Avaliação de Banco de Dados II

1 Sistema de Biblioteca (Nível Básico)

Crie procedures garantindo que transações e triggers sejam usados para manter a integridade dos dados, para as implementar as seguintes funcionalidades:

- Cadastrar Livro
- Atualizar Livro
- Consultar Livro
- Excluir Livro
- Emprestar Livro

  Observações: Certifique-se de que cada uma dessas tarefas seja executada com cuidado, garantindo a integridade e a consistência dos dados. Para operações de exclusão e atualização, é crucial verificar as condições necessárias antes de prosseguir com as mudanças no banco de dados. Em caso de erros, utilize exceções.

Criação das Tabelas Crie uma tabela `livros` com os seguintes campos:

- `id` (chave primária, inteiro, autoincremento)
- `titulo` (texto)
- `autor` (texto)
- `ano_publicacao` (inteiro)
- `disponivel` (booleano, padrão `true`)

Crie uma tabela `emprestimos` com os seguintes campos:

- `id_emprestimo` (chave primária, inteiro, autoincremento)
- `livro_id` (inteiro, chave estrangeira para `livros`)
- `data_emprestimo` (data)
- `data_devolucao` (data, pode ser nula)

Criação das Procedures:

- Cadastrar Livro

  Escreva uma procedure para inserir um novo livro na tabela `livros`. O livro deve incluir os seguintes detalhes:

  Verifique se todos os campos necessários estão preenchidos antes de inserir o livro na tabela.

- Atualizar Livro

  Escreva uma procedure de atualização do título e outra do ano de publicação.

Garanta que o livro especificado exista antes de tentar atualizá-lo. Caso o livro não exista, a operação não deve ser realizada.

- Consultar Livro

  Escreva uma procedure para buscar todos os detalhes de um livro. A consulta deve retornar todos os campos disponíveis para o livro, incluindo seu status de disponibilidade.

- Excluir Livro

  Escreva uma procedure para excluir um livro da tabela `livros` com base, tomando como parâmetro o seu id.

  Antes de excluir o livro, verifique se ele não está atualmente emprestado. Se o livro estiver emprestado (campo `disponivel` igual a `false`), a exclusão não deve ser realizada. Informe o usuário sobre a impossibilidade de exclusão se o livro estiver emprestado.

- Empréstimo

  Desenvolva uma procedure chamada `realizar_emprestimo` que aceite `livro_id` e `data_emprestimo`. A procedure deve verificar se o livro está disponível. Se estiver, o empréstimo deve ser registrado e o livro marcado como não disponível.

Criação de Trigger Implemente um trigger que, ao inserir um novo registro na tabela `emprestimos`, automaticamente atualize o campo `disponivel` na tabela `livros` para `false`.

---

2 Controle de Estoque (Nível Intermediário)

Desenvolva um sistema de controle de estoque que use transações para garantir que operações sejam atômicas e que use triggers para manter a consistência dos dados.

Criação das Tabelas:

- `produtos`:
  - `id_produto` (chave primária, inteiro, autoincremento)
  - `nome` (texto)
  - `quantidade_estoque` (inteiro)
  - `preco` (decimal)
- `vendas`:
  - `id_venda` (chave primária, inteiro, autoincremento)
  - `produto_id` (inteiro, chave estrangeira para `produtos`)
  - `quantidade` (inteiro)
  - `data_venda` (data)

Criação de procedures

- Crie procedures para inserir, consultar, atualizar e apagar produtos. Caso o produto já tenha sido vendido, não deve ser possível apagá-lo.
- Criar Procedure para Venda:
  - Desenvolva uma procedure chamada `registrar_venda` que aceite `produto_id`, `quantidade` e `data_venda`. A procedure deve verificar se há estoque suficiente para o produto. Se houver, a venda deve ser registrada e o estoque atualizado.

Implementar Transações

- Assegure que as operações dentro da procedure `registrar_venda` sejam atômicas. Use transações para garantir que, ou todas as operações são realizadas com sucesso, ou nenhuma alteração é feita no banco de dados em caso de falha.

Criação de Trigger de Reabastecimento

- Crie um trigger que, sempre que a `quantidade_estoque` de um produto cair abaixo de um determinado limiar, uma notificação seja gerada (por exemplo, inserindo em uma tabela de `alertas`).

---

3 Sistema de Reservas de Hotéis (Nível Avançado)

Implemente um sistema de reservas para um hotel, utilizando transações, procedures complexas, triggers e funções de agregação condicional para gestão avançada de dados.

Criação das Tabelas

- `quartos`:
  - `id_quarto` (chave primária, inteiro, autoincremento)
  - `numero` (inteiro)
  - `tipo` (texto, ex.: 'simples', 'duplo', 'suíte')
  - `preco_diaria` (decimal)
  - `status` (texto, valores: 'disponível', 'ocupado', 'manutenção')
- `clientes`:
  - `id_cliente` (chave primária, inteiro, autoincremento)
  - `nome` (texto)
  - `email` (texto, único)
  - `telefone` (texto)
- `reservas`:
  - `id_reserva` (chave primária, inteiro, autoincremento)
  - `cliente_id` (inteiro, chave estrangeira para `clientes`)
  - `quarto_id` (inteiro, chave estrangeira para `quartos`)
  - `data_entrada` (data)
  - `data_saida` (data)
  - `valor_total` (decimal, calculado automaticamente)

Procedures para quartos e clientes

- Desenvolva as procedures para as operações de criação, consulta, atualização e deleção nas tabelas quartos e clientes.
- Se o cliente já tiver feito reservas, não pode ser apagado.
- Se o quarto já tiver sido utilizado, não pode ser apagado.

Criação de Procedure para Reserva

- Desenvolva uma procedure chamada `efetuar_reserva` que aceite `cliente_id`, `quarto_id`, `data_entrada` e `data_saida`. A procedure deve:
  - Verificar se o quarto está disponível para todo o período da reserva.
  - Calcular o valor total da reserva multiplicando o número de dias pela diária.
  - Registrar a reserva e atualizar o status do quarto para 'ocupado'.
  - Usar transações para garantir que todas as operações sejam concluídas com sucesso ou revertidas em caso de falha.

Implementação de Triggers

- Crie um trigger que atualize o status do quarto para 'disponível' quando uma reserva chegar ao fim (ou seja, na `data_saida`).
- Implemente um trigger que detecte alterações na tabela `quartos` (por exemplo, mudança de `status` para 'manutenção') e atualize automaticamente as reservas futuras, notificando os clientes afetados por e-mail (simule isto inserindo um registro em uma tabela de `notificacoes`).

Funções de Agregação Condicionais

- Crie uma view `relatorio_reservas` que utilize funções de agregação para fornecer um sumário mensal de reservas, incluindo total de reservas, número de noites reservadas e receita total, agrupados por tipo de quarto.

Procedure de Cancelamento de Reserva:

- Desenvolva uma procedure `cancelar_reserva` que aceite `id_reserva` e efetue o cancelamento da reserva. Esta procedure deve ajustar o status do quarto e garantir que todas as mudanças sejam atômicas, usando transações.