valor total investido
     no estoque daquele produto (QTD_REAL * PRECO_CUSTO)
     da tabela TB_PRODUTO */
SELECT
  COD_PRODUTO, DESCRICAO,
  QTD_REAL * PRECO_CUSTO AS VLR_INVESTIDO
FROM TB_PRODUTO;

/* 4.Listar TB_ITENSPEDIDO, campos NUM_PEDIDO, NUM_ITEM, COD_PRODUTO,
     PR_UNITARIO, QUANTIDADE, DESCONTO e valor
     PR_UNITARIO * QUANTIDADE * (1-DESCONTO/100) */
SELECT
   NUM_PEDIDO, NUM_ITEM, COD_PRODUTO, PR_UNITARIO,
   QUANTIDADE, DESCONTO,
   PR_UNITARIO * QUANTIDADE * (1-DESCONTO/100) AS VALOR
FROM TB_ITENSPEDIDO;

/* 5.Listar tabela TB_PRODUTO, campos COD_PRODUTO, DESCRICAO,
     PRECO_CUSTO, PRECO_VENDA e lucro estimado em reais
     QTD_REAL * (PRECO_VENDA - PRECO_CUSTO) */
SELECT
   COD_PRODUTO, DESCRICAO, PRECO_CUSTO, PRECO_VENDA,
   QTD_REAL * (PRECO_VENDA - PRECO_CUSTO) AS LUCRO_EST
FROM TB_PRODUTO;

/* 6.Listar COD_PRODUTO, DESCRICAO, PRECO_CUSTO,
     PRECO_VENDA e lucro unitário em reais
     (PRECO_VENDA – PRECO_CUSTO) e percentual de lucro
    (100 * (PRECO_VENDA - PRECO_CUSTO) / PRECO_CUSTO) */
SELECT
  COD_PRODUTO, DESCRICAO, PRECO_CUSTO, PRECO_VENDA,
  PRECO_VENDA - PRECO_CUSTO AS [Lucro R$],
  100 * (PRECO_VENDA - PRECO_CUSTO) / PRECO_CUSTO
                            AS [Lucro %]
FROM TB_PRODUTO
WHERE PRECO_CUSTO > 0;
```

Laboratório 2 das Aulas 14 a 19

```
/* 1. Usar o banco de dados PEDIDOS. */
USE PEDIDOS;

/* 2.Listar TB_PRODUTO, criando campo calculado
   (QTD_REAL - QTD_MINIMA), e filtrar os
   registros resultantes, mostrando somente aqueles que
   tiverem a quantidade real abaixo da quantidade mínima. */
SELECT
 COD_PRODUTO, DESCRICAO, QTD_REAL, QTD_MINIMA,
 QTD_REAL - QTD_MINIMA AS DIFERENCA
FROM TB_PRODUTO
WHERE QTD_REAL < QTD_MINIMA;

/* 3.Listar TB_PRODUTO com quantidade real acima
   de 5000. */
SELECT * FROM TB_PRODUTO
WHERE QTD_REAL > 5000
ORDER BY QTD_REAL;

/* 4. Listar produtos com preço de venda abaixo de R$0,50.*/
SELECT * FROM TB_PRODUTO
WHERE PRECO_VENDA < 0.5
ORDER BY PRECO_VENDA;

/* 5.Listar TB_PEDIDO com valor total (VLR_TOTAL)
   acima de R$15.000,00. */
SELECT * FROM TB_PEDIDO
WHERE VLR_TOTAL > 15000
ORDER BY VLR_TOTAL;

/* 6.Listar produtos com QTD_REAL entre 500 e 1000 unidades.*/
SELECT * FROM TB_PRODUTO
WHERE QTD_REAL >= 500 AND QTD_REAL <= 1000
ORDER BY QTD_REAL;
-- OU
SELECT * FROM TB_PRODUTO
WHERE QTD_REAL BETWEEN 500 AND 1000
ORDER BY QTD_REAL;

/* 7.Listar pedidos com valor total entre
   R$15.000,00 e R$25.000,00. */
SELECT * FROM TB_PEDIDO
WHERE VLR_TOTAL BETWEEN 15000 AND 25000
ORDER BY VLR_TOTAL;

/* 8.Listar produtos com QTD_REAL > 5000 e COD_TIPO = 6. */
SELECT * FROM TB_PRODUTO
WHERE QTD_REAL > 5000 AND COD_TIPO = 6;

/* 9.Listar produtos com QTD_REAL > 5000 ou COD_TIPO = 6. */
SELECT * FROM TB_PRODUTO
WHERE QTD_REAL > 5000 OR COD_TIPO = 6;

/* 10.Listar pedidos com valor total abaixo de
   R$100,00 ou acima de R$100.000,00. */
SELECT * FROM TB_PEDIDO
WHERE VLR_TOTAL < 100 OR VLR_TOTAL > 100000
ORDER BY VLR_TOTAL;

/* 11.Listar produtos com QTD_REAL menor que 500 ou
   maior que 1000. */
SELECT * FROM TB_PRODUTO
WHERE QTD_REAL < 500 OR QTD_REAL > 1000
ORDER BY QTD_REAL;
```

Laboratório 3 das Aulas 14 a 19

```
/* 1. Usar o banco de dados PEDIDOS. */
USE PEDIDOS;

/* 2. Listar clientes do estado de São Paulo (SP). */
-- Campo ESTADO da tabela CLIENTES é CHAR(2) e contém a sigla do estado
SELECT * FROM TB_CLIENTE WHERE ESTADO = 'SP';

