Teste de SQL

Considere a seguinte tabela:

|  |  |  |
| --- | --- | --- |
| Tabela de produtos | | |
| Campo | Tipo de Campo | Chave |
| cod\_prod | Integer (8) | X |
| loj\_prod | Integer (8) | X |
| desc\_prod | Char (40) |  |
| dt\_inclu\_prod | Data (dd/mm/yyyy) |  |
| preco\_prod | decimal (8,3) |  |

Com base na tabela de “produtos” acima favor inserir um registro na referida tabela passando os seguintes valores : cod\_prod =170, loj\_prod=2, desc\_prod=LEITE CONDESADO MOCOCA, dt\_inclu\_prod=30/12/2010 e preço\_prod = R$45,40.

**INSERT INTO produtos VALUES (‘17’, ‘2’, ‘LEITE CONDENSADO MOCOCA’, ‘30/12/2010’, ‘45.40’);**

O Índice da tabela de “produtos é o cód\_prod e a loj\_prod, com base no referido índice faça a alteração do preço do produto para R$95,40, lembrando que o cod\_prod =170 e a loj\_prod=2:

**UPDATE produtos SET preco\_prod = ‘95.40’ WHERE cod\_prod = 170 AND loj\_prod = 2;**

Com base na tabela de “produtos” monte um select trazendo todos os registros da loja 1 e 2:

**SELECT \* FROM produtos WHERE loj\_prod = 1 OR loj\_prod = 2;**

Com base na tabela de “produtos” monte um select para trazer a maior e a menor data de inclusão do produto “dt\_inclu\_prod”:

**SELECT MIN(dt\_inclu,prod), MAX(dt\_inclu\_prod) FROM produtos;**

Com base na tabela de “produtos” monte um select para trazer a quantidade total de registros existentes na tabela de “produtos”:

**SELECT COUNT(\*) FROM produtos;**

Com base na tabela de “produtos” monte um select para trazer todos os produtos que comecem com a letra “L” na tabela de “produtos”:

**SELECT \* FROM produtos WHERE desc\_prod LIKE 'L%';**

Com base na tabela de “produtos” monte um select para trazer a soma de todos os preços dos produtos totalizado por loja:

**SELECT loj\_prod, SUM(preco\_prod) FROM produtos GROUP BY loj\_prod;**

Com base na tabela de “produtos” monte um select para trazer a soma de todos os preços dos produtos totalizados por loja que seja maior que R$100.000

**SELECT loj\_prod, SUM(preco\_prod) FROM produtos GROUP BY loj\_prod HAVING SUM(preco\_prod) > 100000;**

Observe as Tabelas Abaixo:

|  |  |  |  |
| --- | --- | --- | --- |
| Tabela de Produtos | | | |
| Campo | Tipo de Campo | Chave | Comentário |
| Cód\_prod | Integer (8) | X | Código do Produto |
| loj\_prod | Integer (8) | X | Código da Loja |
| desc\_prod | Char (40) |  | Descrição do Produto |
| Dt\_inclu\_prod | Data (dd/mm/yyyy) |  | Data de Inclusão do Produto |
| preco\_prod | decimal (8,3) |  | Preço do Produto |
| Tabela de Estoque | | | |
| Campo | Tipo de Campo | Chave | Comentário |
| Cód\_prod | Integer (8) | X | Código do Produto |
| loj\_prod | Integer (8) | X | Código da Loja |
| qtd\_prod | decimal(15,3) |  | Quantidade em Estoque do Produto |
|  |  |  |  |
|  |  |  |  |
| Tabela de Lojas | | | |
| Campo | Tipo de Campo | Chave | Comentário |
| loj\_prod | Integer (8) | X | Código da Loja |
| desc\_loj | Char (40) |  | Descrição da Loja |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

A)Montar um unico select para trazer os seguintes campos: o código da loja do produto, a descrição da loja, código do produto, a descrição do produto, o preço do produto, a quantidade em estoque do produto. Considere que o código da loja para esta consulta seja igual a 1.

**SELECT p.loj\_prod, l.desc\_loj, p.cod\_prod, p.desc\_prod, p.preco\_prod, e.qtd\_prod FROM produtos p INNER JOIN lojas l ON l.loj\_prod = p.loj\_prod INNER JOIN estoque e ON e.loj\_prod = l.loj\_prod WHERE p.loj\_prod =1;**

B)Observe a estrutura da tabela de estoque e da tabela de produtos, monte um select para trazer todos os produtos que existem na tabela de produtos que não existem na tabela de estoque.

**SELECT \* FROM produtos p WHERE NOT EXISTS (SELECT e.cod\_prod FROM estoque e WHERE e.cod\_prod = p.cod\_prod);**

C)Observe a estrutura da tabela de estoque e da tabela de produtos, monte um select para trazer todos os produtos que existem na tabela de estoque que não existem na tabela de produtos.

**Obs: Pela concepção geral do modelo tanto o cod\_prod e loj\_prod de Estoque são chaves estrangeiras que impossibilita o cadastramento em Estoque um Produto inexistente, mas se ambos os campos são chaves primárias ha um erro conceitual no modelo.**

**O SELECT proposto deve ter a função de retornar inconsistências da tabela Estoque, mas de forma geral deve ser sempre nulo.**

**SELECT \* FROM estoque e WHERE NOT EXISTS (SELECT p.cod\_prod FROM produtos p WHERE p.cod\_prod = e.cod\_prod);**