**Atividade: Cap 1 – Ampliando a Consistência do Negócio e Analisando o Sigilo e Propriedade do Dados**

**Grupo:** Data Avengers

**Integrantes:**

Ana Clara da Silva Pinto RM: 555242

Luiz Otavio Batista Prado RM: 557590

Matheus Camilo Alves RM: 554449

Rafaela Oliveira de Souza RM: 554410

Ryan Rodrigues da Silva RM: 555790

**1.3 – Criar um bloco anônimo em Oracle PL/SQL para a empresa Melhores Compras**

**EVIDÊNCIAS**

1. **Execução dos arquivos ASSET**

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente com confiança médiaTabela

Descrição gerada automaticamenteAntes de iniciar o desenvolvimento do bloco anônimo, utilizamos os arquivos disponibilizados como *“Asset”* para iniciar nossa estrutura de banco de dados. O primeiro arquivo que usamos foi o **“apaga.sql”**, para eliminar as tabelas, constraints e sequences criadas anteriormente e recriá-las, garantindo a estrutura de acordo com o solicitado na atividade. Com o arquivo, eliminamos as seguintes tabelas:

Uma imagem contendo Interface gráfica do usuário

Descrição gerada automaticamenteTexto

Descrição gerada automaticamente com confiança média

Tabela

Descrição gerada automaticamente com confiança baixaCom as tabelas eliminadas, podemos recriá-las utilizando o arquivo “cria.sql”, disponibilizado também como “Asset” nessa fase. Após executar o script disponível no arquivo as seguintes tabelas foram criadas e populadas (exceto MC\_SGV\_OCORRENCIA\_SAC):

Para finalizar a população das tabelas, executamos o script “SCRIPT\_DML.sql” que criamos para atender a primeira requisição da atividade. Com ele, inserimos os seguintes dados na tabela “MC\_SGV\_SAC”.

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente com confiança média

Outra necessidade que tivemos foi a criação da tabela “MC\_ALIQUOTA\_MEDIA\_ICMS\_ESTADO”, com o script **“MC\_ALIQUOTA.sql”** disponibilizado para que pudéssemos utilizar a função que calcula o valor de ICMS dos produtos futuramente.

Com as tabelas prontas para uso, iniciamos o desenvolvimento do bloco anônimo.

1. **Desenvolvimento do bloco anônimo**

O primeiro passo foi declarar um CURSOR (o nomeamos como sac\_cursor) que realiza um SELECT nos dados das tabelas MC\_SGV\_SAC, MC\_PRODUTO e MC\_CLIENTE através de JOINS. O SELECT executado retorna os seguintes campos:

* Número da ocorrência do SAC (coluna nr\_sac na tabela mc\_sgv\_sac);
* Data de abertura do SAC (coluna dt\_abertura\_sac na tabela mc\_sgv\_sac);
* Hora de abertura do SAC (coluna hr\_abertura\_sac na tabela mc\_sgv\_sac);
* Tipo do SAC (coluna tp\_sac na tabela mc\_sgv\_sac);
* Código do produto (coluna cd\_produto na tabela mc\_produto);
* Nome do produto (coluna ds\_produto na tabela mc\_produto);
* Valor unitário do produto (coluna vl\_unitario na tabela mc\_produto);
* Percentual do lucro unitário do produto (coluna vl\_perc\_lucro na tabela mc\_produto);
* Número do Cliente (coluna nr\_cliente na tabela mc\_cliente);
* Nome do Cliente (coluna nm\_cliente na tabela mc\_cliente).

Texto

Descrição gerada automaticamente

Na sequência, declaramos algumas variáveis, conforme segue:

Texto

Descrição gerada automaticamente

* **v\_tipo\_sac:** variável declarada como VARCHAR(25), que posteriormente receberá o tipo de classificação do SAC (sugestão, dúvida, elogio ou classificação inválida);
* **v\_unitario\_lucro\_prod:** variável declarada utilizando o %TYPE, assim, recebendo o mesmo tipo de dado da coluna vl\_unitario da tabela MC\_PRODUTO. Essa variável receberá, posteriormente, o resultado de um cálculo.
* **v\_sg\_estado:** variável declarada utilizando o %TYPE, assim, recebendo o mesmo tipo de dado da coluna sg\_estado da tabela MC\_ESTADO. Essa variável receberá, posteriormente, a sigla do estado de acordo com o cliente.
* **v\_nm\_estado:** variável declarada utilizando o %TYPE, assim, recebendo o mesmo tipo de dado da coluna nm\_estado da tabela MC\_ESTADO. Essa variável receberá, posteriormente, o nome do estado de acordo com o cliente.
* **v\_aliquota\_icms:** variável declarada utilizando o %TYPE, assim, recebendo o mesmo tipo de dado da coluna vl\_perc\_aliquota\_media da tabela MC\_ALIQUOTA\_MEDIA\_ICMS\_ESTADO. Essa variável receberá, posteriormente, o valor da alíquota de ICMS de acordo com o estado.
* **v\_icms\_produto:** variável declarada utilizando o %TYPE, assim, recebendo o mesmo tipo de dado da coluna vl\_icms\_produto da tabela MC\_SGV\_OCORRENCIA\_SAC. Essa variável receberá, posteriormente, o resultado de um cálculo.
* **v\_cursor:** variável declarada utilizando o %ROWTYPE, assim, recebendo os mesmos tipos de dados e colunas do cursor SAC\_CURSOR. Essa variável será utilizada na iteração em LOOP.

Com todas as variáveis declaradas, partimos para o desenvolvimento do BEGIN.

Iniciamos verificando se o cursor SAC\_CURSOR está aberto. Se ele não estiver, será aberto. Se sim, continuamos com a sequência do código. Em seguida, iniciamos um LOOP que realizará a iteração nos registros recuperados no SELECT do cursor SAC\_CURSOR. Inicialmente, realizamos com o FETCH a recuperação do registro atual e atribuímos os valores para a variável **v\_cursor** declarada anteriormente. Para finalizar, se não for encontrado alguma tupla no SAC\_CURSOR (ou seja, se terminamos de iterar em todos os registros) sairemos do LOOP.

Texto

Descrição gerada automaticamente

Para atender o primeiro procedimento do item B, utilizamos o CASE para verificar o valor do tipo\_sac do cursor. O seguinte procedimento deveria ser atendido: Caso o conteúdo seja S o texto correto deve ser SUGESTÃO. Caso seja D o texto correto deve ser DÚVIDA. Caso seja E o texto correto deve ser ELOGIO. Caso não seja nenhuma dessa classificação retorne o texto CLASSIFICAÇÃO INVÁLIDA. Assim, verificamos cada uma das condições e o retorno é atribuído à variável v\_tipo\_sac, conforme segue:

Texto

Descrição gerada automaticamente

Atendendo o segundo procedimento do item B, realizamos o cálculo do lucro unitário sobre o produto ofertado. O valor foi calculado de acordo com os valores disponíveis no cursor e o resultado foi atribuído à variável v\_unitario\_lucro\_prod, conforme segue:

Texto

Descrição gerada automaticamente

Para atender o terceiro procedimento do item B, realizamos a recuperação das colunas SG\_ESTADO e NM\_ESTADO, ao qual referem-se ao estado onde o cliente se localiza. Para recuperar essa informação realizamos JOINS entre as tabelas MC\_END\_CLI, MC\_LOGRADOURO, MC\_BAIRRO, MC\_CIDADE e MC\_ESTADO. Os valores foram atribuídos às variáveis v\_sg\_estado e v\_nm\_estado. Para o tratamento de erros, incluímos uma EXCEPTION NO\_DATA\_FOUND, para o caso de não encontrar informações do estado do cliente, conforme segue:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Atendendo o quarto procedimento do item B, recuperamos a alíquota média de ICMS do estado do cliente utilizando a função disponibilizada fun\_mc\_gera\_aliquota\_media\_icms\_estado. A função recebe a variável v\_sg\_estado e, de acordo com o estado, retorna o valor da alíquota média de ICMS. Após recuperar o valor, calculamos o valor do ICMS do produto conforme a seguir:

Tela de celular com texto preto sobre fundo branco

Descrição gerada automaticamente

Após a finalização dos procedimentos requiridos entramos na parte final do bloco anônimo, onde vamos exibir as informações do cliente e inseri-las na tabela MC\_SGV\_OCORRENCIA\_SAC. Iniciamos o bloco BEGIN e usamos o INSERT para incluir os valores para cada coluna da tabela, utilizando os valores disponíveis no cursor e das variáveis que receberam valores calculados e recuperados de outras tabelas. Por fim, após a inclusão do registro na tabela e COMMIT, exibimos o valor de cada informação conforme solicitado. Segue o código desenvolvido:

Tabela

Descrição gerada automaticamente

Para evitar a violação de restrição de PK (nos casos em que o cliente já está cadastrado na tabela MC\_SGV\_OCORRENCIA\_SAC), incluímos uma EXCEPTION que, quando identificado que o valor já está cadastrado, realizamos o UPDATE das informações dele, ao invés do INSERT. Assim, evitamos o erro no código, conforme segue:

Tabela

Descrição gerada automaticamente

Após todas as operações e fim da iteração no LOOP, realizamos o fechamento do cursor com CLOSE e encerramos o bloco anônimo.

Executando o bloco anônimo, recebemos a seguinte saída:

Texto

Descrição gerada automaticamenteTexto

Descrição gerada automaticamente

Quando executamos novamente, com os registros já cadastrados na tabela MC\_SGV\_OCORRENCIA\_SAC, recebemos a mensagem de atualização, conforme segue:

Texto, Tabela

Descrição gerada automaticamente com confiança média

Tabela

Descrição gerada automaticamente

Por fim, após a execução do bloco anônimo, a tabela MC\_SGV\_OCORRENCIA\_SAC possui os seguintes registros:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente