**AVALIAÇÃO TÉCNICA**

**VAGA: Desenvolvedor .net**

**Data: 03/07/2019**

**.NET**

1. Em um determinado banco, você foi designado para desenhar o novo processamento de liberação de crédito.

**Existem 5 tipos de créditos, cujos :**

* Credito Direto - Taxa de 2% ao mês
* Credito Consignado - Taxa de 1% ao mês
* Credito Pessoa Jurídica - Taxa de 5% ao mês
* Credito Pessoa Física - Taxa de 3% ao mês
* Credito Imobiliário - Taxa de 9% ao ano

Cada crédito tem uma forma de validação diferente, porém, todos precisam passar neste método de validação para liberação do crédito.

Implemente as classes que você julgue necessárias, para implementação do processamento, e demonstre exemplos em uma aplicação simples de console.

**Defina como entradas as seguintes variáveis:**

* Valor do crédito
* Tipo de crédito
* Quantidade de parcelas
* Data do primeiro vencimento

**As validações das entradas são as seguintes:**

* O valor máximo a ser liberado para qualquer tipo de empréstimo é de R$ 1.000.000,00
* A quantidade de parcelas máximas é de 72x e a mínima é de 5x
* Para o crédito de pessoa jurídica, o valor mínimo a ser liberado é de R$ 15.000,00
* A data do primeiro vencimento sempre será no mínimo D+15 (Dia atual + 15 dias), e no máximo, D+40 (Dia atual + 40 dias)

**Os resultados precisam conter as seguintes informações:**

* Status do crédito (Aprovado ou recusado, de acordo com as premissas acima)
* Valor total com juros
* Valor do juros

**Para este exercício, os juros são calculados da seguinte forma, incremente a porcentagem de juros no valor total do crédito.**

**Resposta: A solução segue em anexo junto ao e-mail, irei usar essa mesma solução para responder a questão de Arquitetura.**

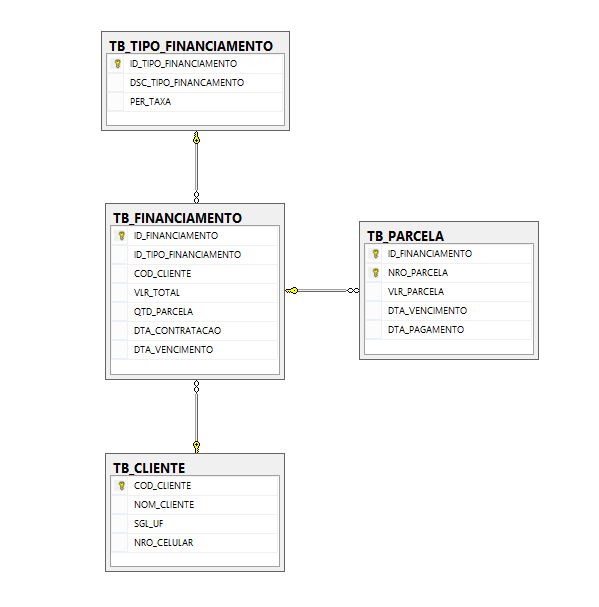
**ORACLE**

2. Modele uma estrutura de dados do seguinte caso:

* Um cliente tem os seguintes campos : Nome, Id Cliente, UF, Celular
* Um cliente tem N financiamentos.
* Um financiamento tem os seguintes campos : Id Cliente, Tipo Financiamento, Valor Total, Data Vencimento
* Cada financiamento tem N parcelas, cujas tem os seguintes campos : Id Financiamento, Número da Parcela, Valor Parcela, Data Vencimento, Data Pagamento;

**Crie as tabelas que julgue necessárias e insira alguns registros de testes na mesma.**

**Diagrama do Modelo de Dados:**

****

Elabore as seguintes querys:

* Listar todos os clientes do estado de SP que tenham mais de 60% das parcelas pagas.

SELECT

A.\*

FROM TB\_CLIENTE A

INNER JOIN TB\_FINANCIAMENTO B ON (A.COD\_CLIENTE = B.COD\_CLIENTE)

OUTER APPLY

(SELECT

COUNT(NRO\_PARCELA) AS TOT\_PARCELA

FROM TB\_PARCELA

WHERE

ID\_FINANCIAMENTO = B.ID\_FINANCIAMENTO AND

DTA\_PAGAMENTO IS NOT NULL) AS C

WHERE

A.SGL\_UF = 'SP' AND

((C.TOT\_PARCELA \* 100) / B.QTD\_PARCELA) > 60

* Listar os primeiros 4 clientes que tenham alguma parcela com mais de 05 dias atrasadas (Data Vencimento maior que data atual E data pagamento nula)

SELECT

TOP 4

C.COD\_CLIENTE,

C.NOM\_CLIENTE

FROM TB\_PARCELA A

INNER JOIN TB\_FINANCIAMENTO B ON (A.ID\_FINANCIAMENTO = B.ID\_FINANCIAMENTO)

INNER JOIN TB\_CLIENTE C ON (B.COD\_CLIENTE = C.COD\_CLIENTE)

WHERE

A.DTA\_PAGAMENTO IS NULL AND

DATEDIFF(day, A.DTA\_VENCIMENTO, GETDATE()) > 5

GROUP BY

C.COD\_CLIENTE,

C.NOM\_CLIENTE

* Listar todos os clientes que já atrasaram em algum momento duas ou mais parcelas em mais de 10 dias, e que o valor do financiamento seja maior que R$ 10.000,00.

SELECT

TOP 4

C.COD\_CLIENTE,

C.NOM\_CLIENTE

FROM TB\_PARCELA A

INNER JOIN TB\_FINANCIAMENTO B ON (A.ID\_FINANCIAMENTO = B.ID\_FINANCIAMENTO)

INNER JOIN TB\_CLIENTE C ON (B.COD\_CLIENTE = C.COD\_CLIENTE)

WHERE

A.DTA\_PAGAMENTO IS NOT NULL AND

DATEDIFF(day, A.DTA\_VENCIMENTO, A.DTA\_PAGAMENTO) > 10 AND

B.VLR\_TOTAL > 10000

GROUP BY

C.COD\_CLIENTE,

C.NOM\_CLIENTE

**ARQUITETURA**

**Questão conceitual, escreva em detalhes ou diagramas como você montaria uma arquitetura para o cenário abaixo, informando as tecnologias, arquitetura, e o que mais achar necessário.**

Em tempos de expansão digital, sua empresa foi contratada para desenhar uma arquitetura moderna que sustente o crescimento digital e de vários novos canais, e também tenha formas de manter o legado funcionando.

Descreva ou desenhe o que e como você utilizaria para suportar este crescimento, tendo em vista que é necessária uma arquitetura que agregue os meios de comunicação com Mainframe, e que todos os sistemas web possam se comunicar entre eles sem a reescrita de códigos.

Todos os sistemas são extremamente críticos e de alta performance, também contando com um volume consideravelmente alto de dados sendo transacionados a todo tempo.

Comente também como você desenharia a solução para implantação deste cenário, visando que quanto menor a dependência com áreas de operação para executar a implantação na mão, mais produtivo e assertivo será a empresa.

**Resposta:**

**Em cima do cenário descrito, desenvolvi o teste acima, pensando na arquitetura deste requisito de negocio.**

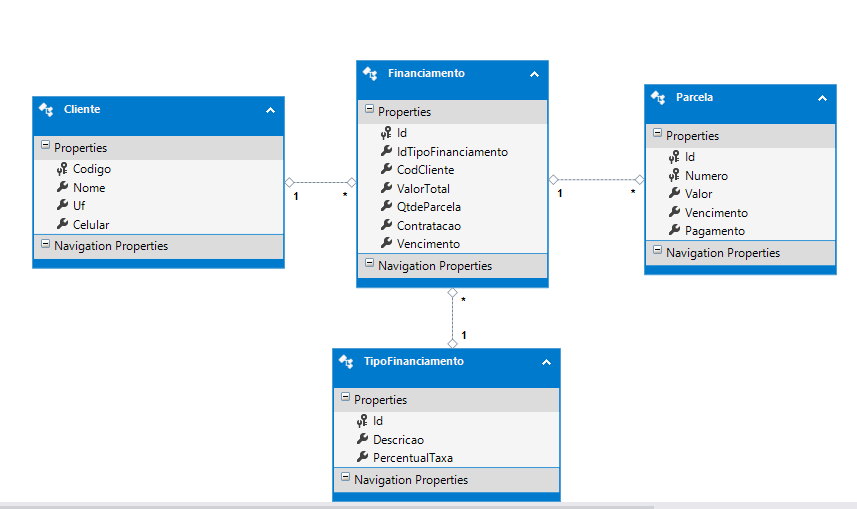
**Utilizei arquitetura DDD, focando diretamente no entendimento do domínio e prezando pela melhor abstração possível de cada nível de desenvolvimento.**

**O padrão permite criar n formas de UI (Windows Forms, App, Web, Console), desacoplada das regras de negócio, pois só consomem os serviços.**

**Os designer partness utilizados foram: ORM, SOAP, POCO, Linq to Entity e Interfaces.**

**Segue o porque de adotar cada padrão:**

**ORM: A ferramenta adotada foi Entity Framework, os ganhos são na agilidade de eventuais trocas de repositórios de Banco de Dados (nesta aplicação estou usando SQL Server, mas podemos alterar o repositorio para Oracle alterando a Connection String, sem impacto para o restante da solução). Outro motivo é por poder evitar conflitos entre as áreas de DBAs e Desenvolvimento, por discussões de conceitos OO, nomenclatura, etc.**

****

**Este é Data Context criado para contemplar o modelo de dados demonstrado no item anterior, as nomenclaturas estão diferentes, porém reflete o modelo sem conflito e sem necessidade de desgaste entre equipes. Para ganho de performance, utilizei injeção de dependências na construção do ContextModel (Mapping).**

**SOAP: foi adotado o WCF, para contemplar o SOAP. Todas as regras estão contempladas e podem ser expostas através de serviço e podendo evoluir facilmente para tecnologias mais novas e também para MicroServiços a medida que o negócio ganha maturidade. Possibilita a integração entre áreas distintas de forma menos traumática.**

**POCO: Foi adotada para não correr o risco de se quebrar o Dominio, entendemos na arquitetura adotada como o Core. Ferramentas ORMs tem a opção de criarem automaticamente os modelos, porém trazem uma complexidade e um risco muito alto na evolução da produto. Por isso a escolha de tomar conta do Dominio utilizando esse DP.**

**Linq to Entity: Padrão adotado, para escritas de instruções de bancos de dados, select, insert, etc. Evitando a grande dependência de sintaxes em instruções transact-sql.**

**Interfaces: Expor os serviços de forma abstrata, sem risco a acesso a regras de negócios, etc.**

**AO CONCLUIR**

Crie um repositório (**Não utilize no projeto ou em qualquer lugar o nome de nenhuma empresa**)

e nos envie o link, ou envie os arquivos compactados para o e-mail: [monica.conz@to-brasil.com](mailto:monica.conz@to-brasil.com) e diego.postacchini@safra.com.br