/* 3. Listar clientes do estado de  Minas Gerais e Rio de Janeiro
      (MG, RJ). */
SELECT * FROM TB_CLIENTE WHERE ESTADO IN ('MG','RJ');

/* 4. Listar clientes do estado de  São Paulo, Minas Gerais e
      Rio de Janeiro (SP,MG,RJ). */
SELECT * FROM TB_CLIENTE WHERE ESTADO IN ('SP','MG','RJ');

/* 5. Listar os vendedores com o nome LEIA. */
SELECT * FROM TB_VENDEDOR WHERE NOME LIKE 'LEIA%';

/* 6. Listar todos os clientes que tenham NOME
      começando com BRINDES. */
SELECT * FROM TB_CLIENTE WHERE NOME LIKE 'BRINDES%';

/* 7. Listar todos os clientes que tenham NOME
      terminando com BRINDES. */
SELECT * FROM TB_CLIENTE WHERE NOME LIKE '%BRINDES';

/* 8. Listar todos os clientes que tenham NOME
      contendo BRINDES. */
SELECT * FROM TB_CLIENTE WHERE NOME LIKE '%BRINDES%';

/* 9. Listar todos os produtos com DESCRICAO
      começando por "CANETA".*/
SELECT * FROM TB_PRODUTO WHERE DESCRICAO LIKE 'CANETA%';

/* 10. Listar todos os produtos com
       DESCRICAO contendo "SPECIAL". */
SELECT * FROM TB_PRODUTO WHERE DESCRICAO LIKE '%SPECIAL%';

/* 11. Listar todos os produtos com
       DESCRICAO terminando por "GOLD".*/
SELECT * FROM TB_PRODUTO WHERE DESCRICAO LIKE '%GOLD';

/* 12. Listar todos os clientes que tenham a
       letra A como segundo caractere do nome. */
SELECT * FROM TB_CLIENTE WHERE NOME LIKE '_A%';

/* 13. Listar todos os produtos que tenham
       "0" (ZERO) como segundo caractere do
       campo COD_PRODUTO. */
SELECT * FROM TB_PRODUTO WHERE COD_PRODUTO LIKE '_0%';

/* 14. Listar todos os produtos que tenham
       "A" como terceiro caractere do
       campo COD_PRODUTO.*/
SELECT * FROM TB_PRODUTO WHERE COD_PRODUTO LIKE '__A%';
```

Laboratório 1 das Aulas 20 a 22

```
-- 01
USE PEDIDOS;
/* 02- Listar os campos NUM_PEDIDO, DATA_EMISSAO,
   VLR_TOTAL de TB_PEDIDO seguido do NOME do vendedor. */
SELECT P.NUM_PEDIDO, P.DATA_EMISSAO, P.VLR_TOTAL,
       V.NOME AS VENDEDOR
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN;

/* 03- Listar os campos NUM_PEDIDO, DATA_EMISSAO,
   VLR_TOTAL de TB_PEDIDO seguido do NOME do cliente. */
SELECT P.NUM_PEDIDO, P.DATA_EMISSAO, P.VLR_TOTAL,
       C.NOME AS CLIENTE
FROM TB_PEDIDO P
     JOIN TB_CLIENTE C ON P.CODCLI = C.CODCLI;

/* 04- Listar os pedidos com o nome do vendedor e o nome do cliente */
SELECT P.NUM_PEDIDO, P.DATA_EMISSAO, P.VLR_TOTAL,
       V.NOME AS VENDEDOR, C.NOME AS CLIENTE
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
     JOIN TB_CLIENTE C   ON P.CODCLI = C.CODCLI;

/* 05- Listar os itens de pedido (TB_ITENSPEDIDO) com o
   nome do produto (TB_PRODUTO.DESCRICAO)*/
SELECT I.*, PR.DESCRICAO
FROM TB_ITENSPEDIDO I
     JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO;

/* 06- Listar os produtos (tabela TB_PRODUTO, campos
   COD_PRODUTO, DESCRICAO) com a
   descrição do tipo de produto (TB_TIPOPRODUTO.TIPO)*/
SELECT
   PR.COD_PRODUTO, PR.DESCRICAO, T.TIPO
FROM TB_PRODUTO PR
   JOIN TB_TIPOPRODUTO T ON PR.COD_TIPO = T.COD_TIPO;

/* 07- Listar os produtos com a descrição do
    tipo (TB_TIPOPRODUTO.TIPO) e da unidade (TB_UNIDADE.UNIDADE). */
SELECT
   PR.COD_PRODUTO, PR.DESCRICAO, T.TIPO, U.UNIDADE
FROM TB_PRODUTO PR
   JOIN TB_TIPOPRODUTO T ON PR.COD_TIPO = T.COD_TIPO
   JOIN TB_UNIDADE U ON PR.COD_UNIDADE = U.COD_UNIDADE;

/* 08- Listar os itens de pedido (tabela TB_ITENSPEDIDO, campos NUM_PEDIDO, NUM_
ITEM,
   COD_PRODUTO, QUANTIDADE, PR_UNITARIO) com o nome do
   produto (TB_PRODUTO.DESCRICAO), descrição do
    tipo (TB_TIPOPRODUTO.TIPO) e da unidade (TB_UNIDADE.UNIDADE). */
SELECT
   I.*, PR.COD_PRODUTO, PR.DESCRICAO, T.TIPO, U.UNIDADE
FROM TB_ITENSPEDIDO I
   JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO
   JOIN TB_TIPOPRODUTO T ON PR.COD_TIPO = T.COD_TIPO
   JOIN TB_UNIDADE U ON PR.COD_UNIDADE = U.COD_UNIDADE;
```

```
/* 09- Listar os itens de pedido (tabela TB_ITENSPEDIDO, campos NUM_PEDIDO, NUM_
ITEM,
   COD_PRODUTO, QUANTIDADE, PR_UNITARIO) com o nome do
   produto (TB_PRODUTO.DESCRICAO), descrição do
   tipo (TB_TIPOPRODUTO.TIPO), nome da unidade (TB_UNIDADE.UNIDADE) e
   nome da cor (TB_COR.COR). */
SELECT
   I.*, PR.COD_PRODUTO, PR.DESCRICAO, T.TIPO, U.UNIDADE,
   CR.COR
FROM TB_ITENSPEDIDO I
   JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO
   JOIN TB_TIPOPRODUTO T ON PR.COD_TIPO = T.COD_TIPO
   JOIN TB_UNIDADE U ON PR.COD_UNIDADE = U.COD_UNIDADE
   JOIN TB_COR CR ON I.CODCOR = CR.CODCOR;

/* 10- Listar todos os pedidos (TB_PEDIDO) do vendedor 'MARCELO'
em Jan/2014.*/
SELECT P.*, V.NOME AS VENDEDOR
FROM TB_PEDIDO P JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
      V.NOME = 'MARCELO';

/* 11- Listar os nomes dos clientes (TB_CLIENTE.NOME) que compraram
   em janeiro de 2014. */
--- Não é a melhor solução
SELECT C.NOME AS CLIENTE
FROM TB_CLIENTE C JOIN TB_PEDIDO P ON C.CODCLI = P.CODCLI
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
ORDER BY C.NOME;
-- solução 1
SELECT C.NOME AS CLIENTE, P.*
FROM TB_CLIENTE C JOIN TB_PEDIDO P ON C.CODCLI = P.CODCLI
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
ORDER BY C.NOME;
-- solução 2
SELECT DISTINCT C.NOME AS CLIENTE
FROM TB_CLIENTE C JOIN TB_PEDIDO P ON C.CODCLI = P.CODCLI
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
ORDER BY C.NOME;

/* 12- Listar os nomes de produto (TB_PRODUTO.DESCRICAO) que foram
   vendidos em Janeiro de 2014. */
SELECT DISTINCT PR.DESCRICAO
FROM TB_PRODUTO PR
     JOIN TB_ITENSPEDIDO I ON PR.ID_PRODUTO = I.ID_PRODUTO
     JOIN TB_PEDIDO PE ON I.NUM_PEDIDO = PE.NUM_PEDIDO
WHERE PE.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31';

/* 13- Listar NUM_PEDIDO, VLR_TOTAL (TB_PEDIDO) e
   NOME do CLIENTE. Deve mostrar apenas TB_PEDIDO
   de janeiro de 2014 e de clientes que tenham
   NOME começando por 'MARCIO' */
SELECT P.NUM_PEDIDO, P.VLR_TOTAL, C.NOME AS CLIENTE
FROM TB_PEDIDO P JOIN TB_CLIENTE C ON P.CODCLI = C.CODCLI
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
      C.NOME LIKE 'MARCIO%';

 /* 14- Listar NUM_PEDIDO, QUANTIDADE vendida,
   o PR_UNITARIO (de TB_ITENSPEDIDO), DESCRICAO (TB_PRODUTO),
   e o NOME do vendedor (TB_VENDEDOR) que vendeu cada
   item de pedido */
SELECT
   I.NUM_PEDIDO, I.QUANTIDADE, I.PR_UNITARIO,
   PR.DESCRICAO, V.NOME AS VENDEDOR
FROM TB_PEDIDO PE
   JOIN TB_VENDEDOR V ON PE.CODVEN = V.CODVEN
   JOIN TB_ITENSPEDIDO I ON PE.NUM_PEDIDO = I.NUM_PEDIDO
   JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO;
```

```sql
/* 15- Listar todos os itens de Pedido com
   Desconto superior a 7%. Deve mostrar
   NUM_PEDIDO, DESCRICAO do produto,
   NOME do cliente, nome do VENDEDOR e
   QUANTIDADE vendida */
SELECT
  I.NUM_PEDIDO, I.QUANTIDADE, I.DESCONTO,
  PR.DESCRICAO, V.NOME AS VENDEDOR, C.NOME AS CLIENTE
FROM
  TB_ITENSPEDIDO I
  JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO
  JOIN TB_PEDIDO PE ON I.NUM_PEDIDO = PE.NUM_PEDIDO
  JOIN TB_CLIENTE C ON PE.CODCLI = C.CODCLI
  JOIN TB_VENDEDOR V ON PE.CODVEN = V.CODVEN
WHERE I.DESCONTO > 7;

 /* 16- Listar os itens de pedido com o nome do
    produto, descrição do tipo e descrição da
    unidade e o nome da cor,
    mas apenas os itens vendidos em
    janeiro de 2014 na cor 'LARANJA' */
-- CÓPIA DO 8
SELECT
   I.*, PR.COD_PRODUTO, PR.DESCRICAO, T.TIPO, U.UNIDADE,
   CR.COR
FROM TB_ITENSPEDIDO I
   JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO
   JOIN TB_TIPOPRODUTO T ON PR.COD_TIPO = T.COD_TIPO
   JOIN TB_UNIDADE U ON PR.COD_UNIDADE = U.COD_UNIDADE
   JOIN TB_COR CR ON I.CODCOR = CR.CODCOR
   JOIN TB_PEDIDO PE ON I.NUM_PEDIDO = PE.NUM_PEDIDO
WHERE PE.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
   CR.COR = 'LARANJA';

/* 17- Listar NOME e FONE1 dos fornecedores
   de quem compramos o produto 'CANETA STAR I' */
SELECT F.NOME, F.FONE1
FROM TB_FORNECEDOR F
     JOIN TB_PROD_FORN PF ON F.COD_FORNECEDOR = PF.COD_FORNECEDOR
     JOIN TB_PRODUTO P ON PF.ID_PRODUTO = P.ID_PRODUTO
WHERE P.DESCRICAO = 'CANETA STAR I';

/* 18- Listar a DESCRICAO dos produtos que compramos do fornecedor cujo NOME
     começa com 'LINCE'     */
SELECT P.DESCRICAO
FROM TB_FORNECEDOR F
     JOIN TB_PROD_FORN PF ON F.COD_FORNECEDOR = PF.COD_FORNECEDOR
     JOIN TB_PRODUTO P ON PF.ID_PRODUTO = P.ID_PRODUTO
WHERE F.NOME LIKE 'LINCE%';

/* 19- Listar os NOME e FONE1 dos fornecedores,
   bem como DESCRICAO dos produtos
    com QTD_REAL abaixo de QTD_MINIMA */
SELECT F.NOME, F.FONE1, P.DESCRICAO
FROM TB_FORNECEDOR F
     JOIN TB_PROD_FORN PF ON F.COD_FORNECEDOR = PF.COD_FORNECEDOR
     JOIN TB_PRODUTO P ON PF.ID_PRODUTO = P.ID_PRODUTO
WHERE P.QTD_REAL < P.QTD_MINIMA;

/* 20- Listar todos os produtos que compramos do fornecedor
   cujo nome começa com 'FESTO' */
SELECT P.DESCRICAO
FROM TB_FORNECEDOR F
     JOIN TB_PROD_FORN PF ON F.COD_FORNECEDOR = PF.COD_FORNECEDOR
     JOIN TB_PRODUTO P ON PF.ID_PRODUTO = P.ID_PRODUTO
WHERE F.NOME LIKE 'FESTO%';
```

Laboratório 1 da Aula 23

```
-- 1. Colocar em uso o banco de dados PEDIDOS
-- Resp.:
USE PEDIDOS;

/* 2. Alterar a tabela TB_CARGO, mudando o salário inicial do cargo
      OFFICE BOY para 600,00 */
-- Resp.:
BEGIN TRAN
--
UPDATE TB_CARGO SET SALARIO_INIC = 600
WHERE CARGO = 'OFFICE BOY';
--
COMMIT
/* 3. Alterar a tabela de cargos, estipulando 10% de aumento para o campo SALARIO_
INIC
      de todos os cargos */
-- Resp.:
UPDATE TB_CARGO SET SALARIO_INIC *= 1.1;

/* 4. Transferir para o campo SALARIO da tabela TB_EMPREGADO o salário inicial
      cadastrado no cargo correspondente da TB_CARGO */
-- Resp.:
-- Solução 1
UPDATE TB_EMPREGADO
SET SALARIO = (SELECT SALARIO_INIC FROM TB_CARGO
               WHERE COD_CARGO = TB_EMPREGADO.COD_CARGO);
-- Solução 2
UPDATE TB_EMPREGADO
SET SALARIO = C.SALARIO_INIC
FROM TB_EMPREGADO E
     JOIN TB_CARGO C ON E.COD_CARGO = C.COD_CARGO;

/* 5. Reajustar os preços de venda de todos os produtos de modo que
      fiquem 30% acima do preço de custo  (PRECO_VENDA = PRECO_CUSTO * 1.3) */
-- Resp.:
BEGIN TRAN

UPDATE TB_PRODUTO SET PRECO_VENDA = PRECO_CUSTO * 1.3;
-- Testando
SELECT
  ID_PRODUTO, PRECO_CUSTO, PRECO_VENDA,
  100 * (PRECO_VENDA - PRECO_CUSTO) / PRECO_CUSTO
FROM TB_PRODUTO
WHERE PRECO_CUSTO > 0;

COMMIT
```

```
/* 6. Reajustar os preços de venda de todos os produtos de modo que
      fiquem 20% acima do preço de custo, mas somente dos produtos com
      COD_TIPO = 5 */
-- Resp.:
BEGIN TRAN

UPDATE TB_PRODUTO SET PRECO_VENDA = PRECO_CUSTO * 1.2
WHERE COD_TIPO = 5;

-- Testando
SELECT
  ID_PRODUTO, PRECO_CUSTO, PRECO_VENDA,
  100 * (PRECO_VENDA - PRECO_CUSTO) / PRECO_CUSTO
FROM TB_PRODUTO
WHERE PRECO_CUSTO > 0 AND COD_TIPO = 5;

COMMIT

/* 7. Reajustar os preços de venda de todos os produtos de modo que
   fiquem 40% acima do preço de custo, mas somente dos produtos com
   descrição do tipo igual a REGUA
   Tabela TB_PRODUTO -> PRECO_VENDA = PRECO_CUSTO * 1.4
   para TB_PRODUTO com TB_TIPOPRODUTO.TIPO = 'REGUA'
   É preciso fazer um JOIN de TB_PRODUTO com TB_TIPOPRODUTO */
-- Resp.:
BEGIN TRAN

UPDATE TB_PRODUTO SET PRECO_VENDA = PRECO_CUSTO * 1.4
FROM TB_PRODUTO P
     JOIN TB_TIPOPRODUTO T ON P.COD_TIPO = T.COD_TIPO
WHERE T.TIPO = 'REGUA';

COMMIT

/* 8.  Alterar a tabela TB_ITENSPEDIDO, mudando todos os itens com produto
       de cor 'VERMELHO' para cor 'LARANJA', mas somente os pedidos com data de
       entrega em Outubro de 2014 */
-- Resp.:
-- SOLUÇÃO 1
       BEGIN TRAN

    UPDATE TB_ITENSPEDIDO
    SET CODCOR = (SELECT CODCOR FROM TB_COR
                  WHERE COR = 'LARANJA')
    WHERE CODCOR = (SELECT CODCOR FROM TB_COR
                  WHERE COR = 'VERMELHO') AND
          DATA_ENTREGA BETWEEN '2014.10.1' AND '2014.10.31';

    ROLLBACK
-- SOLUÇÃO 2
       BEGIN TRAN

    UPDATE TB_ITENSPEDIDO
    SET CODCOR = L.CODCOR
    FROM TB_ITENSPEDIDO I JOIN TB_COR V ON I.CODCOR = V.CODCOR
                          JOIN TB_COR L ON L.COR = 'LARANJA'
    WHERE V.COR = 'VERMELHO' AND
          I.DATA_ENTREGA BETWEEN '2014.10.1' AND '2014.10.31';

    ROLLBACK
```

```
/* 9. Alterar o campo ICMS para 12 da tabela TB_CLIENTE para clientes dos
      estados RJ, RO, AC, RR, MG, PR, SC, RS, MS e MT */
-- Resp.:
UPDATE TB_CLIENTE SET ICMS = 12
WHERE ESTADO IN ('RJ', 'RO', 'AC', 'RR', 'MG', 'PR', 'SC',
                 'RS', 'MS', 'MT');

/* 10. Alterar o campo ICMS para 18 para clientes de SP */
-- Resp.:
UPDATE TB_CLIENTE SET ICMS = 18
WHERE ESTADO = 'SP';

/* 11. Alterar o campo ICMS para 7 da tabela TB_CLIENTE para clientes que
--     NÃO SEJAM dos estados RJ, RO, AC, RR, MG, PR, SC, RS, MS, MT e SP */
-- Resp.:
UPDATE TB_CLIENTE SET ICMS = 7
WHERE ESTADO NOT IN ('RJ', 'RO', 'AC', 'RR', 'MG', 'PR', 'SC',
                     'RS', 'MS', 'MT', 'SP');

/* 12. Criar a tabela ESTADOS com os campos
     COD_ESTADO      inteiro, autonumeração e chave primária
     SIGLA           Char(2)
     ICMS            numérico, tamanho 4 com 2 decimais
*/
-- Resp.:
CREATE TABLE ESTADOS
(      COD_ESTADO          INT          IDENTITY     PRIMARY KEY,
       SIGLA               CHAR(2) UNIQUE,
       ICMS                NUMERIC(4,2) );

/* 13. Copiar para a tabela ESTADOS o resultado do SELECT abaixo
SELECT DISTINCT ESTADO, ICMS FROM TB_CLIENTE
WHERE ESTADO IS NOT NULL
*/
-- Obs.: Este SELECT deve retornar 21 linhas e sem repetir estado
-- Se der diferente é porque seus UPDATES de ICMS estão errados
-- Resp.:
INSERT INTO ESTADOS ( SIGLA, ICMS )
SELECT DISTINCT ESTADO, ICMS FROM TB_CLIENTE
WHERE ESTADO IS NOT NULL;

-- 14. Criar o campo COD_ESTADO na tabela TB_CLIENTE
ALTER TABLE TB_CLIENTE ADD COD_ESTADO INT;

/* 15. Copiar para TB_CLIENTE.COD_ESTADO o código do estado gerado na tabela ESTADOS
*/
-- Resp.:
---- COM SUB-QUERY
BEGIN TRAN

UPDATE TB_CLIENTE
SET COD_ESTADO = (SELECT COD_ESTADO FROM ESTADOS
                  WHERE SIGLA = TB_CLIENTE.ESTADO);

ROLLBACK
---- COM JOIN
BEGIN TRAN

UPDATE TB_CLIENTE
SET COD_ESTADO = E.COD_ESTADO
FROM TB_CLIENTE C JOIN ESTADOS E ON C.ESTADO = E.SIGLA;

COMMIT
-- Testando
SELECT C.NOME, C.ESTADO AS ESTADO_CLI,
       E.SIGLA AS ESTADO_EST
FROM TB_CLIENTE C JOIN ESTADOS E
     ON C.COD_ESTADO = E.COD_ESTADO;
```

Laboratório 1 das Aulas 24 e 25

```
-- 1. Colocar em uso o banco de dados CURSOS_INFORMATICA, abrir e executar
-- os comandos de inserção dos arquivos.
USE CURSOS_INFORMATICA;

-- 2. Apresentar todas as salas de aula para as quais não há nenhum
--    curso marcado
-- Com JOIN
SELECT S.*
FROM CAD_SALAS S LEFT JOIN MOV_CURSOS C ON S.ID_SALA = C.ID_SALA
WHERE C.ID_SALA IS NULL;
-- Com Sub-Query
SELECT * FROM CAD_SALAS
WHERE ID_SALA NOT IN (SELECT ID_SALA FROM MOV_CURSOS);


-- 3. Apresentar todos os treinamentos para os quais não temos instrutor
-- Com JOIN
SELECT T.*
FROM CAD_TREINAMENTOS T
LEFT JOIN CAD_INSTRUTORES_TREINAMENTOS IT ON T.ID_TREINAMENTO = IT.ID_TREINAMENTO
WHERE IT.ID_TREINAMENTO IS NULL;
-- Com Sub-Query
SELECT * FROM CAD_TREINAMENTOS
WHERE ID_TREINAMENTO NOT IN (SELECT ID_TREINAMENTO
                             FROM CAD_INSTRUTORES_TREINAMENTOS);

-- 4. Apresentar os alunos (CAD_PESSOAS) que não tem e nem tiveram
--    nenhum curso agendado
-- Com JOIN
SELECT P.*
FROM CAD_PESSOAS P
   LEFT JOIN MOV_CURSOS_ALUNOS CA ON P.ID_PESSOA = CA.ID_PESSOA_ALUNO
WHERE CA.ID_PESSOA_ALUNO IS NULL AND P.SN_ALUNO = 'S';
-- Com Sub-Query
SELECT * FROM CAD_PESSOAS
WHERE ID_PESSOA NOT IN (SELECT ID_PESSOA_ALUNO FROM MOV_CURSOS_ALUNOS)
      AND SN_ALUNO = 'S';

-- 5. Apresentar os departamentos onde não existem funcionários cadastrados
-- Com JOIN
SELECT D.*
FROM CAD_DEPTOS D
     LEFT JOIN CAD_FUNCIONARIOS F ON D.ID_DEPTO = F.ID_DEPTO
WHERE F.ID_DEPTO IS NULL;
-- Com Sub-Query
SELECT * FROM CAD_DEPTOS
WHERE ID_DEPTO NOT IN (SELECT ID_DEPTO FROM CAD_FUNCIONARIOS);

-- 6. Apresentar os cargos para os quais não existem funcionários cadastrados
-- Com JOIN
SELECT C.*
FROM CAD_CARGOS C LEFT JOIN CAD_FUNCIONARIOS F ON C.ID_CARGO = F.ID_CARGO
WHERE F.ID_CARGO IS NULL;
-- Com Sub-Query
SELECT * FROM CAD_CARGOS
WHERE ID_CARGO NOT IN (SELECT ID_CARGO FROM CAD_FUNCIONARIOS);
```

```
-- 7. Apresentar as pessoas que sejam de estados cujo ICMS seja menor que 7
-- Com JOIN
SELECT P.*, E.ESTADO, E.ICMS
FROM CAD_PESSOAS P JOIN CAD_MUNICIPIOS M ON P.ID_MUNICIPIO = M.ID_MUNICIPIO
                   JOIN CAD_ESTADOS E ON M.ID_ESTADO = E.ID_ESTADO
WHERE E.ICMS < 7;

-- Com Sub-Query
SELECT * FROM CAD_PESSOAS
WHERE ID_MUNICIPIO IN (SELECT ID_MUNICIPIO FROM CAD_MUNICIPIOS
                       WHERE ID_ESTADO IN ( SELECT ID_ESTADO
                                            FROM CAD_ESTADOS
                                            WHERE ICMS < 7 ) );

-- 8. Apresentar os dados do instrutor que possui o maior valor hora
-- Solução 1
SELECT * FROM CAD_PESSOAS
WHERE ID_PESSOA IN
(
SELECT ID_PESSOA_INSTR FROM CAD_INSTRUTORES_TREINAMENTOS
WHERE VLR_HORA = (SELECT MAX(VLR_HORA) FROM CAD_INSTRUTORES_TREINAMENTOS)
);
-- Solução 2
SELECT TOP 1 ID_PESSOA, NOME, (SELECT TOP 1 VLR_HORA FROM CAD_INSTRUTORES_TREINAMEN-
TOS
                 WHERE ID_PESSOA_INSTR = CAD_PESSOAS.ID_PESSOA
                 ORDER BY VLR_HORA DESC) AS VLR_HORA
FROM CAD_PESSOAS
WHERE SN_INSTRUTOR = 'S'
ORDER BY 3 DESC;
-- Solução 3
SELECT TOP 1 WITH TIES P.*, I.VLR_HORA
FROM CAD_PESSOAS P JOIN CAD_INSTRUTORES_TREINAMENTOS I ON P.ID_PESSOA = I.ID_PES-
SOA_INSTR
ORDER BY I.VLR_HORA DESC;

-- 9. Apresentar os dados do instrutor que possui o MENOR valor hora
-- Solução 1
SELECT * FROM CAD_PESSOAS
WHERE ID_PESSOA IN
(
SELECT ID_PESSOA_INSTR FROM CAD_INSTRUTORES_TREINAMENTOS
WHERE VLR_HORA = (SELECT MIN(VLR_HORA) FROM CAD_INSTRUTORES_TREINAMENTOS)
);
-- Solução 2
SELECT TOP 1 ID_PESSOA, NOME, (SELECT TOP 1 VLR_HORA FROM CAD_INSTRUTORES_TREINAMEN-
TOS
                 WHERE ID_PESSOA_INSTR = CAD_PESSOAS.ID_PESSOA
                 ORDER BY VLR_HORA) AS VLR_HORA
FROM CAD_PESSOAS
WHERE SN_INSTRUTOR = 'S'
ORDER BY 3;
-- Solução 3
SELECT TOP 1 WITH TIES P.*, I.VLR_HORA
FROM CAD_PESSOAS P JOIN CAD_INSTRUTORES_TREINAMENTOS I ON P.ID_PESSOA = I.ID_PES-
SOA_INSTR
ORDER BY I.VLR_HORA;
```

Laboratório 1 das Aulas 26 e 27

```
--=====================================
/* 01- Colocar em uso o banco de dados PEDIDOS.*/
USE PEDIDOS;

/* 02- Calcular a média de preço de venda (PRECO_VENDA)
   do cadastro de PRODUTOS. */
SELECT AVG(PRECO_VENDA) AS PRECO_MEDIO
FROM TB_PRODUTO;

/* 03- Calcular a quantidade de pedidos cadastrados em janeiro de 2014,
   o maior e o menor valor total (VLR_TOTAL).*/
SELECT COUNT(*) AS QTD_PEDIDOS,
       MAX(VLR_TOTAL) AS MAIOR_PEDIDO,
       MIN(VLR_TOTAL) AS MENOR_PEDIDO
FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31';

/* 04- Calcular o valor total vendido (soma de tb_PEDIDO.VLR_TOTAL)
   em janeiro de 2014. */
SELECT SUM(VLR_TOTAL) AS TOT_VENDIDO
FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31';

/* 05- Calcular o valor total vendido pelo vendedor de
   código 1 em janeiro de 2014. */
SELECT SUM(VLR_TOTAL) AS TOT_VENDIDO
FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
      CODVEN = 1;

/* 06- Calcular o valor total vendido pela
   vendedora 'LEIA' em janeiro de 2014. */
SELECT SUM(P.VLR_TOTAL) AS TOT_VENDIDO
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
      V.NOME = 'LEIA';

/* 07- Calcular o valor total vendido pelo vendedor
   'MARCELO' em janeiro de 2014. */
SELECT SUM(P.VLR_TOTAL) AS TOT_VENDIDO
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
      V.NOME = 'MARCELO';

/* 08- Calcular o valor da comissão
   (soma de TB_PEDIDO.VLR_TOTAL * TB_VENDEDOR.PORC_COMISSAO/100)
   que a vendedora 'LEIA' recebeu em janeiro de 2014. */
SELECT SUM(P.VLR_TOTAL * V.PORC_COMISSAO / 100 ) AS TOT_VENDIDO
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
      V.NOME = 'LEIA';
```

```sql
/* 09- Calcular o valor da comissão que o
   vendedor 'MARCELO' recebeu em janeiro de 2014. */
SELECT SUM(P.VLR_TOTAL * V.PORC_COMISSAO / 100 ) AS TOT_VENDIDO
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31' AND
      V.NOME = 'MARCELO';

/* 10- Listar os totais vendidos por cada vendedor
   (mostrar TB_VENDEDOR.NOME e a soma de TB_PEDIDO.VLR_TOTAL)
   em janeiro de 2014. Deve exibir o nome do vendedor. */
SELECT V.CODVEN, V.NOME, SUM(P.VLR_TOTAL) AS TOT_VENDIDO
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY V.CODVEN, V.NOME;

/* 11- Listar o total comprado por cada cliente
   em janeiro de 2014. Deve mostrar o nome do cliente. */
SELECT C.CODCLI, C.NOME, SUM(P.VLR_TOTAL) AS TOT_COMPRADO
FROM TB_PEDIDO P
     JOIN TB_CLIENTE C ON P.CODCLI = C.CODCLI
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY C.CODCLI, C.NOME;

/* 12- Listar o valor e a quantidade total vendida
   de cada produto em janeiro de 2014. */
SELECT
   PR.ID_PRODUTO, PR.DESCRICAO,
   SUM( I.QUANTIDADE ) AS QTD_TOTAL,
   SUM( I.PR_UNITARIO * I.QUANTIDADE ) AS VLR_TOTAL
FROM TB_ITENSPEDIDO I
   JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO
   JOIN TB_PEDIDO PE ON I.NUM_PEDIDO = PE.NUM_PEDIDO
WHERE PE.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY PR.ID_PRODUTO, PR.DESCRICAO;

/* 13- Listar os totais vendidos por cada vendedor em janeiro de 2014
   Deve exibir o nome do vendedor e mostrar apenas os vendedores que
   venderam mais de R$ 80.000,00. */
-- CÓPIA DO EXERC 9 COM HAVING
SELECT V.CODVEN, V.NOME, SUM(P.VLR_TOTAL) AS TOT_VENDIDO
FROM TB_PEDIDO P
     JOIN TB_VENDEDOR V ON P.CODVEN = V.CODVEN
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY V.CODVEN, V.NOME
    HAVING SUM(P.VLR_TOTAL) > 80000
ORDER BY 3;

/* 14- Listar o total comprado por cada cliente em janeiro de 2014.
   Deve mostrar o nome do cliente e somente os clientes que
   compraram mais de R$ 6.000,00. */
-- CÓPIA DO EXERC 10 COM HAVING
SELECT C.CODCLI, C.NOME, SUM(P.VLR_TOTAL) AS TOT_COMPRADO
FROM TB_PEDIDO P
     JOIN TB_CLIENTE C ON P.CODCLI = C.CODCLI
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY C.CODCLI, C.NOME
    HAVING SUM(P.VLR_TOTAL) > 6000
ORDER BY 3;
```

```sql
/* 15- Listar o total vendido de cada produto em janeiro de 2014
   Deve mostrar apenas os produtos que venderam mais de
   R$ 16.000,00. */
-- CÓPIA DO EXERC 11 COM HAVING
SELECT
   PR.ID_PRODUTO, PR.DESCRICAO,
   SUM( I.PR_UNITARIO * I.QUANTIDADE ) AS VLR_TOTAL
FROM TB_ITENSPEDIDO I
   JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO
   JOIN TB_PEDIDO PE ON I.NUM_PEDIDO = PE.NUM_PEDIDO
WHERE PE.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY PR.ID_PRODUTO, PR.DESCRICAO
    HAVING SUM( I.PR_UNITARIO * I.QUANTIDADE ) > 16000
ORDER BY 3;

/* 16- Listar o total comprado por cada cliente em janeiro de 2014.
   Deve mostrar o nome do cliente e somente os 10 primeiros do ranking. */
-- CÓPIA DO EXERC 10 COM TOP n + ORDER BY
SELECT TOP 10
  C.CODCLI, C.NOME, SUM(P.VLR_TOTAL) AS TOT_COMPRADO
FROM TB_PEDIDO P
     JOIN TB_CLIENTE C ON P.CODCLI = C.CODCLI
WHERE P.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY C.CODCLI, C.NOME
ORDER BY 3 DESC;

/* 17- Listar o total vendido de cada produto em janeiro de 2014.
   Deve mostrar os 10 produtos que mais venderam. */
-- CÓPIA DO EXERC 11 COM TOP n + ORDER BY
SELECT TOP 10
   PR.ID_PRODUTO, PR.DESCRICAO,
   SUM( I.QUANTIDADE ) AS QTD_TOTAL,
   SUM( I.PR_UNITARIO * I.QUANTIDADE ) AS VLR_TOTAL
FROM TB_ITENSPEDIDO I
   JOIN TB_PRODUTO PR ON I.ID_PRODUTO = PR.ID_PRODUTO
   JOIN TB_PEDIDO PE ON I.NUM_PEDIDO = PE.NUM_PEDIDO
WHERE PE.DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
GROUP BY PR.ID_PRODUTO, PR.DESCRICAO
ORDER BY 4 DESC;

-----------------------------------------------------
```

```sql
/*   18- Listar o total vendido em cada um dos meses de 2013.*/
SELECT MONTH( DATA_EMISSAO ) AS MES,
       YEAR( DATA_EMISSAO ) AS ANO,
       SUM( VLR_TOTAL ) AS TOT_VENDIDO
FROM TB_PEDIDO
WHERE YEAR(DATA_EMISSAO) = 2013
GROUP BY MONTH( DATA_EMISSAO ), YEAR( DATA_EMISSAO )
ORDER BY MES;
```

```sql
--1
USE PEDIDOS

--2
SELECT NOME, SALARIO, DATA_ADMISSAO,
       CASE DATEPART(MONTH,DATA_ADMISSAO)
        WHEN 1 THEN 'Janeiro'
        WHEN 2 THEN 'Fevereiro'
        WHEN 3 THEN 'Março'
        WHEN 4 THEN 'Abril'
        WHEN 5 THEN 'Maio'
        WHEN 6 THEN 'Junho'
        WHEN 7 THEN 'Julho'
        WHEN 8 THEN 'Agosto'
        WHEN 9 THEN 'Setembro'
        WHEN 10 THEN 'Outubro'
        WHEN 11 THEN 'Novembro'
        WHEN 12 THEN 'Dezembro'
       END AS MES
FROM tb_EMPREGADO;

--3
SELECT NOME ,
CONCAT(ENDERECO, ' - ' , BAIRRO , ' - ' , CIDADE , '/', ESTADO) AS ENDERECO
FROM tb_CLIENTE

--4
SELECT SUBSTRING(NOME, 1, CHARINDEX(' ' , NOME)) as Nome,
FORMAT( DATA_NASCIMENTO , 'dd/MM' ) AS [Aniversário]
FROM tb_EMPREGADO

--6
SELECT * FROM TBPROD
INTERSECT
SELECT * FROM TB_PRODUTO

--7
SELECT * FROM TBPROD
EXCEPT
SELECT * FROM TB_PRODUTO
```

Laboratório 2 das Aulas 28 a 30

```
/* 1. Usar o banco de dados PEDIDOS. */
USE PEDIDOS;

/* 2. Listar todos os pedidos com data de
      emissão anterior a Jan/2014. */
-- SOLUÇÃO 1
SELECT * FROM TB_PEDIDO
WHERE DATA_EMISSAO < '2014.1.1';
-- SOLUÇÃO 2
SELECT * FROM TB_PEDIDO
WHERE YEAR( DATA_EMISSAO ) < 2014;

/* 3. Listar todos os pedidos com data de
      emissão no primeiro semestre de 2014. */
SELECT * FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.6.30'
ORDER BY DATA_EMISSAO;
------- JANEIRO
SELECT * FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31';
------- JUNHO
SELECT * FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.6.1' AND '2014.6.30';

/* 4. Listar todos os pedidos com data de
      emissão em janeiro e junho de 2014. */
SELECT * FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
      OR
      DATA_EMISSAO BETWEEN '2014.6.1' AND '2014.6.30';

/* 5. Listar todos os pedidos do Vendedor
      Código 1 em Jan/2014. */
SELECT * FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
      AND CODVEN = 1;

/* 6. Listar os pedidos emitidos em Jan/2014
      em uma sexta-feira. */
SELECT * FROM TB_PEDIDO
WHERE DATA_EMISSAO BETWEEN '2014.1.1' AND '2014.1.31'
      AND DATEPART( WEEKDAY,DATA_EMISSAO) = 6;
```