Objetivo do trabalho
Esse documento foi elaborado pelo grupo 1 como material de apoio para a sala de aula invertida, cujo objetivo é inverter a metodologia tradicional de ensino. Em vez de o professor expor o conteúdo, passando conhecimento a um grupo de estudantes, a ideia é que esses estudantes tenham esse papel.

O que é Trigger?
Trigger, trata-se de um termo que traduzido do inglês significa “gatilho”, esse gatilho refere-se a uma função disparada dentro do banco de dados com propriedades de: alterar, atualizar e deletar dados.

⚠ É importante frisar que o trigger não está diretamente ligado com a ação feita na tabela, e sim que ele é um “Acionador” dessa ação! ⚠

Analogia com a pesca:
Para melhor compreensão, podemos fazer uma analogia com o ato de pescar. Para que o pescador consiga capturar um peixe, é necessário jogar o anzol na água e esperar o sinal de que ele mordeu a isca, para assim o pescador puxar a vara e capturar o peixe. De maneira semelhante, podemos associar o pescador com os Triggers. Eles esperam um sinal para serem disparados, e quando isso acontece executam uma determinada ação que definimos dentro da sintaxe do Trigger.

Imagem Ilustrativa

Classes de Triggers:
Existem duas classes de Triggers no SQL, sendo elas: Triggers DDL (Data Definition Language) e Triggers DML (Data Modification Language).

Triggers DDL (Data Definition Language) Triggers DML (Data Modification Language).
A classe dos triggers DDL é acionada em eventos que alteram a estrutura (CREATE), modificar (ALTER) ou soltar (DROP)) ou em determinados eventos relacionados ao servidor, como alterações de segurança ou atualização de eventos estatísticos. Os triggers DML é a classe de Triggers mais usada. Nesse caso, o evento de disparo é uma declaração de modificação de dados; poderia ser uma instrução de inserção, atualização ou exclusão em uma tabela ou em uma exibição.

Tipos de Triggers:
Um trigger pode ser de três tipos diferentes, sendo eles: FOR, AFTER e o INSTEAD OF.

FOR AFTER INSTEAD OF

É o valor padrão e indica que a trigger será executada JUNTO com o comando de INSERT, DELETE ou UPDATE que a disparou. Faz com que a Trigger seja executada APÓS a ação que a gerou ser concluída.

Faz com que o trigger seja executado NO LUGAR da ação que o gerou.

Orientações Básicas:
A sintaxe geral dos triggers é semelhante, o que diferencia é o modo de disparo.

Criação de um Trigger:
Podemos criar um Trigger usando o comando Create Trigger do SQL Server ou através do Enterprise Manager. A Sintaxe de um Trigger é a seguinte:

Imagem Ilustrativa
Para criação de um Trigger você deverá definir:

1. O nome.
2. A Tabela para o qual o Trigger irá ser criado.
3. Quando o Trigger deverá ser disparado.
4. Os comandos que determinam qual ação o Trigger deverá executar.

Habilitar ou Desabilitar um Trigger:
Para HABILITAR ou DESABILITAR triggers em uma tabela de uma forma geral, podemos utilizar o comando no DDL, o ALTER TABLE.

```
ALTER TABLE nome_tabela
ENABLE|DISABLE TRIGGER nome_trigger
```

Alterar um Trigger:
O comando ALTER suporta a alteração de vários objetos criados no banco de dados. Podemos facilmente chamar para alterar o conteúdo de TRIGGER, alterar a palavra CREATE para ALTER e, em seguida, executar o comando.

⚠ Para definir um gatilho, também podemos usar o comando COM ENCRYPTION, mas se você não se lembra do que foi escrito em TRIGGER, você não pode alterá-lo.⚠

```
ALTER TRIGGER
```

Dropar um Trigger:
Se você deseja excluir TRIGGER do banco de dados, use DROP TRIGGER mais o nome do TRIGGER que deseja excluir.

```
DROP TRIGGER
```

As vantagens de usar um Trigger:
O Trigger fornece uma forma alternativa de executar tarefas agendadas. Com trigger, você não tem que esperar para executar as tarefas agendadas. Você pode lidar com as tarefas antes ou após as alterações feitas para tabelas de banco de dados. Pode apenas fornecer validação estendida e não pode substituir todas as validações.

Exemplos Práticos:

Gerar alguns valores de coluna derivados automaticamente: Temos um banco com tabelas de produtos e vendas, quando um item é vendido e isso é referenciado na tabela de “produtos” poderíamos ter um gatilho de “vendido” nessa tabela, com base nisso, quando o trigger da tabela for acionado ele atualiza os dados na tabela de ”Vendas do mês” automaticamente.

Outro exemplo prático do uso do trigger seria em uma situação onde fosse enviado um email quando um aluno novo entrasse na escola, sendo que toda vez que um registro for inserido na tabela de aluno o trigger vai ser acionado, inserindo um registro na tabela de email e dessa forma enviando um email de confirmação do cadastro do aluno novo.

```
--Trigger para enviar um email ao aluno novo que foi cadastrado
ALTER TRIGGER trgEmailAluno
ON cadastroAluno
FOR INSERT
AS
BEGIN

DECLARE @nomeAluno VARCHAR(150), @nomeEscola VARCHAR(100);

SELECT @nomeAluno = nomeAluno, @nomeEscola = nomeEscola
FROM INSERTED

INSERT INTO email(destinatario, assunto, texto)
VALUES ('gustavo@gmail.com', 'Aluno novo cadastrado','O aluno ' + @nomeAluno + ' da escola ' + @nomeEscola + ' foi cadastrado com sucesso')
END;
GO

INSERT INTO cadastroAluno(nomeAluno, nomeEscola)
VALUES ('Gustavo', 'SENAI Santa Cecilia');
GO

select * from cadastroAluno
select * from email
```

Aplicar a integridade referencial: Em resumo, integridade referencial é um conceito de banco de dados que garante que todos os relacionamentos propostos entre tabelas no modelo de entidade-relacionamento (ER) serão respeitados dando a certeza que os dados de um banco de dados estarão íntegros. Este conceito diz que o valor que é chave estrangeira em uma tabela destino, deve ser chave primária de alguma tabela de origem, quando essa regra é desrespeitada, então temos o caso em que a integridade referencial é violada. Quando estamos criando as tabelas no banco de dados podemos acabar esquecendo de colocar as chaves estrangeiras nas tabelas que necessitam das mesmas, o que, consequentemente, não respeita a integridade referencial, como consequência disso alguns campos não podem ser apagados sem apagar a tabela toda, isso se aplica também

ao inserimento de dados ou a atualização dos mesmos, para prevenir desse tipo de situação é possível aplicar um Trigger que cria essas relações automaticamente.

Replicação síncrona de tabelas: A replicação síncrona se refere a gravação de dados em dois ou mais locais ao mesmo tempo, caso você queira preencher dois campos em duas tabelas diferentes ao mesmo tempo ou até mesmo para a criação de tabelas.

Registro de eventos e armazenamento de informações: Registrar e armazenar e referem a inserir.

Imposição de autorizações de segurança: Aplicação de uma senha para alterar, atualizar ou deletar um dado de uma tabela, basicamente um gerenciamento de permissão.

Impedir transações inválidas: Podemos criar um Trigger que tem requisitos que devem ser preenchidos para não acionar, quando esses requisitos não são preenchidos esse Trigger aciona um “evento de impedimento” (Tipo palavras mágicas que a gente aprende quando criança, o trigger é uma pessoa fresca que está atrás da porta e você (uma transação) quer passar mas para isso precisa pedir “por favor”).