Curso SQL07
Laboratório – Módulo 02

Questão 01 – Simulando uma aplicação cliente

01.a . Utilizando o banco de dados Nortwhind, execute o script abaixo (entre os itens 1.1 e 1.2):

```
USE tempdb
go

IF OBJECT_ID('Vendas') IS NOT NULL
     DROP TABLE Vendas
go

IF OBJECT_ID('DetalhesVenda') IS NOT NULL
     DROP TABLE DetalhesVenda
go

SELECT *
INTO dbo.Vendas
FROM Northwind.dbo.Orders
go

SELECT *
INTO dbo.DetalhesVenda
FROM Northwind.dbo.[Order Details]
go

ALTER TABLE dbo.DetalhesVenda
ADD Codigo INT IDENTITY(1,1) PRIMARY KEY
GO

SELECT * FROM dbo.Vendas
SELECT * FROM dbo.DetalhesVenda
go
```

- Imagine uma aplicação que exibe em um grid as últimas 100 vendas que foram feitas, mostrando os campos OrderId, CustomerID e OrderDate (consulta 01).
- Após escolher um pedido específico a aplicação mostra mais dados do pedido e os detalhes do pedido (Consultas 02 e 03), exibindo na tela um formulário com os campos: OrderID, CustomerID, EmployeeId, OrderDate, ShipName, e os itens: ProductId e UnitPrice * Quantity.

Como você faria para otimizar o código de chamada e as consultas.

Questão 02 – Otimizando uma consulta

02.a Utilizando o banco de dados Nortwind, como você faria para otimizar a consulta abaixo:

```
SELECT ProductName, p.UnitPrice, CompanyName, Country, quantity
FROM Products as P inner join Suppliers as S
ON P.SupplierID = S.SupplierID
inner join [order details] as od
on p.productID = od.productid
WHERE CategoryID in (1,2,3) and p.Unitprice < 20
and Country = 'uk' and Quantity < 30
GO
```

Questão 03 – Leitura excessiva de páginas

03.a Imagine um banco que armazena em uma tabela o valor atual dos saldos de todos os seus clientes. Juntamente com o saldo está a data na qual aquele valor era válido, então existirá mais de uma entrada por cliente na tabela.

O script abaixo (entre 3.1 e 3.2) cria a tabela e insere o saldo atual de 10.000 clientes.

```
-- 3.1) Início do setup
USE tempdb
go

--   Cria a tabela SaldoConta com chave primária e um
índice não clusterizado
if exists(select [name] from sysobjects where xtype = 'U'
and [name] = 'SaldoConta')
     DROP TABLE SaldoConta
go
CREATE TABLE SaldoConta (
CodigoCliente INT NOT NULL,
DataSaldo DATETIME NOT NULL,
SaldoAtual MONEY NOT NULL,
NomeBanco CHAR(100) NOT NULL,
OutraInformacao CHAR(100) NULL
)
go

ALTER TABLE SaldoConta
ADD CONSTRAINT PK_SaldoConta
PRIMARY KEY (CodigoCliente, DataSaldo)
go

CREATE NONCLUSTERED INDEX idx_DataSaldo
ON SaldoConta (DataSaldo)
go
```


```
--    Insere 10.000 clientes com diferentes saldos
diferentes.
DECLARE @Cont INT
SET @Cont = 0

WHILE @Cont < 10000
BEGIN

     INSERT INTO SaldoConta VALUES (@Cont, '20070131',
((RAND() * 1000) * DATEPART(ss, GETDATE())), 'Qualquer um',
'SQL Server 2008')
     SET @Cont = @Cont + 1
END
go

-- 3.1) Fim do setup
```

Faça uma análise do custo de IO da consulta "SELECT * FROM dbo.SaldoConta".

Supondo que durante a noite o dinheiro em conta gere dividendo e, para simular esse cenário, o script abaixo é executado, inserindo mais 10.000 registros na tabela.

```
-- Durante a noite, o dinheio em conta rende um valor
variável...
DECLARE @SaldoAtual MONEY
DECLARE @Cont INT
SET @Cont = 0

WHILE @Cont < 10000
BEGIN

     SELECT @SaldoAtual = SaldoAtual FROM SaldoConta WHERE
CodigoCliente = @Cont
     -- Esse meu banco é massa, o dinheiro somente aumenta
nas contas...
     INSERT INTO SaldoConta VALUES (@Cont, '20070201',
@SaldoAtual + ((RAND() * 10) * DATEPART(ss, GETDATE())),
'BANCO DO LUTI', 'Keep walking')
     SET @Cont = @Cont + 1
END
go
```

Faça novamente uma análise do custo de IO da consulta "SELECT * FROM dbo.SaldoConta", mas antes responda a pergunta: **o que você espera ver como resultado?** Leve em conta os novos registros que foram inseridos.

O número de registros foi maior do esperado, então:

- Explique o problema
- Comprove o problema

- Forneça uma solução

Questão 04 – Análise estrutural

04.a Sem conhecer em nada a estrutura das aplicações que acessam um banco de dados e o perfil de consulta, faça uma análise crítica da tabela abaixo (e seus índices).

```
USE tempdb
go

IF OBJECT_ID('RegistroPessoal') IS NOT NULL
     DROP TABLE RegistroPessoal
GO

CREATE TABLE RegistroPessoal
(Identificador UNIQUEIDENTIFIER NOT NULL PRIMARY KEY
DEFAULT(NEWID()),
 Nome VARCHAR(200) NOT NULL,
 Idade SMALLINT NOT NULL,
 CPF CHAR(11) NOT NULL,
 RG VARCHAR(50) NULL,
 DataEmissaoRG DATETIME NULL,
 Sexo CHAR(1) NOT NULL,
 DataNascimento DATETIME NULL
 )

CREATE NONCLUSTERED INDEX idx_NCL_RegistroPessoal_Sexo
ON RegistroPessoal (Sexo)
go

CREATE NONCLUSTERED INDEX
idx_NCL_RegistroPessoal_CoverIndex1
ON RegistroPessoal (Idade, CPF)
go

CREATE NONCLUSTERED INDEX
idx_NCL_RegistroPessoal_CoverIndex2
ON RegistroPessoal (DataNascimento, CPF)
INCLUDE (Idade, RG, DataEmissaoRG, Nome, Sexo)
go

CREATE NONCLUSTERED INDEX idx_NCL_RegistroPessoal_Nome
ON RegistroPessoal (Nome)
go
```