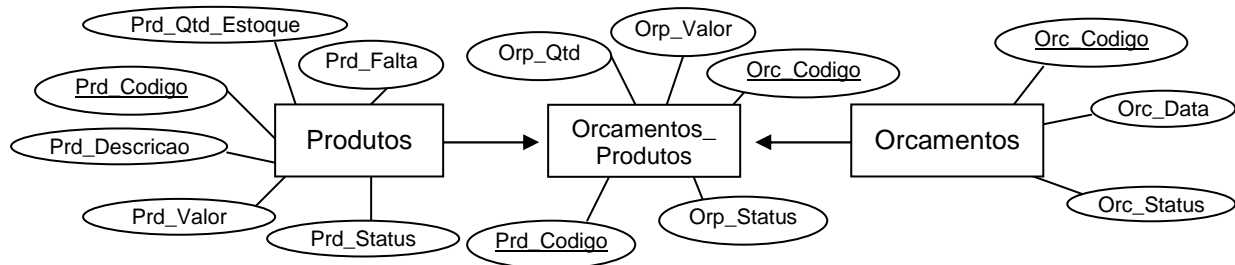


## Triggers – Exercícios

Considere o diagrama abaixo:



Elabore:

1. Faça um *trigger* para armazenar em uma tabela chamada `produtos_atualizados` (`prd_codigo`, `prd_qtd_anterior`, `prd_qtd_atualizada`, `prd_valor`) quando ocorrer quaisquer alterações nos atributos da tabela `produtos`. No entanto, caso a alteração atribua o valor zero para o atributo `prd_qtd_estoque`, a tabela `produtos_em_falta` deverá ser alimentada com as mesmas informações da tabela produto, exceto o atributo `prd_valor`. Além disso, o atributo `prd_status` do produto atualizado deve ser modificado para `NULL` e o atributo `orp_status` de todos os `orcamentos_produtos` desse produto deverá ser modificado também para `NULL`.
2. Faça um *trigger* para armazenar em uma tabela chamada `Histórico_Produtos_Excluídos` (`prd_codigo`, `prd_qtd_estoque`, `prd_preco_venda`) todos os produtos que foram excluídos da tabela `Produtos`, mais a informação de qual usuário do sistema realizou a exclusão e em qual data e hora. Atenção, essa *trigger* somente excluirá os produtos se eles **não** tiverem quantidades em estoque (`prd_qtd_estoque`). Caso isso aconteça a tabela chamada `TentativasLog` (`data`, `operacao`, `prd_codigo`, `usuario_bd`) é alimentada com os dados dos produtos que seriam excluídos, ou seja, com os código dos produtos que estavam no cursor. Além disso, o atributo `prd_status` desses produtos (que seriam excluídos) terão seu valor atualizado para 0 (zero), que indica que um produto deve ser vendido rapidamente.



3. Faça um trigger para que no momento do cancelamento de orçamento (`orc_status` é modificado para 0), a tabela `orcamento_produtos_cancelados` seja alimentada com as informações da tabela `orcamentos_produtos` e todos os `orcamentos_produtos` do orçamento cancelado devem ser fisicamente excluídos. Além disso, a tabela `orcamentos_cancelados` deve ser alimentada com as mesmas informações do orçamento cancelado e, o orçamento cancelado deve ser excluído fisicamente. Dessa forma, nas tabelas do diagrama acima somente ficarão os `orcamentos` confirmados ou a serem confirmados.
  
4. Faça um trigger que após a confirmação de orçamento (`orc_status` é modificado para 1), seja verificado em cada `orcamentos_produtos` se a quantidade orçada (`orp_qtd`) é suficiente no estoque (`prd_qtd_estoque`). Caso afirmativo, o atributo `orp_status` deve ser modificado para 1. Caso contrário, o atributo `orp_status` deve ser modificado para 2 e a tabela `produtos_requisitados` (`prd_codigo`, `orc_codigo`, `qtd_em_falta`, `orp_valor`, `data`, `usuario_sistema`) é alimentada. Dessa forma, essa tabela manterá todos os produtos que já foram vendidos e que não tinham no estoque no momento da confirmação do orçamento. O atributo `qtd_em_falta` é a diferença da quantidade orçada e a quantidade no estoque.
  
5. Faça um *trigger* que após a atualização do estoque de produtos faça uma verificação na tabela `orcamentos_produtos` para verificar quais os `orcamentos` do produto que foi atualizado estão com o status (`orp_status`) com o valor 2. Para esses casos, o trigger deverá efetuar a baixa do estoque considerando o atributo `orp_qtd` registrado para cada `orcamento` desse produto.
  
6. Faça uma *trigger* que faça o cálculo da quantidade necessária para atender os `orcamentos` dos produtos registrados na tabela `produtos_requisitados` (`prd_codigo`, `orc_codigo`, `qtd_em_falta`, `orp_valor`, `data`, `usuario_sistema`). Dessa forma, a quantidade necessária será armazenada no atributo `prd_falta` da tabela de produtos.



7. Faça um *trigger* para armazenar em uma tabela chamada Histórico\_Precos (prd\_codigo, data, prd\_preco\_antigo, prd\_preco\_novo, usuario\_bd, usuario\_sistema) as alterações de preços ocorridas nos produtos cadastrados na tabela Produtos (prd\_codigo, prd\_descricao, prd\_qtd, prd\_preco, prd\_status). Atenção, esse *trigger* somente deverá ser disparado quando houver alteração no atributo prd\_preco da tabela produtos e, deve permitir as alterações de preços de mais de um produto.
  
8. Em uma operação de atualização dos dados de produtos, implemente um *trigger* que armazene os dados (registro inteiro) a serem modificados em uma tabela denominada dados\_produtos\_antigos e, os novos dados (registro inteiro) em uma tabela denominada dados\_produtos\_novos. Em ambas as tabelas (dados\_produtos\_antigos e dados\_produtos\_novos) a estrutura dos atributos é a mesma da tabela produtos.
  
9. Faça um *trigger* para que seja disparado na operação de exclusão da tabela de produtos. Esse trigger não permitirá que seja feita a exclusão física, mas alterará o valor do atributo prd\_status para 4, de um ou mais produtos a serem excluídos.
  
10. Faça um *trigger* para armazenar em uma tabela chamada Histórico\_Produtos\_Excluídos (prd\_codigo, prd\_descricao, prd\_qtd, prd\_preco\_venda) todos os produtos que foram excluídos da tabela Produtos (prd\_codigo, prd\_descricao, prd\_qtd, prd\_preco\_venda), mais a informação de qual usuário do sistema realizou a exclusão e em qual data e hora. Atenção, essa *trigger* somente excluirá os produtos se eles **não** tiverem quantidades em estoque (prd\_qtd). Caso isso aconteça a tabela chamada TentativasLog (data, operação, prd\_codigo, usuário\_bd) é alimentada com os dados dos produtos que seriam excluídos, ou seja, com os código dos produtos que estavam no cursor.