|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Banco de Dados**  **Segurança de Dados**  **AC4 – Alta Disponibilidade – Replicação** | | |
| *Nome do Aluno:* | | | | *RA:* |
| *Nome do Aluno:* | | | | *RA:* |
| *Turma:* | *Série:* | | *Data:* \_\_\_/\_\_\_/\_\_\_\_\_\_ | |
| *Professor(a) Responsável:* Gustavo Bianchi Maia | | | | |
| Nota: ( ) Visto do Prof. | | | | |

**Regras:**

* Entrega individual para implementação na mesma instância do SQL Server.
* Entrega em duplas para quem implementar utilizando dois computadores diferentes.

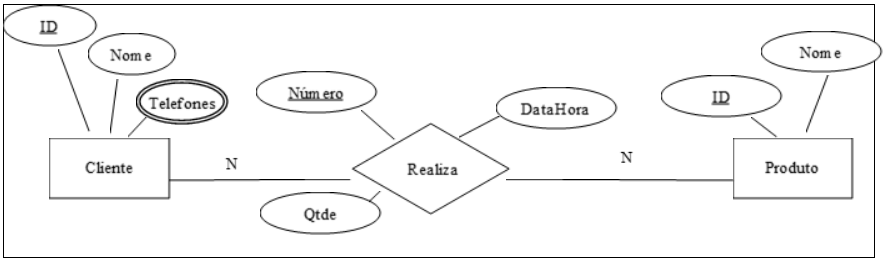
**Implementar Replicação, utilizando as bases conforme orientação abaixo:**

* Crie 2 databases ( preferencialmente em 2 computadores/instâncias diferentes ).
* Crie o seguinte banco de exemplo nos dois ambientes ( faça as adaptações necessárias )

**Dados a serem inseridos no principal:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome** | **Endereço** | **Telefones** | **Pedidos** | **DataHora** |
| José | Av. Três, numero 1 | 9378-3259 8745-3452 | 1. 3x Calabresa 2. 2x Frango | 20/08/2017 23:11 27/08/2017 22:30 |
| Maria | Av. Impar, nr 33, apto 12 | *-* | 1. 7x Calabresa + 4x Brócolis | 21/09/2017 19:59 |

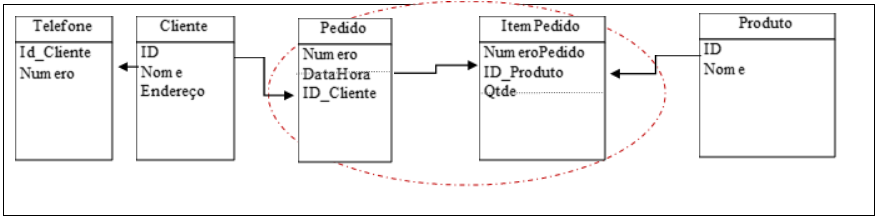
**DER:**



Uma Cliente pode comprar vários produtos, cada compra é chamada de Pedido.

Como um cliente pode ter múltiplos telefones, eles focam incluídos como um atributo multivalorado.

**Modelo lógico relacional:**



**Nota:** Foi adicionado um passo de normalização ( 3NF ), separando Pedido de ItemPedido, pois: DataHora do pedido e Id\_cliente tem DF ( dependência funcional ) apenas do Número do Pedido ( ou seja, são os mesmos valores independente do produto, repetindo-se ), enquanto que Qtde tem DF sobre Numero do pedido + Id\_produto.

**Critérios de sucesso / Pontuação:**

2 pts – Você configurou os dois ambientes corretamente ( criação do banco e tabelas ).

+2pts – Você populou a base principal com dados ( carga inicial do servidor principal ).

+4pts – Você implementou replicação ( transactional ) de todos os objetos ( *Publisher e Subscriber* ), crie 1 artigo com todas as tabelas.

+2pts – Você criou uma visão cujo resultado é similar ao seguinte resultset: Nome do cliente, Nome do produto comprado, qtde total comprada. Filtre apenas pedidos no mês atual ( use a função getdate () ), este será seu *Article*. Utilizando Snapshot Replication, você finalmente replicou o conteúdo desta visão para uma tabela de destino ( *Subscriber* ).

**Demonstre a implementação para o professor.**

**Esta entrega não pode ser feita por email, ou seja, é uma apresentação presencial.**

**Se prepare para responder à 1 ou 2 questões ( chamada oral ) sobre seu processo.**

**Sejam os seguintes exemplos de comandos para ajuda-lo neste trabalho:**

**CREATE TABLE cliente**

**(**

**id *INT* NOT NULL IDENTITY(1, 1),**

**nome *VARCHAR*(50) NOT NULL,**

**endereco *VARCHAR*(255) NULL,**

**CONSTRAINT pk\_cliente PRIMARY KEY ( id )**

**)**

**go**

**CREATE TABLE telefone**

**(**

**id\_cliente *INT* NOT NULL,**

**numero *INT* NOT NULL,**

**CONSTRAINT pk\_telefone PRIMARY KEY ( id\_cliente, numero ),**

**CONSTRAINT fk\_telefonecliente FOREIGN KEY ( id\_cliente ) REFERENCES cliente**

**( id )**

**)**

**go**

**CREATE TABLE produto**

**(**

**id *INT* NOT NULL IDENTITY(1, 1),**

**nome *VARCHAR*(50) NOT NULL,**

**CONSTRAINT pk\_produto PRIMARY KEY ( id )**

**)**

**go**

**CREATE TABLE pedido**

**(**

**numero *INT* NOT NULL IDENTITY(1, 1),**

**datahora *DATETIME* NOT NULL,**

**id\_cliente *INT* NOT NULL,**

**CONSTRAINT pk\_pedido PRIMARY KEY (numero ),**

**CONSTRAINT fk\_pedidocliente FOREIGN KEY ( id\_cliente ) REFERENCES cliente (**

**id )**

**)**

**go**

**CREATE TABLE itempedido**

**(**

**numeropedido *INT* NOT NULL,**

**id\_produto *INT* NOT NULL,**

**qtde *INT* NOT NULL,**

**CONSTRAINT pk\_pedido PRIMARY KEY (numeropedido, id\_produto),**

**CONSTRAINT fk\_itempedidopedido FOREIGN KEY ( numeropedido) REFERENCES**

**pedido ( numero ),**

**CONSTRAINT fk\_itempedidoproduto FOREIGN KEY ( id\_produto) REFERENCES**

**produto ( id )**

**)**

**Conferência / Demonstração para o Professor:**

Apresente duas consultas, que façam selects na mesma\* tabela em ambos os bancos / servidores ( utilizar um linked server )

EX: SELECT \* FROM [Principal].impacta.dbo.cliente

SELECT \* FROM [Secundario].impacta.dbo.cliente

Insira uma linha nova no banco principal

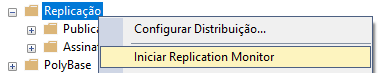
INSERT INTO [Principal].impacta.dbo.cliente(...)VALUES(...)

Demonstre que os dois servidores ficaram atualizados e idênticos novamente

SELECT \* FROM [Principal].impacta.dbo.cliente

SELECT \* FROM [Secundario].impacta.dbo.cliente

Depois, abra o Replication Monitor ( botão direito sobre replicação → iniciar Replication Monitor )



E demonstre que sua aplicação está rodando sem erros.

**Observações:**

* Avaliação Individual.
* Organização, clareza e legibilidade na resolução da prova serão consideradas na correção.
* A interpretação da atividade faz parte do processo de avaliação
* Qualquer tentativa de fraude, cola será punido com nota zero.
* O total desta avaliação corresponde à 1 de 10 avaliações continuadas.

**Boa Avaliação!**