



SHOPPING BUDDY

**we make  
better  
shop carts**

FIAP + PLUSOFT

---

SÃO PAULO, 2023

# DATABASE APPLICATION & DATA SCIENCE



## CHALLENGE

### SOBRE O PROJETO

Neste projeto, será desenvolvido um sistema de recomendação personalizado para fornecer sugestões precisas aos clientes com base em seus históricos de compras e interações anteriores com a marca. O sistema será baseado em técnicas de Machine Learning, permitindo que o ChatGPT aprenda com as interações anteriores e melhore continuamente suas sugestões. O objetivo do projeto é aumentar a satisfação do cliente, melhorar a fidelidade do cliente e aumentar as vendas da marca.

### INTEGRANTES

Enzo Perazolo	<b>RM95657</b>
Giovanna Sousa	<b>RM94767</b>
Henry Kinoshita	<b>RM93443</b>
Luiz Felipe	<b>RM94538</b>
Matheus Felipe	<b>RM93772</b>
Victor Mendes	<b>RM92843</b>

Descrição do Projeto e Regras de Negócios Declaração de Visão e Escopo do Projeto  
Descrição do problema a resolver:

A jornada do cliente (CX) é uma parte crucial do sucesso de qualquer negócio. No entanto, muitas empresas lutam para fornecer uma experiência personalizada e satisfatória para seus clientes. O problema é a falta de uma solução eficiente para fornecer recomendações personalizadas e prever as necessidades dos clientes com base em seu histórico de compras e interações anteriores com a marca.

Descrição dos objetivos da solução idealizada:

O objetivo desta solução é utilizar a tecnologia de Machine Learning para criar um sistema que possa aprender com as interações anteriores dos clientes e fornecer recomendações personalizadas cada vez mais precisas. A solução deve ser capaz de prever as necessidades dos clientes, fornecendo-lhes ofertas personalizadas e melhorando sua experiência de compra.

O sistema é composto por dois componentes principais: um modelo de recomendação e um modelo de previsão:

O modelo de recomendação é treinado com dados históricos de interações do cliente

- com a marca, como compras anteriores, navegação em sites, pesquisas e outros dados relevantes. O modelo usa esses dados para gerar recomendações personalizadas para cada cliente, levando em consideração suas preferências, histórico de compras e outros fatores.

O modelo de previsão utiliza técnicas de Machine Learning para prever as necessidades

- futuras dos clientes com base em seus dados históricos. Ele considera fatores como sazonalidade, tendências de mercado e mudanças nos comportamentos dos clientes. Com base nessas previsões, o modelo é capaz de fornecer recomendações mais precisas e relevantes para os clientes, aumentando a probabilidade de que eles realizem uma compra.

Além disso, a solução também inclui um sistema de feedback que permite aos clientes

- avaliar as recomendações recebidas. Esses feedbacks são usados para ajustar e melhorar continuamente o modelo de recomendação e o modelo de previsão.

Definição do público-alvo que comprará (cliente/pagante) e/ou usará a solução (consumidor):

O público-alvo desta solução são empresas que buscam melhorar a experiência de seus clientes. Os consumidores finais são os clientes das empresas que utilizarão a solução.

As necessidades e desejos do público-alvo em relação ao problema incluem

- Personalização: os clientes desejam uma experiência personalizada e única que atenda às suas necessidades e preferências individuais.

- Facilidade de uso: os clientes querem que a experiência de compra seja fácil e intuitiva, sem barreiras ou obstáculos desnecessários.

- Agilidade: os clientes esperam uma experiência rápida e ágil, sem atrasos ou tempos de espera prolongados.

- Confiança: os clientes desejam confiar na marca e em seus produtos, ter certeza de que estão fazendo uma compra com boa relação custo-benefício.

- Comunicação clara: os clientes querem que as informações fornecidas pela marca sejam claras, precisas e relevantes para suas necessidades.

- Relevância: os clientes esperam que as recomendações e sugestões fornecidas pela marca sejam relevantes e apropriadas para seus interesses e necessidades individuais.

Personalização contínua: os clientes querem que a marca continue a aprender e ajustar

- as recomendações e sugestões ao longo do tempo, para que a experiência continue a ser personalizada e relevante.

Descrição das possíveis bibliotecas e frameworks Python que poderão ser utilizados:

TensorFlow: é uma biblioteca de código aberto para aprendizado de máquina e inteligência

- artificial desenvolvida pela Google. É amplamente utilizada para criação de modelos de redes neurais profundas e pode ser utilizado para a criação de chatbots avançados.

Keras: é uma biblioteca de alto nível para aprendizado de máquina e redes neurais

- escrita em Python, que roda em cima do TensorFlow. É uma das bibliotecas mais utilizadas para criação de modelos de redes neurais e possui uma sintaxe simples e intuitiva.

Scikit-learn: é uma biblioteca de aprendizado de máquina em Python que oferece uma

- ampla variedade de algoritmos para tarefas de classificação, regressão, clusterização, entre outras. É uma das bibliotecas mais populares para análise de dados e criação de modelos de aprendizado de máquina.

- NLTK (Natural Language Toolkit): é uma biblioteca em Python para processamento de linguagem natural. Pode ser utilizado para lidar com tarefas como tokenização, lematização, análise de sentimentos, entre outras.

- Pandas: é uma biblioteca em Python para manipulação e análise de dados. É frequentemente utilizada para limpeza, transformação e análise de dados em projetos de aprendizado de máquina e inteligência artificial.

No projeto proposto, a IA (Inteligência Artificial) será utilizada para desenvolver um sistema de recomendação personalizada para os clientes, a fim de prever suas necessidades com base em seu histórico de compras e interações anteriores com a marca. Para isso, serão utilizados algoritmos de aprendizado de máquina e redes neurais, que serão treinados com os dados históricos dos clientes, a fim de prever suas necessidades e fornecer recomendações precisas e personalizadas.

O conceito de LOT (Language of Thought) também será utilizado na solução proposta. Isso porque, a partir das interações com os clientes, será possível aprender a linguagem específica utilizada pelos clientes e criar um modelo cognitivo da linguagem desses clientes. Com essa informação, a IA poderá ser treinada para compreender melhor as necessidades e desejos dos clientes e fornecer recomendações mais precisas e personalizadas.

Além disso, a utilização da linguagem natural na interação com os clientes também será fundamental para a solução proposta. Com o uso de bibliotecas como NLTK (Natural Language Toolkit) e TensorFlow, será possível treinar modelos de processamento de linguagem natural para compreender melhor as necessidades e desejos dos clientes e fornecer recomendações mais precisas e personalizadas.

Em resumo, a IA e o conceito de LOT serão utilizados de forma integrada na solução proposta, a fim de criar um sistema de recomendação personalizada e inteligente, capaz de compreender a linguagem dos clientes e prever suas necessidades com base em seu histórico de compras e interações anteriores com a marca.

Estudo de produtos semelhantes já existentes no mercado (que solucionam o mesmo problema, mesmo que de forma diferente, parcial ou totalmente):

Há muitas soluções de recomendação personalizadas no mercado, desde sistemas de recomendação de filmes até recomendações de produtos em lojas online. No entanto, a maioria dessas soluções não utiliza Machine Learning para melhorar a precisão das recomendações e não levam em conta o histórico completo de interações com a marca. A solução proposta irá superar essas limitações.

- Amazon Personalize: a Amazon oferece um serviço de personalização de recomendação que permite que as empresas personalizem suas recomendações de produtos com base no histórico de compras e no comportamento do usuário.
- Salesforce Einstein: o Salesforce Einstein é uma plataforma de inteligência artificial que ajuda as empresas a oferecer recomendações personalizadas para seus clientes.
- Adobe Target: a Adobe Target é uma plataforma de personalização que usa dados do cliente e inteligência artificial para fornecer recomendações personalizadas.
- Google Analytics 360: o Google Analytics 360 é uma plataforma de análise de dados que ajuda as empresas a entender o comportamento do cliente e fornecer recomendações personalizadas.
- Dynamic Yield: a Dynamic Yield é uma plataforma de personalização omnichannel que usa aprendizado de máquina para fornecer recomendações personalizadas e experiências personalizadas para os clientes.

Para se destacar em relação aos produtos concorrentes, a solução proposta pode explorar as seguintes oportunidades de diferenciação:

- Integração com outras ferramentas de CX: a solução pode se integrar com outras ferramentas de CX (Customer Experience), como plataformas de e-commerce e ferramentas de análise de dados, para fornecer uma experiência de compra mais integrada e completa.
- Foco em segmentos específicos de mercado: a solução pode se concentrar em atender a segmentos específicos de mercado, como nichos de mercado ou setores da indústria, com recursos personalizados e adaptados às suas necessidades.
- Monitoramento contínuo do feedback do cliente: a solução pode incorporar recursos de monitoramento e análise contínua do feedback do cliente para garantir que as recomendações e sugestões permaneçam relevantes e atualizadas ao longo do tempo.



Avaliação do potencial de mercado (fatia de mercado a conquistar):

O mercado para soluções de recomendação personalizadas está em crescimento, com muitas empresas buscando maneiras de melhorar a experiência do cliente. A solução proposta tem potencial para conquistar uma fatia significativa do mercado, uma vez que será capaz de fornecer recomendações cada vez mais precisas e personalizadas, melhorando a satisfação do cliente e aumentando a fidelidade do cliente para com a marca.

1. Aumento da demanda por experiências de compra personalizadas: os consumidores estão cada vez mais em busca de experiências de compra personalizadas, o que cria uma oportunidade para soluções que utilizam aprendizado de máquina e análise de dados para fornecer recomendações e sugestões personalizadas.

2. Crescimento do mercado de CX: o mercado de CX está em constante crescimento e deve continuar crescendo nos próximos anos, criando uma oportunidade para soluções que oferecem uma experiência de compra mais personalizada e integrada.

3. Ampliação da adoção de tecnologias de IA: a adoção de tecnologias de inteligência artificial, como aprendizado de máquina, está em constante crescimento, o que cria uma oportunidade para soluções que utilizam essas tecnologias para fornecer uma experiência de compra mais personalizada e adaptada às necessidades do cliente.

4. Diferenciação em relação aos concorrentes: oferecer recursos e funcionalidades exclusivas e diferentes dos concorrentes pode ajudar a solução proposta a se destacar no mercado e conquistar uma fatia maior de clientes em busca de uma experiência de compra personalizada.

5. Expansão para novos mercados: a solução proposta pode ser expandida para novos mercados e setores da indústria, adaptando-se às necessidades e demandas específicas de cada segmento e aumentando seu potencial de crescimento no mercado.

A solução proposta cria valor para o público-alvo de diversas formas:

- Melhora a experiência do cliente: Ao fornecer recomendações mais precisas e personalizadas, a solução melhora a experiência do cliente, aumentando a satisfação e a fidelidade à marca.

- Economiza tempo: Ao antecipar as necessidades do cliente, a solução economiza tempo do cliente ao tornar mais rápida e fácil a busca por produtos que atendam suas necessidades.
- Aumenta a relevância: Ao fornecer recomendações personalizadas, a solução aumenta a relevância das ofertas, aumentando a probabilidade de o cliente comprar e, consequentemente, aumentando as vendas da marca.
- Melhora a eficiência: Ao utilizar técnicas de inteligência artificial, a solução pode processar grandes quantidades de dados de forma eficiente e automatizada, permitindo que a marca ofereça recomendações personalizadas para um grande número de clientes simultaneamente.
- Estimula a compra recorrente: A solução pode ser utilizada para incentivar a compra recorrente, fornecendo recomendações para produtos complementares e incentivando o cliente a comprar novamente.

#### Definindo os modelos de receita para a solução

Existem diferentes modelos de receita que podem ser utilizados para a solução proposta, dependendo da estratégia de negócio da marca. Abaixo, estão alguns modelos que podem ser considerados:

- Venda de software: A marca pode vender a solução como um software para outras empresas, permitindo que elas usem as funcionalidades de inteligência artificial em suas próprias plataformas.
- Assinatura: A marca pode oferecer a solução como um serviço de assinatura, cobrando dos clientes um valor mensal ou anual para ter acesso às funcionalidades de recomendação personalizada.
- Comissão sobre vendas: A marca pode cobrar uma comissão sobre as vendas realizadas a partir das recomendações geradas pela solução.
- Publicidade: A marca pode utilizar a solução para exibir anúncios relevantes aos clientes, cobrando dos anunciantes uma taxa de publicidade.
- Licenciamento de tecnologia: A marca pode licenciar a tecnologia de inteligência artificial desenvolvida para outras empresas, permitindo que elas desenvolvam suas próprias soluções de recomendação personalizada.



### Regras de Negócio

RN01: nossos clientes serão chamados de parceiros de negócios;

- RN02: os clientes dos nossos clientes (ou seja, os compradores) serão chamados de usuários;

RN03: o parceiro será identificado por um código numérico;

RN04: o parceiro deve informar um nome fantasia;

- RN05: deverá existir uma coluna de data para informar a entrada deste parceiro em nosso sistema, bem como uma coluna informando a data de encerramento de contrato com este parceiro;

- RN06: a data de encerramento deverá ser obrigatoriamente preenchida somente quando se encerrar o contrato com o parceiro de negócios;

RN07: o parceiro deve informar um CNPJ;

- RN08: se o parceiro for um ecommerce, ele pode informar o link do produto comprado

RN09: o usuário será identificado por um código numérico;

RN10: o parceiro deve informar um nome para o usuário;

RN11: o parceiro deve informar o CPF do usuário;

RN12: o parceiro pode informar o CEP da residência do usuário;

RN13: a data de nascimento do usuário pode ser informada;

RN14: o gênero do usuário pode ser informado;

RN15: o item de compra será chamado produto;

RN16: o produto será identificado por um código numérico;

RN17: o produto deve ter uma categoria;

RN18: o produto deverá ter um nome;

RN19: o produto deverá ter uma categoria;

RN20: um parceiro pode ter vários produtos;

RN21: a transação será identificada por um código numérico;

- RN22: cada transação deve conter o código do usuário e o código do parceiro de negócios;

RN23: cada transação tem obrigatoriamente um usuário e um parceiro; RN24: uma transação pode ter mais de um produto;

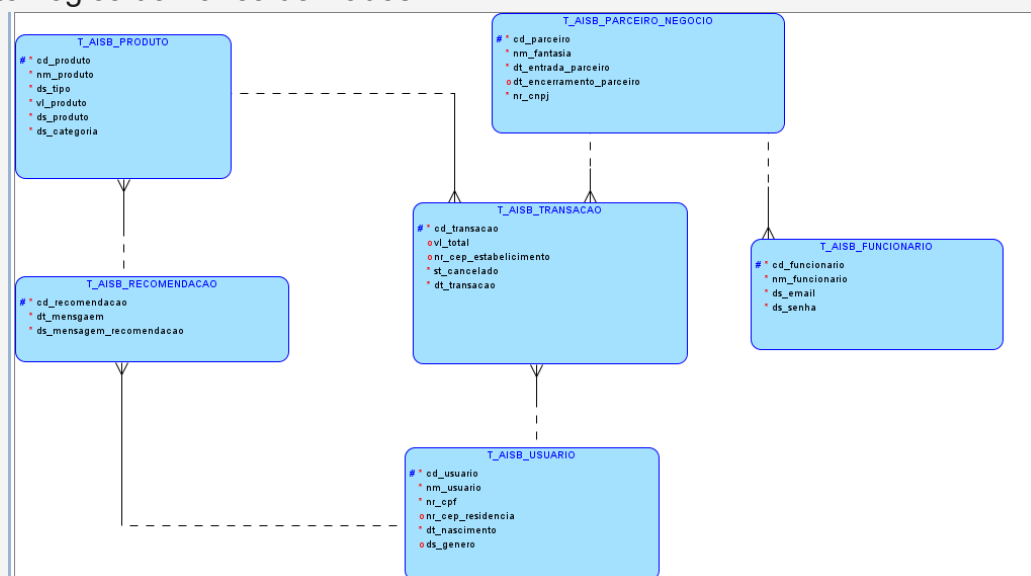
RN25: a data da transação deve ser mantida;

RN26: um usuário pode ter várias transações;

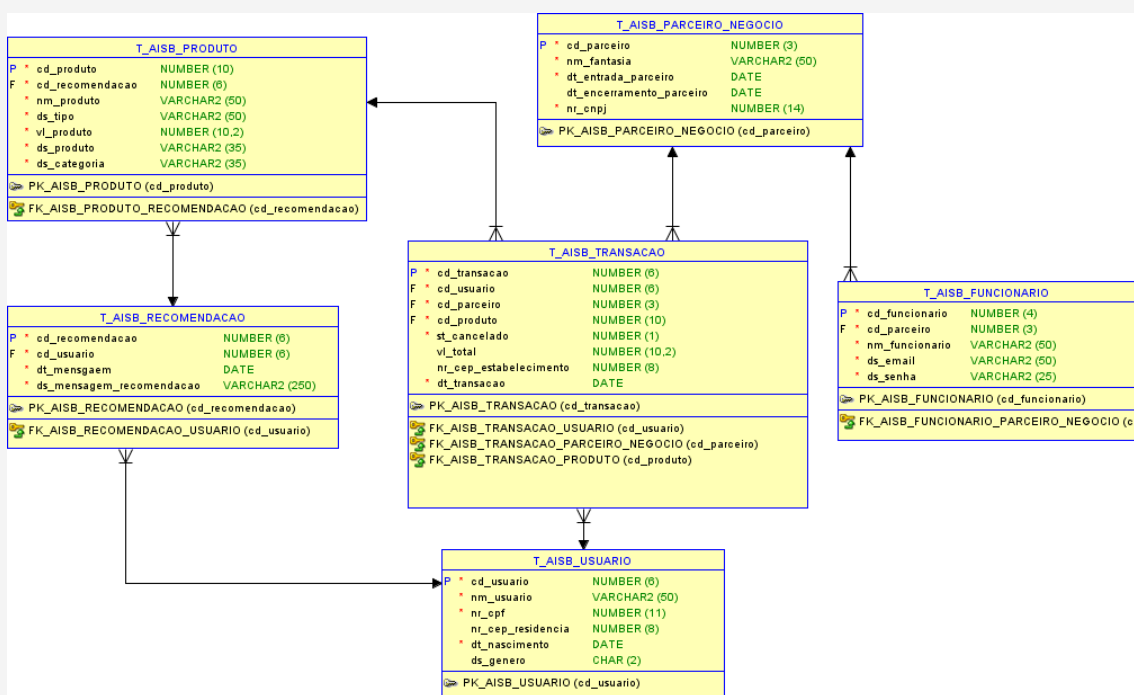
-

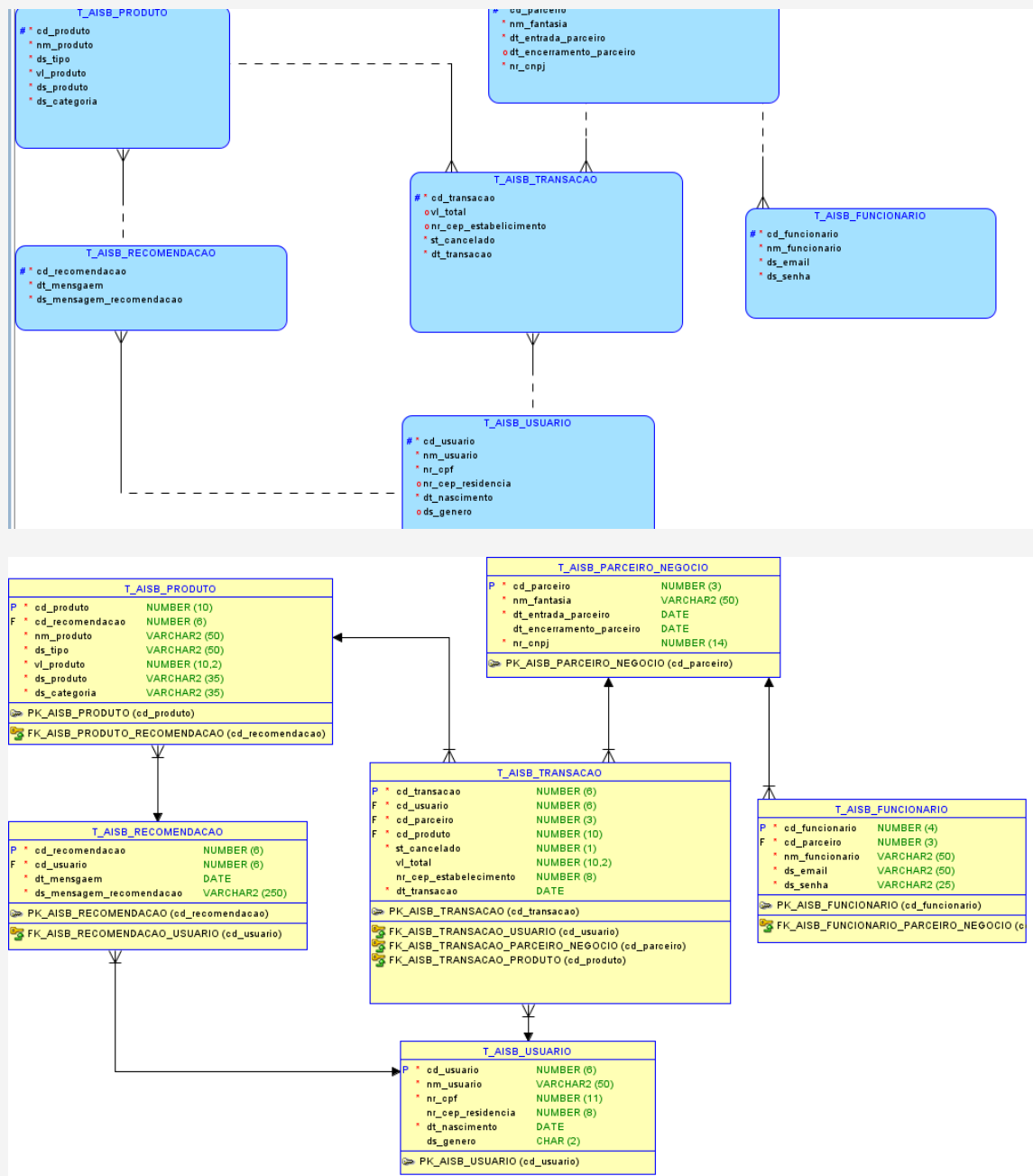
- RN27: um parceiro pode ter várias transações;
- RN28: o valor de cada produto deve ser apresentado individualmente;
- RN29: o valor total da compra deve ser apresentado;
- RN30: a transação pode informar o CEP do estabelecimento da compra;
- RN31: o status da transação deve ser apresentado em valor booleano (sim ou não);
- RN32: cada recomendação é identificada por um código;
- RN33: cada recomendação possui um usuário, mas um usuário pode ter várias recomendações;
- RN34: a recomendação tem obrigatoriamente uma data de quando foi feita;
- RN35: a mensagem de recomendação feita pela API do ChatGPT deve ser salva em uma coluna;
- RN36: uma recomendação pode ter vários produtos;
- RN37: o funcionário deve ser identificado por um código numérico;
- RN38: o parceiro deve informar o nome do funcionário e seu email;
- RN39: um parceiro pode ter um ou mais funcionários cadastrados;
- RN40: um parceiro pode ter várias recomendações de um mesmo usuário;

## Projeto Lógico do Banco de Dados



## Projeto Relacional do Banco de Dados





**Função que faz a verificação ao cadastrar um funcionário, em que, caso esse funcionário já exista no sistema armazenar as informações na tabela de erro, e caso ele não exista, fará o cadastro do novo funcionário com as novas informações.**

```
CREATE OR REPLACE FUNCTION cadastrar_funcionario(
    p_cd_funcionario IN NUMBER,
    p_nm_funcionario IN VARCHAR2,
    p_ds_email IN VARCHAR2,
    p_ds_senha IN VARCHAR2,
    p_cd_parceiro IN NUMBER
) RETURN VARCHAR2
IS
    v_parceiro_exist NUMBER(1) := 0;
BEGIN
    SELECT COUNT(*) INTO v_parceiro_exist
    FROM t_aishb_parceiro_negocio
    WHERE cd_parceiro = p_cd_parceiro;

    IF v_parceiro_exist = 0 THEN
        INSERT INTO t_aishb_erro (cd_erro, nm_erro, dt_ocorrencia, nm_usuario)
        VALUES (t_aishb_erro_seq.nextval, 'Parceiro de negócios não encontrado', SYSDATE, USER);

        RETURN 'Não foi possível cadastrar esse funcionário pois o parceiro de negócios não foi encontrado.';
    END IF;

    INSERT INTO t_aishb_funcionario (cd_funcionario, nm_funcionario, ds_email, ds_senha, parceiro_fk)
    VALUES (p_cd_funcionario, p_nm_funcionario, p_ds_email, p_ds_senha, p_cd_parceiro);

    RETURN 'O funcionário foi cadastrado com sucesso.';
EXCEPTION
    WHEN OTHERS THEN
        INSERT INTO t_aishb_erro (cd_erro, nm_erro, dt_ocorrencia, nm_usuario)
        VALUES (t_aishb_erro_seq.nextval, 'Erro ao cadastrar funcionário', SYSDATE, USER);

        RETURN 'Ocorreu um erro ao cadastrar o funcionário.';
END cadastrar_funcionario;
```

```
DECLARE
    v_cd_funcionario NUMBER(4);
    v_nm_funcionario VARCHAR2(50);
    v_ds_email VARCHAR2(50);
    v_ds_senha VARCHAR2(25);
    v_cd_parceiro NUMBER(4);
    v_resultado VARCHAR2(200);
BEGIN
    v_cd_funcionario := scd_funcionario;
    v_nm_funcionario := 'snome';
    v_ds_email := 'semail';
    v_ds_senha := 'ssenha';
    v_cd_parceiro := scd_parceiro;

    v_resultado := cadastrar_funcionario(v_cd_funcionario, v_nm_funcionario, v_ds_email, v_ds_senha, v_cd_parceiro);
    DEMS_OUTPUT.PUT_LINE(v_resultado);
END;
```

**Função que faz a verificação ao cadastrar um funcionário, em que, caso esse funcionário já exista no sistema armazenar as informações na tabela de erro, e caso ele não exista, fará o cadastro do novo funcionário com as novas informações.**

Enter Substitution Variable

Enter value for cd\_funcionario:

OK Cancel

Enter Substitution Variable

Enter value for nome:

OK Cancel

Enter Substitution Variable

Enter value for email:

OK Cancel

Enter Substitution Variable

Enter value for senha:

OK Cancel

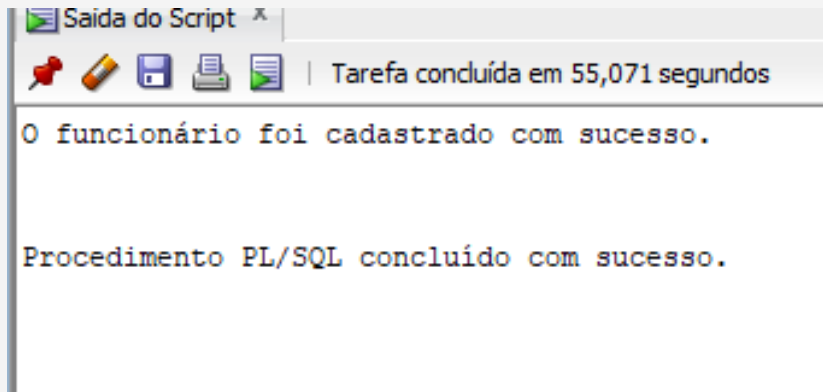
Digitar Variável de Substituição

Informe o valor para cd\_parceiro:

OK Cancelar



**Função que faz a verificação ao cadastrar um funcionário, em que, caso esse funcionário já exista no sistema armazenar as informações na tabela de erro, e caso ele não exista, fará o cadastro do novo funcionário com as novas informações.**



```

Saída do Script
Tarefa concluída em 55,071 segundos

O funcionário foi cadastrado com sucesso.

Procedimento PL/SQL concluído com sucesso.
  
```

CD_FUNCION...	NM_FUNCIONARIO	DS_EMAIL	DS_SENHA	PARCEIRO_FK
2	121 Marcos Belutti	marcosebelutti@gmail.com	124	101
3	122 Matheus Cauan	matheusecauan@gmail.com	125	102
4	123 Luan Santana	lu.santana@gmail.com	126	103
5	124 Victor Matheus	matheusvictor@gmail.com	127	104
6	125 Karol Dantas	karol.dantas@gmail.com	128	105
7	126 Gustavo Lima	gustavolima@gmail.com	129	106
8	127 Gizele Almeida	gizelealmeida@gmail.com	130	107
9	128 Ana Castelo	anacastelo@gmail.com	131	108
10	129 Cleide Santos	cleidesanto@gmail.com	132	109
11	1 Felipe	felipe@gmail	1306	100

**--Função que faz a verificação ao cadastrar um usuário, em que, caso esse usuário já exista no sistema armazenar as informações na tabela de erro, e caso ele não exista, fará o cadastro do novo usuário com as novas informações**

```
CREATE OR REPLACE FUNCTION cadastrar_usuario(
    p_cd_usuario IN NUMBER,
    p_ds_genero IN CHAR,
    p_nm_usuario IN VARCHAR2,
    p_nr_cpf IN NUMBER,
    p_nr_cep_residencia IN NUMBER,
    p_dt_nascimento IN DATE
) RETURN VARCHAR2
IS
    v_result VARCHAR2(100);
    v_cd_erro NUMBER(3);
BEGIN
    SELECT 'Usuário já cadastrado'
    INTO v_result
    FROM t_aish_usuario
    WHERE cd_usuario = p_cd_usuario;

    SELECT NVL(MAX(cd_erro), 0) + 1 INTO v_cd_erro FROM t_aish_erro;

    INSERT INTO t_aish_erro (cd_erro, nm_erro, dt_ocorrencia, nm_usuario)
    VALUES (v_cd_erro, 'Tentativa de duplicação de usuário', SYSDATE, USER);

    RETURN 'Não foi possível cadastrar esse usuário pois ele já existe no sistema.';
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        INSERT INTO t_aish_usuario (cd_usuario, ds_genero, nm_usuario, nr_cpf, nr_cep_residencia, dt_nascimento)
        VALUES (p_cd_usuario, p_ds_genero, p_nm_usuario, p_nr_cpf, p_nr_cep_residencia, p_dt_nascimento);

        RETURN '0 usuário foi cadastrado com sucesso.';
END cadastrar_usuario;
```

```
DECLARE
    v_cd_usuario NUMBER(6);
    v_ds_genero CHAR(2);
    v_nm_usuario VARCHAR2(50);
    v_nr_cpf NUMBER(11);
    v_nr_cep_residencia NUMBER(8);
    v_dt_nascimento DATE;
    v_resultado VARCHAR2(200);
BEGIN
    v_cd_usuario := &codigo_usuario;
    v_ds_genero := '&genero';
    v_nm_usuario := '&nome';
    v_nr_cpf := &cpf;
    v_nr_cep_residencia := &cep;
    v_dt_nascimento := TO_DATE('&v_dt_nascimento', 'DD/MM/YYYY');

    v_resultado := cadastrar_usuario(
        v_cd_usuario, v_ds_genero, v_nm_usuario,
        v_nr_cpf, v_nr_cep_residencia, v_dt_nascimento
    );

    DBMS_OUTPUT.PUT_LINE(v_resultado);
END;
```

**--Função que faz a verificação ao cadastrar um usuário, em que, caso esse usuário já exista no sistema armazenar as informações na tabela de erro, e caso ele não exista, fará o cadastro do novo usuário com as novas informações**

Enter Substitution Variable

Enter value for codigo\_usuario:

OK Cancel

Enter Substitution Variable

Enter value for genero:

OK Cancel

Enter Substitution Variable

Enter value for nome:

OK Cancel

Enter Substitution Variable

Enter value for cpf:

OK Cancel

Enter Substitution Variable

Enter value for cep:

OK Cancel

Enter Substitution Variable

Enter value for v\_dt\_nascimento:

OK Cancel

O usuário foi cadastrado com sucesso.

PL/SQL procedure successfully completed.

**Função que faz a verificação ao cadastrar um usuário, em que, caso esse usuário já exista no sistema armazenar as informações na tabela de erro, e caso ele não exista, fará o cadastro do novo usuário com as novas informações**

	CD_USUARIO	DS_GENERO	NM_USUARIO	NR_CPF	NR_CEP_RESIDENCIA	DT_NASCIMENTO
2	100001	M	Giulia Guedes	28928875269	64049815	12-FEB-02
3	100002	M	Lourdes Santos	22654215296	56909440	31-DEC-90
4	100003	NB	Victor Mendes	52136277650	69911460	09-DEC-03
5	100004	M	Glória Maria	38875303304	34710040	21-SEP-98
6	100005	H	Mathes Felipe	45147765565	65606660	24-JAN-90
7	100006	H	Henry Rodrigues	26025840296	65040110	22-MAY-02
8	100007	H	João Carlos	63638173232	77826474	30-JUL-03
9	100008	M	Cármen Dias	62438480351	29055355	30-JUL-03
10	100009	H	Lucas Fernandes	18363724858	68911045	30-JUL-03
11	1	H	Felipe	39566053819	7400605	13-JUN-04

## Procedure para verificar e atualizar a data de encerramento do parceiro de negócio

```
CREATE OR REPLACE PROCEDURE atualizar_encerramento_parceiro (
    p_cd_parceiro NUMBER,
    p_dt_encerramento DATE
) IS
    v_cd_parceiro t_aish_parceiro_negocio.cd_parceiro%TYPE;
    v_cd_erro NUMBER(3);
BEGIN
    SELECT cd_parceiro INTO v_cd_parceiro
    FROM t_aish_parceiro_negocio
    WHERE cd_parceiro = p_cd_parceiro;

    UPDATE t_aish_parceiro_negocio
    SET dt_encerramento_parceiro = p_dt_encerramento
    WHERE cd_parceiro = v_cd_parceiro;

    COMMIT;

    SELECT COALESCE(MAX(cd_erro) + 1, 1) INTO v_cd_erro FROM t_aish_erro;

    INSERT INTO t_aish_erro (cd_erro, nm_erro, dt_ocorrencia, nm_usuario)
    VALUES (v_cd_erro, 'Data de encerramento atualizada com sucesso', SYSDATE, USER);

    DBMS_OUTPUT.PUT_LINE('Data de encerramento atualizada com sucesso.');
```

```
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        SELECT COALESCE(MAX(cd_erro) + 1, 1) INTO v_cd_erro FROM t_aish_erro;
        INSERT INTO t_aish_erro (cd_erro, nm_erro, dt_ocorrencia, nm_usuario)
        VALUES (v_cd_erro, 'Parceiro não encontrado para atualização', SYSDATE, USER);
        COMMIT;

        DBMS_OUTPUT.PUT_LINE('Parceiro não encontrado. Erro registrado na tabela de erro.');
```

```
    WHEN OTHERS THEN
        SELECT COALESCE(MAX(cd_erro) + 1, 1) INTO v_cd_erro FROM t_aish_erro;
        INSERT INTO t_aish_erro (cd_erro, nm_erro, dt_ocorrencia, nm_usuario)
        VALUES (v_cd_erro, 'Erro ao atualizar data de encerramento', SYSDATE, USER);
        COMMIT;

        DBMS_OUTPUT.PUT_LINE('Erro ao atualizar data de encerramento. Erro registrado na tabela de erro.');
```

```
END atualizar_encerramento_parceiro;
```

```
DECLARE
    v_cd_parceiro NUMBER(3);
    v_dt_encerramento DATE := TO_DATE('2023-08-31', 'YYYY-MM-DD');
BEGIN
    v_cd_parceiro := &v_cd_parceiro;

    atualizar_encerramento_parceiro(v_cd_parceiro, v_dt_encerramento);
END;
```

## Procedure para verificar e atualizar a data de encerramento do parceiro de negócio

	CD_PARCEIRO	NM_FANTASIA	DT_ENTRADA_PARCEIRO	DT_ENCERRAMENTO_PARCEIRO	NR_CNPJ
1	100	Amazon	10-MAY-22	(null)	95831661000175
2	101	Roupas legais	22-FEB-22	(null)	28160686000105
3	102	Americanas	09-DEC-21	21-FEB-23	22211739000185
4	103	Apple	30-JUL-17	(null)	49181554000121
5	104	Submarino	22-MAY-20	(null)	58359222000185
6	105	Couto Esportes	13-JUN-21	01-JAN-23	29539514000100
7	106	Casas Bahia	24-JAN-22	(null)	33547947000176
8	107	Paif	31-DEC-20	30-AUG-22	52234883000106
9	108	Magazine Luiza	16-JUN-16	(null)	11751264000101
10	109	Pernambucanas	20-FEB-20	15-MAY-21	12218899000100

Data de encerramento atualizada com sucesso.

PL/SQL procedure successfully completed.

## Nesse procedure foi feita a atualização da data de encerramento do parceiro "Amazon"

	CD_PARCEIRO	NM_FANTASIA	DT_ENTRADA_PARCEIRO	DT_ENCERRAMENTO_PARCEIRO	NR_CNPJ
1	100	Amazon	10-MAY-22	31-AUG-23	95831661000175
2	101	Roupas legais	22-FEB-22	(null)	28160686000105
3	102	Americanas	09-DEC-21	21-FEB-23	22211739000185
4	103	Apple	30-JUL-17	(null)	49181554000121
5	104	Submarino	22-MAY-20	(null)	58359222000185
6	105	Couto Esportes	13-JUN-21	01-JAN-23	29539514000100
7	106	Casas Bahia	24-JAN-22	(null)	33547947000176
8	107	Paif	31-DEC-20	30-AUG-22	52234883000106
9	108	Magazine Luiza	16-JUN-16	(null)	11751264000101
10	109	Pernambucanas	20-FEB-20	15-MAY-21	12218899000100



**Procedure que irá exibir um relatório de todos os parceiros que encerram o contrato conosco e caso dê algum erro, ele irá inserir na tabela de erro.**

```
CREATE OR REPLACE PROCEDURE relatorio_parceiros_encerrados AS
    v_cd_erro NUMBER;
    v_nm_erro VARCHAR2(50);
BEGIN
    DBMS_OUTPUT.PUT_LINE('Relatório de Parceiros Encerrados');
    DBMS_OUTPUT.PUT_LINE('-----');

    BEGIN
        FOR rec IN (SELECT cd_parceiro, nm_fantasia, dt_entrada_parceiro, dt_encerramento_parceiro, nr_cnpj
                     FROM t_aisb_parceiro_negocio
                     WHERE dt_encerramento_parceiro IS NOT NULL) LOOP

            DBMS_OUTPUT.PUT_LINE('Código do Parceiro: ' || rec.cd_parceiro);
            DBMS_OUTPUT.PUT_LINE('Nome Fantasia: ' || rec.nm_fantasia);
            DBMS_OUTPUT.PUT_LINE('Data de Entrada: ' || TO_CHAR(rec.dt_entrada_parceiro, 'DD/MM/YYYY'));
            DBMS_OUTPUT.PUT_LINE('Data de Encerramento: ' || TO_CHAR(rec.dt_encerramento_parceiro, 'DD/MM/YYYY'));
            DBMS_OUTPUT.PUT_LINE('CNPJ: ' || rec.nr_cnpj);
            DBMS_OUTPUT.PUT_LINE('-----');

        END LOOP;

        DBMS_OUTPUT.PUT_LINE('-----');
        DBMS_OUTPUT.PUT_LINE('Fim do Relatório');

    EXCEPTION
        WHEN OTHERS THEN
            v_cd_erro := -1;
            v_nm_erro := SQLERRM;

            INSERT INTO t_aisb_erro (cd_erro, nm_erro, dt_ocorrencia, nm_usuario)
            VALUES (v_cd_erro, v_nm_erro, SYSDATE, USER);

            DBMS_OUTPUT.PUT_LINE('Ocorreu um erro durante a execução da procedure. ');
            DBMS_OUTPUT.PUT_LINE('Detalhes do erro: ' || v_nm_erro);

        END;
END relatorio_parceiros_encerrados;

BEGIN
    relatorio_parceiros_encerrados;
END;
```

**Procedure que irá exibir um relatório de todos os parceiros que encerram o contrato conosco e caso dê algum erro, ele irá inserir na tabela de erro.**

```
Relatório de Parceiros Encerrados
-----
Código do Parceiro: 100
Nome Fantasia: Amazon
Data de Entrada: 10/05/2022
Data de Encerramento: 31/08/2023
CNPJ: 95831661000175
-----
Código do Parceiro: 102
Nome Fantasia: Americanas
Data de Entrada: 09/12/2021
Data de Encerramento: 21/02/2023
CNPJ: 22211739000185
-----
Código do Parceiro: 105
Nome Fantasia: Couto Esportes
Data de Entrada: 13/06/2021
Data de Encerramento: 01/01/2023
CNPJ: 29539514000100
-----
Código do Parceiro: 107
Nome Fantasia: Paif
Data de Entrada: 31/12/2020
Data de Encerramento: 30/08/2022
CNPJ: 52234883000106
-----
Código do Parceiro: 109
Nome Fantasia: Pernambucanas
Data de Entrada: 20/02/2020
Data de Encerramento: 15/05/2021
CNPJ: 12218899000100
-----
Fim do Relatório

PL/SQL procedure successfully completed.
```

## Trigger para atualizar a senha do funcionário

```
CREATE OR REPLACE TRIGGER trg_before_atualizacao_senha
BEFORE UPDATE ON t_aish_funcionario
FOR EACH ROW
BEGIN
    IF :OLD.ds_senha <> :NEW.ds_senha THEN
        INSERT INTO t_atualizacao_senha (cd_funcionario, senha_anterior, senha_nova, data_atualizacao)
        VALUES (:OLD.cd_funcionario, :OLD.ds_senha, :NEW.ds_senha, SYSTIMESTAMP);
    END IF;
END;

UPDATE t_aish_funcionario
SET ds_senha = 'nova_senha'
WHERE cd_funcionario = 123;

select * from t_atualizacao_senha;
select * from t_aish_funcionario;
```

Nesse trigger ele irá atualizar a senha do funcionário Luan Santana de código(123)

	CD_FUNCIONARIO	NM_FUNCIONARIO	DS_EMAIL	DS_SENHA	PARCEIRO_FK
1	120	Valerio Durães	duraesvalerio@gmail.com	123	100
2	121	Marcos Belutti	marcosebelutti@gmail.com	124	101
3	122	Matheus Cauan	matheusecauan@gmail.com	125	102
4	123	Luan Santana	lu.santana@gmail.com	126	103
5	124	Victor Matheus	matheusvictor@gmail.com	127	104
6	125	Karol Dantas	karol.dantas@gmail.com	128	105
7	126	Gustavo Lima	gustavolima@gmail.com	129	106
8	127	Gizele Almeida	gizelealmeida@gmail.com	130	107
9	128	Ana Castelo	anacastelo@gmail.com	131	108
10	129	Cleide Santos	cleidesanto@gmail.com	132	109

## Trigger para atualizar a senha do funcionário

```
select * from t_atualizacao_senha;  
select * from t_aisb_funcionario;
```

Script Output x Query Result x  
SQL | All Rows Fetched: 1 in 0.006 seconds

CD_FUNCIONARIO	SENHA_ANTERIOR	SENHA_NOVA	DATA_ATUALIZACAO
1	123 126	nova_senha	12-SEP-23

	CD_FUNCIONARIO	NM_FUNCIONARIO	DS_EMAIL	DS_SENHA	PARCEIRO_FK
1	120	Valerio Durães	duraesvalerio@gmail.com	123	100
2	121	Marcos Belutti	marcosebelutti@gmail.com	124	101
3	122	Matheus Cauan	matheusecauan@gmail.com	125	102
4	123	Luan Santana	lu.santana@gmail.com	nova_senha	103
5	124	Victor Matheus	matheusvictor@gmail.com	127	104
6	125	Karol Dantas	karol.dantas@gmail.com	128	105
7	126	Gustavo Lima	gustavolima@gmail.com	129	106
8	127	Gizele Almeida	gizelealmeida@gmail.com	130	107
9	128	Ana Castelo	anacastelo@gmail.com	131	108
10	129	Cleide Santos	cleidesanto@gmail.com	132	109