Fundação CECIERJ - Vice Presidência de Educação Superior a Distância

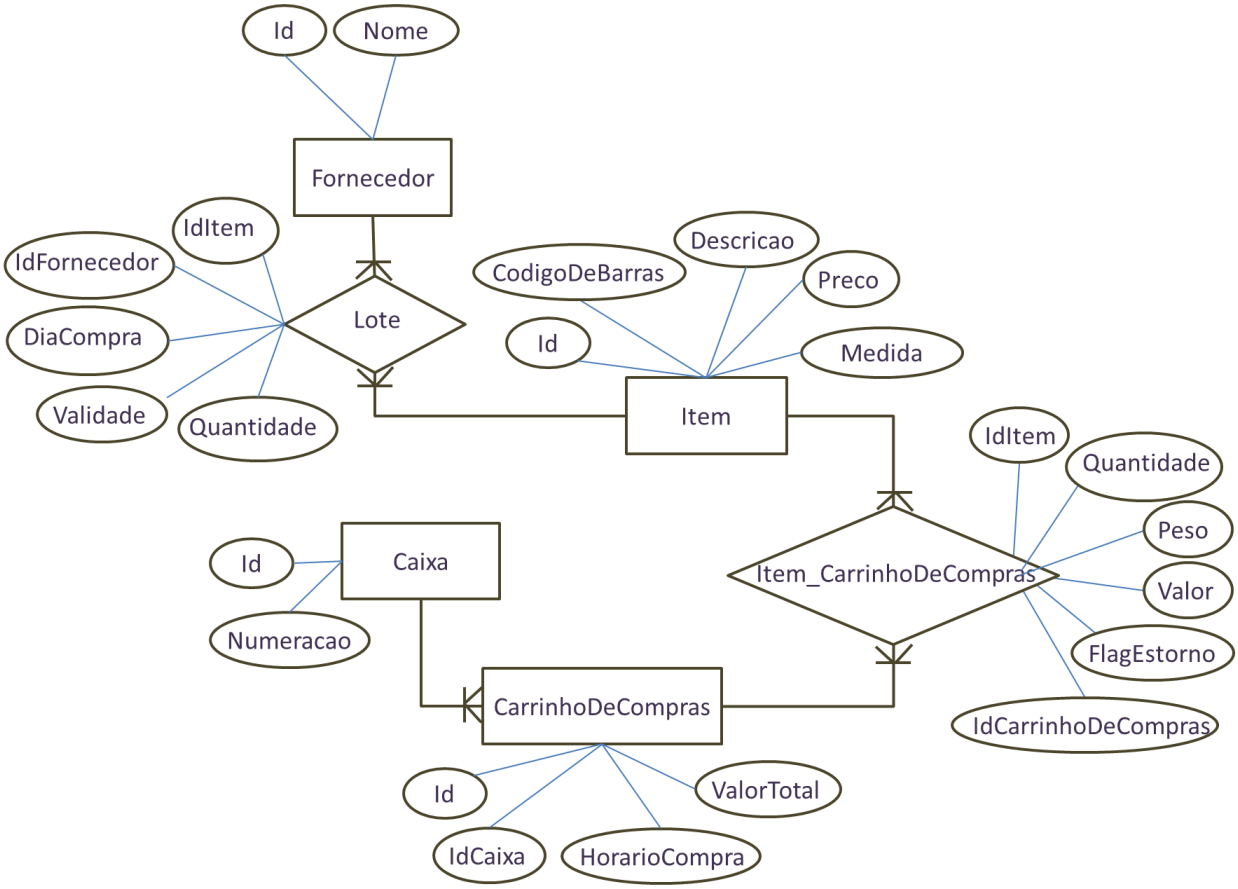
Curso de Tecnologia em Sistemas de Computação.

Disciplina: Programacao II

AD2 - 2° semestre de 2011 - GABARITO

Execício 1:

a)



b) Relativo ao modelo físico do trabalho anterior, as novas tabelas necessárias são:

```
CREATE TABLE IF NOT EXISTS `fornecedor` (
  `Id` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `Nome` varchar(255) CHARACTER SET utf8 NOT NULL,
  PRIMARY KEY (`Id`)
);
CREATE TABLE IF NOT EXISTS `lote` (
```

```
  `IdItem` int(11) unsigned NOT NULL,
  `IdFornecedor` int(11) unsigned NOT NULL,
  `DiaCompra` date NOT NULL,
  `Validade` date NOT NULL,
  `Quantidade` float(10,4) NOT NULL,
  PRIMARY KEY (`IdItem`,`IdFornecedor`,`DiaCompra`),
  FOREIGN KEY (`IdFornecedor`) REFERENCES `fornecedor` (`Id`),
  FOREIGN KEY (`IdItem`) REFERENCES `item` (`Id`)
);
```

Exercício 2:

a) Para saber se possuímos o item 123 em estoque, precisamos garantir que:

- existe lote com IdItem = 123 **e**
- dentro da validade (validade > dia de hoje) **e**
- ainda não foi vendido (assumindo que sempre os lotes mais velhos são vendidos primeiro, toma-se a soma da quantidade de todos os lotes que atendam os 2 critérios anteriores, e substrai-se das quantidades de todos os Item_CarrinhoDeCompra cujo Carrinho de Compra tenha HorarioCompra compreendido dentro deste mesmo período)

Feito com uma única consulta, que retorne TRUE caso haja o item em estoque ou retorne zero registros caso contrário, ficaria:

```
SELECT TRUE
FROM   (SELECT SUM(quantidade) as soma,
                    MIN(diacompra) as lotemaisantigo
             FROM lote
            WHERE IdItem = 123
              AND `Validade` > NOW()) as consulta
WHERE (SELECT SUM(quantidade)
             FROM item_carrinhodecompra ic,
                    carrinhodecompras cc
            WHERE ic.IdCarrinhoDeCompras = cc.Id
              AND cc.`HorarioCompra` >= consulta.lotemaisantigo)
           > consulta.soma
```

No entanto, respostas com 3 ou mais consultas parciais, e código PHP auxiliar também serão aceitas.

b) Este item é uma variação do item ‘a’, já que precisamos do lote mais antigo para poder calcular se temos ou não o item em estoque. Basta calcularmos a menor data e retorná-la ao usuário, como em:

SELECT consulta.lotemaisantigo FROM (SELECT MIN(diacompra) as lotemaisantigo FROM lote WHERE IdItem = 123 AND `Validade` > NOW()) as consulta

c) Neste item, precisamos tomar cuidado para não excluir lotes que já tenham sido vendidos. De fato em um sistema real, esta função jamais deveria existir, pois mesmo os itens vencidos representam informações importantes de histórico.
Quem responder aqui simplesmente algo como “DELETE FROM lote WHERE Validade < NOW()” estará errado. O uso de código PHP correto aliviará o peso do erro.

As etapas a serem seguidas para executar tudo em uma única consulta serão:

- escolher lotes em que a validade é inferior ao dia de hoje
- fazer o saldo de vendas: a soma acumulada dos lotes e subtrair da soma acumulada das vendas (ou seja somar todas as vendas de um determinado produto em estoque, começando de uma data início pré-definida até o dia de hoje e subtrair da soma de todas as vendas de Item_CarrinhoDeCompra cujo CarrinhodeCompra tenha HorarioCompra compreendido dentro deste mesmo período)
- excluir em um loop os lotes mais novos e parar quando o saldo de vendas for se tornar negativo
- Fazer isso para todos os itens em estoque

Para simplificar o processo, fixaremos no item 123. Também definiremos uma data de início, explicada melhor a seguir. Como sugestão, foi escolhido um intervalo de 100 dias

SELECT * FROM `lote` WHERE `Validade` < NOW() AND `IdItem` = 123 AND `DiaCompra` > DATE_SUB(NOW() , INTERVAL 100 DAY) ORDER BY DiaCompra DESC

Em seguida, calcula-se o saldo de vendas. Uma data início deve ser passada como parâmetro, do contrário, ele irá varrer todos os produtos já estocados e todas as transações de venda, e isto seria custoso ao banco. Se este parâmetro foi omitido na resposta de vocês, não se preocupem pois não será descontado.

Para o saldo:

```
SELECT consulta1.soma - consulta2.soma
FROM
( SELECT SUM(`Quantidade`) as soma
   FROM `lote`
  WHERE `Validade` < NOW()
    AND `IdItem` = 123
    AND `DiaCompra` > DATE_SUB(NOW(), INTERVAL 100 DAY)
```

```
 ) consulta1,

( SELECT SUM(quantidade) as soma
   FROM item_carrinhodecompra ic,
            carrinhodecompras cc
 WHERE ic.IdCarrinhoDeCompras = cc.Id
     AND cc.`HorarioCompra` >= DATE_SUB(NOW(), INTERVAL 100 DAY)
) consulta2
```

OBS: Caso consulta1.soma ou consulta2.soma for NULL, a consulta poderá incorrer em uma exceção que deve ser tratada.

Por último, armazenado o resultado da consulta anterior em uma variável $saldo, deve-se iterar nos registros da primeira consulta que estão ordenados por DiaCompra, e excluí-los pelo id (Ex: DELETE FROM lote WHERE id = Y), do mais recente para o mais antigo, enquanto a diferença: “saldo – quantidade do próximo lote da iteração” for positiva.