Bootcamp IGTI

Desafio

Módulo 2	A Linguagem SQL

Objetivos

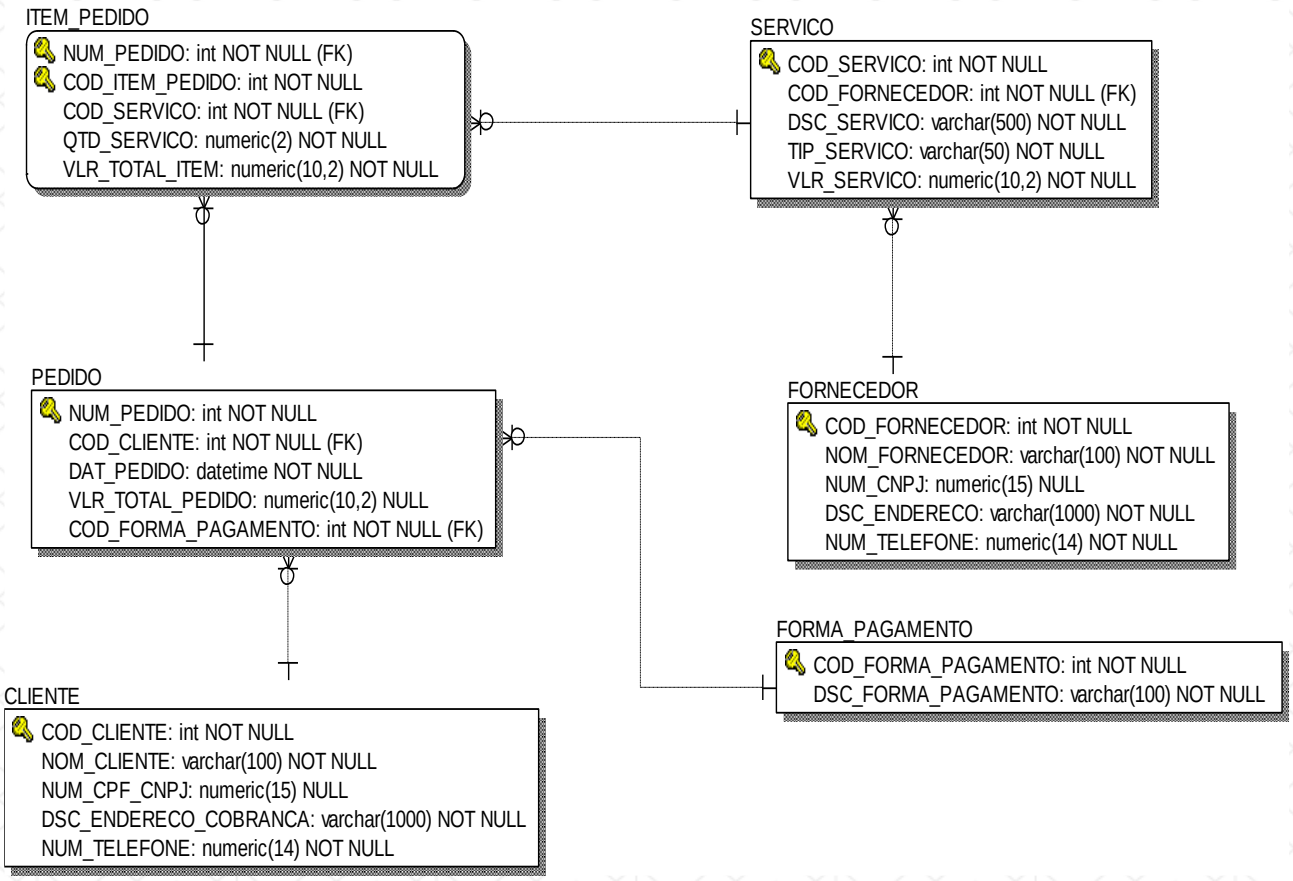
Exercitar os seguintes conceitos trabalhados no Módulo:

- ✓ Modelo de dados relacional;
- ✓ Banco de dados relacional;
- ✓ Instruções da classe DDL para criação de estruturas de dados;
- ✓ Instruções da classe DML para consultar, inserir, excluir e atualizar dados;
- ✓ Instruções da classe TCL para controlar transações;
- ✓ Instruções da classe DCL para conceder e revogar privilégios.

Enunciado

O IGTI inaugurará em breve seu site de e-commerce para serviços educacionais, e necessita de uma aplicação que permita o cadastramento de seus clientes e o controle dos serviços adquiridos por eles. O IGTI trabalha com vários fornecedores (parceiros), que oferecem diversos tipos de serviços, como aulas particulares à distância, monitoria/ tutoria/ produção de conteúdo para cursos rápidos, bootcamps, cursos de graduação, MBAs etc., bem como consultoria para criação de conteúdo/ produtos de ensino à distância. Um mesmo fornecedor pode oferecer mais de um tipo de serviço. Quando um cliente compra um serviço, é emitida uma nota de pedido relacionando todos os serviços (produtos) que envolveram a transação e a forma de pagamento selecionada pelo cliente.

Para este sistema, foi desenvolvido, inicialmente, o modelo de dados físico relacional a seguir:



Atividades

A empresa lhe contratou como o analista de banco de dados responsável por implementar e gerenciar esse modelo de dados, construir as queries demandadas pelos desenvolvedores e sanar as dúvidas da equipe do projeto referentes à Linguagem SQL.

Os alunos deverão desempenhar as seguintes atividades:

1. Gerar o script DDL para criar, no SQL Server, as respectivas estruturas de dados no banco **BDecommerce**;
2. Gerar o script DML para inserir dados fictícios em todas as tabelas do schema físico criado no item anterior;
3. Desenvolver uma query SQL para gerar uma **lista única** com os contatos (nome, endereço e telefone) de **clientes** e **fornecedores**, **sem informações repetidas**, **ordenada alfabeticamente**. Esse relatório deve conter as colunas com o label Nome da Pessoa, Endereço e Telefone.

4. A tabela **FORMA_PAGAMENTO** foi criada com o código DDL abaixo. Depois que a tabela já estava populada, a empresa solicitou a inclusão de mais uma coluna (IND_STATUS char(1) NOT NULL), para indicar se a forma de pagamento está disponível (‘D’) ou indisponível (‘I’). Desenvolva o script para as **duas formas existentes** para se incluir essa nova coluna **sem ter que recriar a tabela,** e considerando que todas as formas de pagamento atuais da tabela estão disponíveis.

```
CREATE TABLE FORMA_PAGAMENTO
(
        [COD_FORMA_PAGAMENTO] [int] NOT NULL,
        [DSC_FORMA_PAGAMENTO] [varchar](100) NOT NULL,
                        CONSTRAINT [PK_FORMAPAG] PRIMARY KEY CLUSTERED
                        (
                                [COD_FORMA_PAGAMENTO] ASC
                        )
)
```

5. A empresa solicitou que a coluna **VLR_TOTAL_PEDIDO** da tabela **PEDIDO,** que já se encontrada populada, seja alterada para **NOT NULL**. Construa uma **query DML com um SELECT** para você executar previamente e **verificar se o comando abaixo será concluído com sucesso ou não.**

```
ALTER TABLE PEDIDO
ALTER COLUMN VLR_TOTAL_PEDIDO numeric (10,2) NOT NULL;
```

6. Na consulta abaixo, criada por um desenvolvedor, ela deveria retornar **5 colunas**, mas estão sendo retornadas **7 colunas**. Foi solicitado que você corrija a query para apresentar as 5 colunas com os labels: COD_CLIENTE, NOME COMPLETO, CPF_CNPJ, DSC_ENDERECO_COBRANCA e AS CELULAR.

```
SELECT      COD_CLIENTE, NOM_CLIENTE, 'NOME COMPLETO',
            NUM_CPF_CNPJ CPF_CNPJ, DSC_ENDERECO_COBRANCA,
            NUM_TELEFONE, ‘CELULAR’
FROM CLIENTE
ORDER BY NOM_CLIENTE
GO
```

7. A empresa que te contratou solicitou que você **explique o motivo** pelo qual a query abaixo (para retornar os pedidos dos clientes) está dando **erro de execução** e a **reescreva-a de forma que ela possa ser executada com sucesso**.

```
SELECT C.NOM_CLIENTE AS "NOME DO CLIENTE", P.NUM_PEDIDO PEDIDO,
            P.DAT_PEDIDO AS "DATA DO PEDIDO", P.VLR_TOTAL_PEDIDO
FROM CLIENTE C JOIN PEDIDO P
ON C.COD_CLIENTE = P.COD_CLIENTE
WHERE YEAR ("DATA DO PEDIDO") = '2020'
ORDER BY "NOME DO CLIENTE";
```

8. A empresa solicitou que você providencie uma query para listar todos os serviços e os respectivos fornecedores, ordenando essa lista **alfabeticamente pelo nome do fornecedor** e **descendentemente pelo valor do serviço**. A lista deve possuir as colunas **NOM_FORNECEDOR, NUM_TELEFONE, DSC_SERVICO, TIP_SERVICO e VLR_SERVICO**. Foi solicitado também que a query use o padrão **ANSI-92** para relacionar as tabelas.

9. A empresa irá fazer uma enxugada no portfólio atual, de forma que consiga oferecer novos serviços que estão sendo criados pelos fornecedores. Para isso, ela precisa que você elabore uma query que liste os **serviços que nunca foram vendidos, ou que tiveram um total de vendas menor que 5 em 2019**. O relatório precisa conter as colunas DSC_SERVICO, TIP_SERVICO, VLR_SERVICO e NOM_FORNECEDOR. Foi solicitado que a query, neste caso, use padrão **ANSI-89** para relacionar as tabelas.

10. Devido à algum problema na aplicação, alguns pedidos estão sem o valor total do pedido (coluna VLR_TOTAL_PEDIDO da tabela PEDIDO está nula), ou com valor calculado erroneamente (deveria ser a soma de todos os valores, do mesmo pedido, da coluna VLR_TOTAL_ITEM da tabela ITEM_PEDIDO). A empresa solicita que você desenvolva uma **query para listar os pedidos que estão com esse problema**. A query deve retornar as **colunas NUM_PEDIDO, DAT_PEDIDO e VLR_TOTAL_PEDIDO**. A empresa também solicita que você escreva a **query para atualizar a coluna VLR_TOTAL_PEDIDO** de forma correta.

11. Foi identificado um problema no modelo de dados original, no qual a coluna NUM_CPF_CNPJ, da tabela CLIENTE, foi criada de forma a aceitar nulo**.** Com o intuito de verificar o tamanho do problema, a empresa solicitou à um de seus desenvolvedores que elaborasse uma query para listar o percentual de clientes cadastrados que estão sem essa informação. O desenvolvedor criou a query abaixo:

```
SELECT CAST
(
        (       SELECT CAST(COUNT (NUM_CPF_CNPJ) AS numeric (3,2))
                FROM CLIENTE WHERE NUM_CPF_CNPJ IS NULL
        )
        /
        (       SELECT CAST(COUNT (NUM_CPF_CNPJ) AS numeric (3,2))
                FROM CLIENTE
        ) * 100 AS int
) AS "Percentual de Clientes sem Informação de Documento"
```

A empresa constatou que não há nenhum erro de sintaxe na query, mas solicitou sua consultoria, para **informar se a query calculará o percentual de clientes cadastrados sem CPF / CNPJ de forma correta**. Em caso negativo, ela solicita que você forneça a query com o ajuste necessário.

12. Foi feita uma regulamentação nova para sistemas de venda de serviços online, onde todas as informações exibidas para os clientes, no site, devem ser exibidas em letras MAIÚSCULAS. A alteração na camada da aplicação para os novos dados serem inseridos já foi feita, mas é preciso tratar o backlog das informações que já se encontravam persistidas no banco de dados. De acordo com o modelo de dados, existem **7 colunas do tipo string que precisam ter os dados alterados para maiúsculo**. O desenvolvedor informou que seriam necessários 7 comandos DML para fazer essa alteração, mas você disse que consegue otimizar para fazer com 4 comandos. **Quais seriam esses 4 comandos?**

13. A empresa precisará de um relatório que liste a quantidade de itens vendidos mensalmente e o valor total desses itens, por fornecedor, no ano corrente. Todos os fornecedores devem estar listados, mesmo se ainda não tiver nenhum produto vendido. **Selecione as instruções abaixo que serão necessárias para essa query e coloque-as na devida ordem.**

- GROUP BY F.NOM_FORNECEDOR, P.DAT_PEDIDO
- SELECT F.NOM_FORNECEDOR, MONTH(P.DAT_PEDIDO) AS MÊS,
- SUM(I.QTD_SERVICO) AS "TOTAL DE ITENS",
- LEFT JOIN SERVICO S
- GROUP BY F.NOM_FORNECEDOR
- FROM FORNECEDOR F
- COUNT(I.QTD_SERVICO) AS "TOTAL DE ITENS",
- ON I.NUM_PEDIDO = P.NUM_PEDIDO
- INNER JOIN SERVICO S
- ORDER BY F.NOM_FORNECEDOR
- GROUP BY F.NOM_FORNECEDOR, MONTH (P.DAT_PEDIDO)
- JOIN ITEM_PEDIDO I
- ON S.COD_SERVICO = I.COD_SERVICO

- SUM(I.VLR_TOTAL_ITEM) AS "VALOR TOTAL"
- ON F.COD_FORNECEDOR = S.COD_FORNECEDOR
- JOIN PEDIDO P
- RIGHT JOIN PEDIDO P
- WHERE YEAR(P.DAT_PEDIDO) = YEAR(GETDATE())
- HAVING YEAR(P.DAT_PEDIDO) = YEAR(GETDATE())

14. Para retornar a lista de clientes que não fizeram pedidos, foi construída a query abaixo.

```
SELECT NOM_CLIENTE, NUM_TELEFONE
FROM CLIENTE
WHERE COD_CLIENTE NOT IN (SELECT COD_CLIENTE FROM PEDIDO)
ORDER BY NOM_CLIENTE;
```

A empresa solicita sua consultoria para **informar se a query em questão atende ao requisito** (listar clientes que não fizeram pedidos) e se há uma **forma mais otimizada de obter o mesmo resultado, e qual seria essa query**.

15. Foi encontrada a query abaixo no código da aplicação e a empresa está solicitando sua ajuda para **identificar o que essa query faz**. Em adição**, informe uma query mais otimizada para atingir o mesmo resultado, sem utilizar uma subconsulta correlacionada ou multivalorada**.

```
SELECT C.NOM_CLIENTE, C.NUM_TELEFONE
FROM CLIENTE C
WHERE EXISTS (
                        SELECT P.COD_CLIENTE
                        FROM PEDIDO P
                        WHERE P.COD_CLIENTE = C.COD_CLIENTE
                )
ORDER BY C.NOM_CLIENTE;
```

16. A empresa solicitou que você **elabore uma query para gerar uma cópia dos dados da tabela ITEM_PEDIDO**. Essa tabela com a cópia dos dados deve se chamar **TMP_ITEM_PEDIDO.**

17. A empresa solicitou que você **elabore uma query para excluir os clientes que estão cadastrados no banco de dados, mas que não fizeram nenhum pedido**.

18. Foi solicitado que você **gere uma query para inserir os dados abaixo na tabela FORMA_PAGAMENTO, que funcione independentemente da ordem que as colunas foram criadas.**

COD_FORMA_PAGAMENTO	DSC_FORMA_PAGAMENTO
1	CARTÃO DE DÉBITO
2	CARTÃO DE CRÉDITO
3	BOLETO
4	FATURA
5	TRANSFERÊNCIA BANCÁRIA

19. Dado o código abaixo:

```
BEGIN TRANSACTION Transacao1
INSERT INTO FORMA_PAGAMENTO
(COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
VALUES (1, 'CARTÃO DE DÉBITO');
BEGIN TRANSACTION Transacao2
INSERT INTO FORMA_PAGAMENTO
(COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
VALUES (2, 'CARTÃO DE CRÉDITO');
SELECT * FROM FORMA_PAGAMENTO ORDER BY 1;
COMMIT TRANSACTION Transacao2
INSERT INTO FORMA_PAGAMENTO
(COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
VALUES (3, 'BOLETO');
SELECT @@TRANCOUNT AS 'PRIMEIRA CONTAGEM';
SELECT *
FROM FORMA_PAGAMENTO
ORDER BY DSC_FORMA_PAGAMENTO;
```

COMMIT TRANSACTION Transacao1
INSERT INTO FORMA_PAGAMENTO
(COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
VALUES (4, 'FATURA');
SELECT @@TRANCOUNT AS 'SEGUNDA CONTAGEM';
*SELECT * FROM FORMA_PAGAMENTO;*

Informe quais os dados serão retornados para cada um dos comandos SELECT executados dentro da query acima:

- SELECT * FROM FORMA_PAGAMENTO ORDER BY 1 ➔
- SELECT @@TRANCOUNT AS 'PRIMEIRA CONTAGEM' ➔
- SELECT * FROM FORMA_PAGAMENTO
  - ORDER BY DSC_FORMA_PAGAMENTO ➔
- SELECT @@TRANCOUNT AS 'SEGUNDA CONTAGEM' ➔
- SELECT * FROM FORMA_PAGAMENTO ➔

20. Após a execução do bloco SQL abaixo, **quantas linhas de fato serão inseridas na tabela FORMA_PAGAMENTO**?

```
BEGIN TRANSACTION Transacao1

    INSERT INTO FORMA_PAGAMENTO
    (COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
    VALUES (199, 'CARTÃO DE DÉBITO');

    BEGIN TRANSACTION Transacao2

    INSERT INTO FORMA_PAGAMENTO
    (COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
    VALUES (299, 'CARTÃO DE CRÉDITO');

    COMMIT TRANSACTION Transacao2
```

```
INSERT INTO FORMA_PAGAMENTO
(COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
VALUES (399, 'BOLETO');

ROLLBACK
COMMIT TRANSACTION Transacao1

INSERT INTO FORMA_PAGAMENTO
(COD_FORMA_PAGAMENTO, DSC_FORMA_PAGAMENTO)
VALUES (499, 'FATURA');
```

21. Foi solicitada a você a criação de um objeto de código compilado (uma procedure), para fazer a inserção de dados na tabela CLIENTE **de forma que os dados sejam passados como parâmetros**. Forneça o código DDL para criação dessa procedure, de nome SP_INSERE_CLIENTE, e o exemplo de execução dela**.**

22. A empresa solicitou que você **construa o script DCL,** de forma que o usuário da aplicação, de nome ***UserApp*** tenha permissão para selecionar, inserir, atualizar e excluir dados de todas as tabelas, **mas não tenha permissão de inserção direta na tabela CLIENTE** (para inserir dados de cliente, o usuário deverá usar uma procedure, de nome SP_INSERE_CLIENTE). Forneça os comandos que devem estar nesse script DCL.

Respostas Finais

Os alunos deverão desenvolver a prática e, depois, responder às seguintes questões objetivas: