



Faculdade de Informática e Administração Paulista

Database Application AND Data Science

INTEGRANTES

RM	NOME COMPLEMENTO
93596	Eduarda Vieira Gomes
94278	Leonardo Silva Macedo
94275	Lívia Carvalho Keller
95324	Luisa Gabriele da Purificação
94395	Samuel Pereira Nascimento

DSL³

Sumário

1 – Descrição do Projeto e Regras de Negócio	05
2 – Dicionário de Dados	08
3 – Projeto Lógico do Banco de Dados	11
4 – Projeto Físico do Banco de Dados	12
5 – Data Definition Language – DDL	13
6 – Data Manipulation Language – DML	17

1 – Descrição do Projeto e Regras de Negócio

SOLUÇÃO

Introduzindo o PlantAI: sua solução inteligente para o cultivo saudável. Nosso aplicativo revolucionário utiliza a mais recente tecnologia de imagem para capturar fotos detalhadas das suas plantações. Não apenas isso, o PlantAI vai além: com a ajuda da inteligência artificial, ele identifica doenças que possam estar afetando suas plantas. Uma vez identificadas, as imagens são enviadas para nosso sistema baseado em IA, alimentado pelo poderoso GPT (Generative AI), que analisa os padrões e recomenda o melhor tratamento para curar suas plantas. Com parcerias estratégicas com líderes na indústria de produtos agrícolas, o PlantAI não apenas cuida das suas colheitas, mas também oferece uma maneira inovadora de otimizar seu processo de cultivo.

Além da detecção e tratamento de doenças, o PlantAI oferece um recurso de histórico abrangente. Cada diagnóstico, imagem e tratamento recomendado são armazenados em uma visualização de histórico intuitiva. Isso não apenas ajuda você a acompanhar o progresso das suas plantações, mas também a tomar decisões informadas temporada após temporada. Através de estatísticas, você pode entender melhor os padrões de doenças que afetam suas colheitas e ajustar suas práticas agrícolas de acordo.

Nossa abordagem inovadora não apenas transforma a maneira como você cuida das suas plantações, mas também oferece uma oportunidade única de monetização. Trabalhamos em estreita colaboração com parceiros renomados na indústria de insumos agrícolas, que fornecem tratamentos recomendados específicos para cada diagnóstico. Essas soluções de parceiros são oferecidas aos usuários como opções de tratamento, criando um ecossistema vantajoso para todos os envolvidos. Ao escolher uma solução parceira, você estará apoiando o PlantAI e permitindo que continuemos a oferecer uma plataforma de qualidade para pequenos agricultores.

No PlantAI, nossa missão é melhorar a eficiência e a produtividade nas plantações, enquanto fornecemos aos agricultores as ferramentas necessárias para tomar decisões fundamentadas e obter os melhores resultados. Experimente o

futuro da agricultura com o PlantAI - onde a tecnologia encontra a terra para colher o sucesso.

Regras de Negócio

RN 01 - Para usar nosso aplicativo, é necessário ter um CNPJ, não trabalhamos com pessoa física. O empreendedor corresponde ao funcionário da empresa.

RN 02 - O aplicativo deve permitir que os usuários acompanhem o histórico de análises de plantações e do solo e assim facilitar a visualização desses dados.

RN 03 - O aplicativo deve se integrar com o ChatGPT, para sanar possíveis dúvidas dos usuários e assim fortalecer o suporte ao cliente.

RN 04 - Reconhecimento de imagem: para realizar a análise do solo e das plantações, o usuário deve tirar uma foto.

RN 05 - O aplicativo deve ser atualizado regularmente com novos recursos e melhorias para garantir que os usuários tenham acesso às tecnologias e funcionalidades mais recentes e eficientes.

RN 06 - O aplicativo deve apresentar a tela de métricas que controlará o estoque de acordo com os dados imputados pelo usuário, assim facilitando a visualização de perdas.

RN 07 - Alertas climáticos: o aplicativo pode incluir alertas climáticos para informar aos usuários sobre condições meteorológicas adversas que possam afetar o crescimento de suas plantas.

RN 08 - As plantações, após análise, serão categorizadas e a partir disso o usuário poderá filtrar por plantas curadas, em tratamento e todas as categorias existentes.

RN 09 - A usabilidade do aplicativo deve ser fluida, para que a jornada do usuário seja a melhor possível e assim satisfazendo suas expectativas.

RN 10 - Recomendações personalizadas de fertilizantes e soluções para plantas danificadas com base nas condições do solo e das plantas, através da foto enviada.

2 – Dicionário de Dados

Tabela	CLIENTE			
Descrição	Tabela de Cliente. Relaciona-se com a tabela TELEFONE, LOGIN, PROBLEMA			
Coluna	Tipo de Dados	Tamanho	Constraint	Descrição
id_cliente	NUMBER	10	PK	Identificador único do cliente
id_login	NUMBER	10	FK	Identificador único do login
nm_cliente	VARCHAR	80	NN	nome do cliente
ds_genero	VARCHAR	20	NN	descrição do genero
ds_email	VARCHAR	150	NN	descrição do email
nr_cpf	NUMBER	11	NN	CNPJ do cliente
nr_cnpj	NUMBER	14	NN	CNPJ do cliente
dt_nascimento	date	-	NN	data de nascimento
dt_cadastro	date	-	NN	data de cadastro
st_cliente	VARCHAR	1	NN	status do cliente

Tabela	LOGIN			
Descrição	Tabela de Empresas. Relaciona-se com a tabela EMPREENDEDOR			
Coluna	Tipo de Dados	Tamanho	Constraint	Descrição
id_login	NUMBER	10	PK	Identificador único do login
ds_email	VARCHAR	80	NN	email do login
ds_senha	VARCHAR	15	NN	senha do login

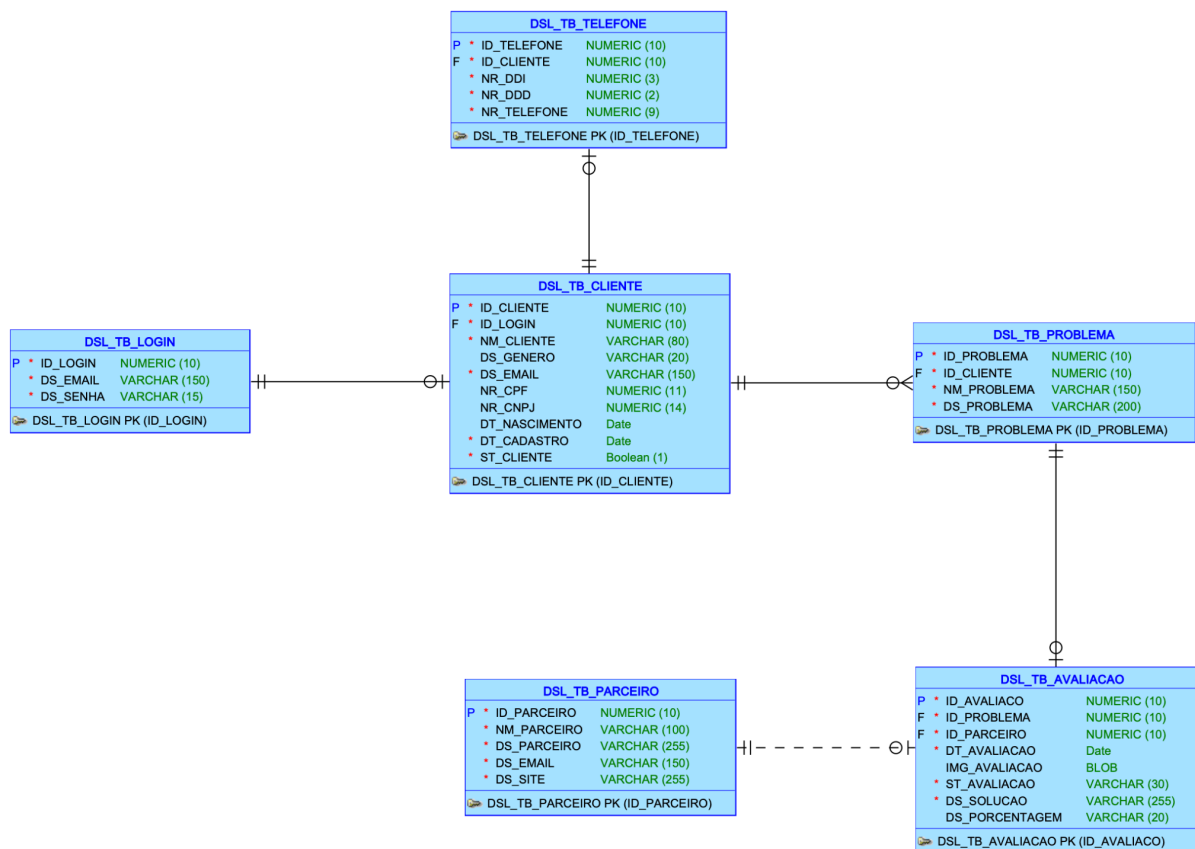
Tabela	PARCEIRO			
Descrição	Tabela de Empresas. Relaciona-se com a tabela PROBLEMA			
Coluna	Tipo de Dados	Tamanho	Constraint	Descrição
id_parceiro	NUMBER	10	PK	Identificador único do parceiro
nm_parceiro	VARCHAR	100	NN	nome do parceiro
ds_email	VARCHAR	150	NN	descrição do email do parceiro
ds_parceiro	VARCHAR	255	NN	descrição do parceiro
ds_site	VARCHAR	255	NN	descrição do site do parceiro

Tabela	PROBLEMA			
Descrição	Tabela de Empresas. Relaciona-se com a tabela CLIENTE E AVALIACAO			
Coluna	Tipo de Dados	Tamanho	Constraint	Descrição
id_problema	NUMBER	10	PK	Identificador único do problema
id_cliente	NUMBER	10	FK	Identificador único do cliente
nm_problema	VARCHAR	100	NN	nome do problema
ds_problema	VARCHAR	200	NN	descrição do problema

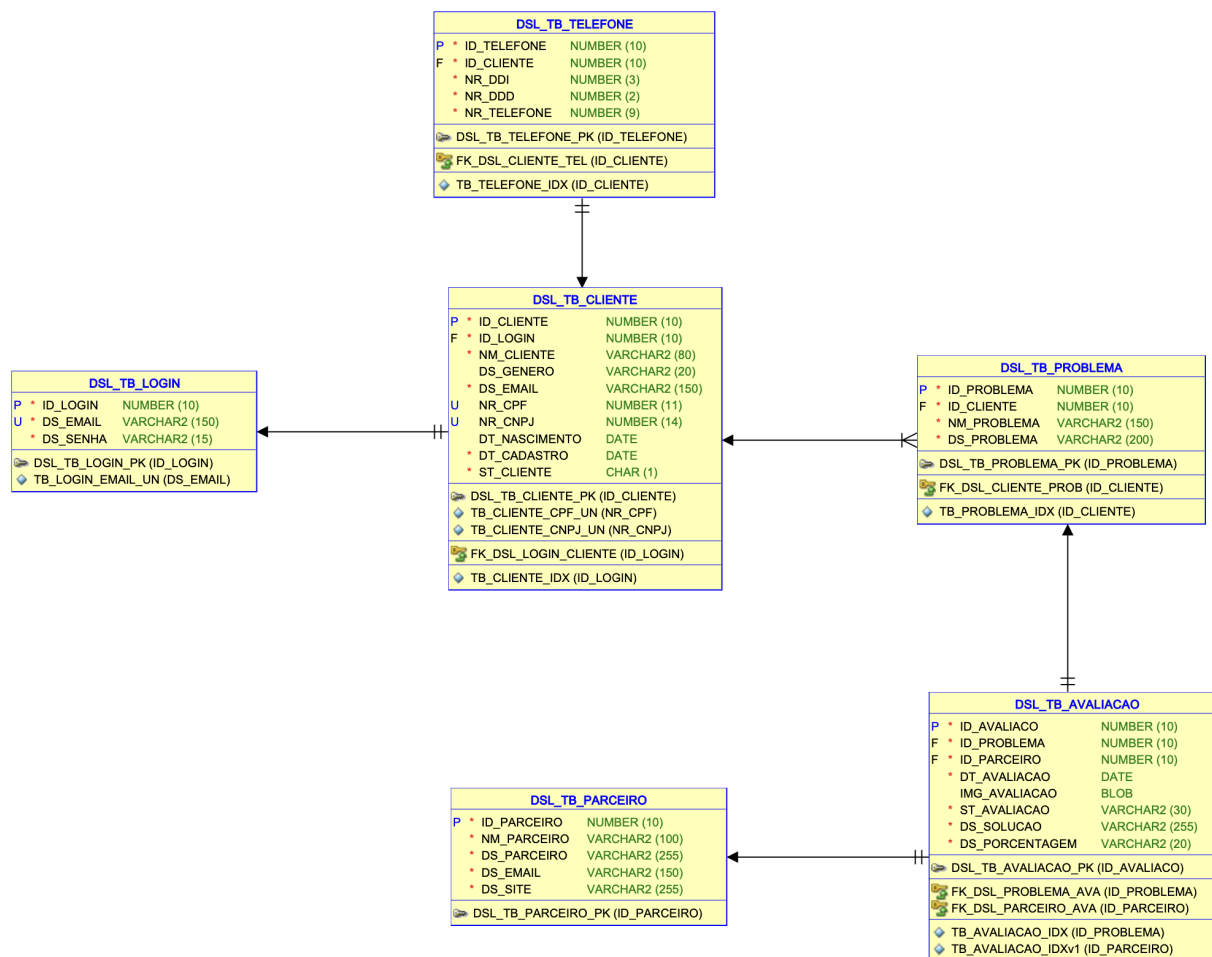
Tabela	TELEFONE			
Descrição	Tabela de Empresas. Relaciona-se com a tabela CLIENTE			
Coluna	Tipo de Dados	Tamanho	Constraint	Descrição
id telefone	NUMBER	10	PK	Identificador único do telefone
id cliente	NUMBER	10	FK	Identificador único do cliente
nr_ddi	NUMBER	3	NN	número do DDI do telefone
nr_ddd	NUMBER	2	NN	número do DDD do telefone
nr telefone	NUMBER	9	NN	número do telefone

Tabela	AVALIAÇÃO			
Descrição	Tabela de Empresas. Relaciona-se com a tabela PROBLEMA, PARCEIRO			
Coluna	Tipo de Dados	Tamanho	Constraint	Descrição
id avaliacao	NUMBER	10	PK	Identificador único da avaliação
id_problema	NUMBER	10	FK	Identificador único do problema
id_parceiro	NUMBER	10	FK	Identificador único do parceiro
dt avaliacao	Date	-	NN	data da avaliação
img avaliacao	BLOB	-	NN	imagem da avaliação
st avaliacao	VARCHAR	30	NN	status da avaliação
ds_solucão	VARCHAR	150	NN	descrição da avaliação
ds porcentagem	VARCHAR	20	NN	valor da porcentagem de precisão

3 – Projeto Lógico do Banco de Dados



4 – Projeto Físico do Banco de Dados



5 – Data Definition Language – DDL

```
DROP TABLE dsl_tb_avaliacao CASCADE CONSTRAINTS;
DROP TABLE dsl_tb_cliente CASCADE CONSTRAINTS;
DROP TABLE dsl_tb_login CASCADE CONSTRAINTS;
DROP TABLE dsl_tb_parceiro CASCADE CONSTRAINTS;
DROP TABLE dsl_tb_problema CASCADE CONSTRAINTS;
DROP TABLE dsl_tb_telefone CASCADE CONSTRAINTS;
DROP TABLE dsl_tb_erros CASCADE CONSTRAINTS;
```

```
DROP SEQUENCE seq_avaliacao;
DROP SEQUENCE seq_cliente;
DROP SEQUENCE seq_login;
DROP SEQUENCE seq_parceiro;
DROP SEQUENCE seq_problema;
DROP SEQUENCE seq_telefone;
DROP SEQUENCE seq_erros;
```

```
-- CREATE SEQUENCES
```

```
CREATE SEQUENCE seq_avaliacao
START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE seq_cliente
START WITH 1 INCREMENT BY 1;
```

```
CREATE SEQUENCE seq_login
START WITH 1 INCREMENT BY 1;
```

```
CREATE SEQUENCE seq_parceiro
START WITH 1 INCREMENT BY 1;
```

```
CREATE SEQUENCE seq_problema
START WITH 1 INCREMENT BY 1;
```

```
CREATE SEQUENCE seq_telefone
START WITH 1 INCREMENT BY 1;
```

```
CREATE SEQUENCE seq_erros
START WITH 1 INCREMENT BY 1;
```

```
-- CREATE TABLES

CREATE TABLE dsl_tb_avaliacao (
  id_avaliacao NUMBER(10) NOT NULL,
  id_problema NUMBER(10) NOT NULL,
  id_parceiro NUMBER(10) NOT NULL,
  dt_avaliacao DATE NOT NULL,
  img_avaliacao BLOB,
  st_avaliacao VARCHAR2(30) NOT NULL,
  ds_solucao VARCHAR2(255) NOT NULL,
  ds_porcentagem VARCHAR2(255) NOT NULL
);

CREATE UNIQUE INDEX tb_avaliacao_idx ON
dsl_tb_avaliacao (
  id_problema
ASC );

CREATE UNIQUE INDEX tb_avaliacao_idxv1 ON
dsl_tb_avaliacao (
  id_parceiro
ASC );

ALTER TABLE dsl_tb_avaliacao ADD CONSTRAINT dsl_tb_avaliacao_pk PRIMARY KEY (
  id_avaliacao );

CREATE TABLE dsl_tb_cliente (
  id_cliente NUMBER(10) NOT NULL,
  id_login NUMBER(10) NOT NULL,
  nm_cliente VARCHAR2(80) NOT NULL,
  ds_genero VARCHAR2(20),
  ds_email VARCHAR2(150) NOT NULL,
  nr_cpf NUMBER(11),
  nr_cnpj NUMBER(14),
  dt_nascimento DATE,
  dt_cadastro DATE NOT NULL,
  st_cliente CHAR(1) NOT NULL
);

CREATE UNIQUE INDEX tb_cliente_idx ON
dsl_tb_cliente (
  id_login
ASC );
```

```
ALTER TABLE dsl_tb_cliente ADD CONSTRAINT dsl_tb_cliente_pk PRIMARY KEY (
id_cliente );

ALTER TABLE dsl_tb_cliente ADD CONSTRAINT tb_cliente_cpf_un UNIQUE ( nr_cpf );

ALTER TABLE dsl_tb_cliente ADD CONSTRAINT tb_cliente_cnpj_un UNIQUE ( nr_cnpj );

CREATE TABLE dsl_tb_login (
id_login NUMBER(10) NOT NULL,
ds_email VARCHAR2(150) NOT NULL,
ds_senha VARCHAR2(15) NOT NULL
);

ALTER TABLE dsl_tb_login ADD CONSTRAINT dsl_tb_login_pk PRIMARY KEY ( id_login );

ALTER TABLE dsl_tb_login ADD CONSTRAINT tb_login_email_un UNIQUE ( ds_email );

CREATE TABLE dsl_tb_parceiro (
id_parceiro NUMBER(10) NOT NULL,
nm_parceiro VARCHAR2(100) NOT NULL,
ds_parceiro VARCHAR2(255) NOT NULL,
ds_email VARCHAR2(150) NOT NULL,
ds_site VARCHAR2(255) NOT NULL
);

ALTER TABLE dsl_tb_parceiro ADD CONSTRAINT dsl_tb_parceiro_pk PRIMARY KEY (
id_parceiro );

CREATE TABLE dsl_tb_problema (
id_problema NUMBER(10) NOT NULL,
id_cliente NUMBER(10) NOT NULL,
nm_problema VARCHAR2(150) NOT NULL,
ds_problema VARCHAR2(200) NOT NULL
);

CREATE INDEX tb_problema_idx ON
dsl_tb_problema (
id_cliente
ASC );

ALTER TABLE dsl_tb_problema ADD CONSTRAINT dsl_tb_problema_pk PRIMARY KEY (
id_problema );

CREATE TABLE dsl_tb_telefone (
id_telefone NUMBER(10) NOT NULL,
```

```

id_cliente NUMBER(10) NOT NULL,
nr_ddi NUMBER(3) NOT NULL,
nr_ddd NUMBER(2) NOT NULL,
nr_telefone NUMBER(9) NOT NULL
);

CREATE UNIQUE INDEX tb_telefone_idx ON
dsl_tb_telefone (
id_cliente
ASC );

ALTER TABLE dsl_tb_telefone ADD CONSTRAINT dsl_tb_telefone_pk PRIMARY KEY (
id_telefone );

-- Tabela de erros
CREATE TABLE dsl_tb_erros (
codigo_erro NUMBER(5) NOT NULL,
nome_erro VARCHAR2(255) NOT NULL,
data_ocorrencia DATE NOT NULL,
usuario_logado VARCHAR2(100) NOT NULL
);

ALTER TABLE dsl_tb_erros ADD CONSTRAINT dsl_t_erros_pk PRIMARY KEY ( codigo_erro
);

ALTER TABLE dsl_tb_problema
ADD CONSTRAINT fk_dsl_cliente_prob FOREIGN KEY ( id_cliente )
REFERENCES dsl_tb_cliente ( id_cliente );

ALTER TABLE dsl_tb_telefone
ADD CONSTRAINT fk_dsl_cliente_tel FOREIGN KEY ( id_cliente )
REFERENCES dsl_tb_cliente ( id_cliente );

ALTER TABLE dsl_tb_cliente
ADD CONSTRAINT fk_dsl_login_cliente FOREIGN KEY ( id_login )
REFERENCES dsl_tb_login ( id_login );

ALTER TABLE dsl_tb_avaliacao
ADD CONSTRAINT fk_dsl_parceiro_ava FOREIGN KEY ( id_parceiro )
REFERENCES dsl_tb_parceiro ( id_parceiro );

ALTER TABLE dsl_tb_avaliacao
ADD CONSTRAINT fk_dsl_problema_ava FOREIGN KEY ( id_problema )
REFERENCES dsl_tb_problema ( id_problema );

```


6 – Data Manipulation Language – DML (INSERT)

```
-- INSERT 1

INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'joao1@email.com', 'joao12345');

INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero,
ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Joao Felipe', 'Masculino',
'joao@email.com', 12345678901, NULL, TO_DATE('1990-01-01', 'YYYY-MM-DD'),
TO_DATE('2023-09-07', 'YYYY-MM-DD'), 'A');

INSERT INTO dsl_tb_telefone (id_telefone, id_cliente, nr_ddi, nr_ddd,
nr_telefone)
VALUES (seq_telefone.NEXTVAL, seq_cliente.CURRVAL, 1, 11, 123456789);

INSERT INTO dsl_tb_parceiro (id_parceiro, nm_parceiro, ds_parceiro,
ds_email, ds_site)
VALUES (seq_parceiro.NEXTVAL, 'Parceiro 1', 'Descrição Parceiro 1',
'parceiro1@email.com', 'www.parceiro1.com');

INSERT INTO dsl_tb_problema (id_problema, id_cliente, nm_problema,
ds_problema)
VALUES (seq_problema.NEXTVAL, seq_cliente.CURRVAL, 'Problema 1', 'Descrição
Problema 1');

INSERT INTO dsl_tb_avaliacao (id_avaliacao, id_problema, id_parceiro,
dt_avaliacao, img_avaliacao, st_avaliacao, ds_solucao, ds_porcentagem)
VALUES (seq_avaliacao.NEXTVAL, seq_problema.CURRVAL, seq_parceiro.CURRVAL,
TO_DATE('2023-09-07', 'YYYY-MM-DD'), NULL, 'Aprovado', 'Solução 1', '95%');

-- INSERT 2

INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'maria1@email.com', 'maria12345');

INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero,
ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
```

```
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Maria Cardoso',
'Feminino', 'maria@email.com', NULL, 98765432101, TO_DATE('1985-05-05',
'YYYY-MM-DD'), TO_DATE('2023-09-08', 'YYYY-MM-DD'), 'A');

INSERT INTO dsl_tb_telefone (id_telefone, id_cliente, nr_ddi, nr_ddd,
nr_telefone)
VALUES (seq_telefone.NEXTVAL, seq_cliente.CURRVAL, 1, 22, 987654321);

INSERT INTO dsl_tb_parceiro (id_parceiro, nm_parceiro, ds_parceiro,
ds_email, ds_site)
VALUES (seq_parceiro.NEXTVAL, 'Parceiro 2', 'Descrição Parceiro 2',
'parceiro2@email.com', 'www.parceiro2.com');

INSERT INTO dsl_tb_problema (id_problema, id_cliente, nm_problema,
ds_problema)
VALUES (seq_problema.NEXTVAL, seq_cliente.CURRVAL, 'Problema 2', 'Descrição
Problema 2');

INSERT INTO dsl_tb_avaliacao (id_avaliacao, id_problema, id_parceiro,
dt_avaliacao, img_avaliacao, st_avaliacao, ds_solucao, ds_porcentagem)
VALUES (seq_avaliacao.NEXTVAL, seq_problema.CURRVAL, seq_parceiro.CURRVAL,
TO_DATE('2023-09-08', 'YYYY-MM-DD'), NULL, 'Reprovado', 'Solução 2', '90%');

-- INSERT 3

INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'wagner1@email.com', 'wagner12345');

INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero,
ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Wagner Brito',
'Masculino', 'wagner@email.com', 33333333333, NULL, TO_DATE('2000-12-15',
'YYYY-MM-DD'), TO_DATE('2023-09-09', 'YYYY-MM-DD'), 'I');

INSERT INTO dsl_tb_telefone (id_telefone, id_cliente, nr_ddi, nr_ddd,
nr_telefone)
VALUES (seq_telefone.NEXTVAL, seq_cliente.CURRVAL, 1, 33, 333333333);

INSERT INTO dsl_tb_parceiro (id_parceiro, nm_parceiro, ds_parceiro,
ds_email, ds_site)
VALUES (seq_parceiro.NEXTVAL, 'Parceiro 3', 'Descrição Parceiro 3',
'parceiro3@email.com', 'www.parceiro3.com');
```

```
INSERT INTO dsl_tb_problema (id_problema, id_cliente, nm_problema,
ds_problema)
VALUES (seq_problema.NEXTVAL, seq_cliente.CURRVAL, 'Problema 3', 'Descrição
Problema 3');

INSERT INTO dsl_tb_avaliacao (id_avaliacao, id_problema, id_parceiro,
dt_avaliacao, img_avaliacao, st_avaliacao, ds_solucao, ds_porcentagem)
VALUES (seq_avaliacao.NEXTVAL, seq_problema.CURRVAL, seq_parceiro.CURRVAL,
TO_DATE('2023-09-09', 'YYYY-MM-DD'), NULL, 'Pendente', 'Solução 3', '85%');

-- INSERT 4

INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'laura1@email.com', 'laura12345');

INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero,
ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Laura Ribeiro',
'Feminino', 'laura@email.com', NULL, 44444444444444, TO_DATE('1999-05-17',
'YYYY-MM-DD'), TO_DATE('2023-09-10', 'YYYY-MM-DD'), 'A');

INSERT INTO dsl_tb_telefone (id_telefone, id_cliente, nr_ddi, nr_ddd,
nr_telefone)
VALUES (seq_telefone.NEXTVAL, seq_cliente.CURRVAL, 1, 44, 4444444444);

INSERT INTO dsl_tb_parceiro (id_parceiro, nm_parceiro, ds_parceiro,
ds_email, ds_site)
VALUES (seq_parceiro.NEXTVAL, 'Parceiro 4', 'Descrição Parceiro 4',
'parceiro4@email.com', 'www.parceiro4.com');

INSERT INTO dsl_tb_problema (id_problema, id_cliente, nm_problema,
ds_problema)
VALUES (seq_problema.NEXTVAL, seq_cliente.CURRVAL, 'Problema 4', 'Descrição
Problema 4');

INSERT INTO dsl_tb_avaliacao (id_avaliacao, id_problema, id_parceiro,
dt_avaliacao, img_avaliacao, st_avaliacao, ds_solucao, ds_porcentagem)
VALUES (seq_avaliacao.NEXTVAL, seq_problema.CURRVAL, seq_parceiro.CURRVAL,
TO_DATE('2023-09-10', 'YYYY-MM-DD'), NULL, 'Aprovado', 'Solução 4', '75%');

-- INSERT 5

INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'antonio1@email.com', 'antonio12345');
```

```
INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero,
ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Antonio Silva', 'Outro',
'antonio@email.com', 55555555555, NULL, TO_DATE('1998-03-20', 'YYYY-MM-DD'),
TO_DATE('2023-09-11', 'YYYY-MM-DD'), 'A');

INSERT INTO dsl_tb_telefone (id_telefone, id_cliente, nr_ddi, nr_ddd,
nr_telefone)
VALUES (seq_telefone.NEXTVAL, seq_cliente.CURRVAL, 1, 55, 5555555555);

INSERT INTO dsl_tb_parceiro (id_parceiro, nm_parceiro, ds_parceiro,
ds_email, ds_site)
VALUES (seq_parceiro.NEXTVAL, 'Parceiro 5', 'Descrição Parceiro 5',
'parceiro5@email.com', 'www.parceiro5.com');

INSERT INTO dsl_tb_problema (id_problema, id_cliente, nm_problema,
ds_problema)
VALUES (seq_problema.NEXTVAL, seq_cliente.CURRVAL, 'Problema 5', 'Descrição
Problema 5');

INSERT INTO dsl_tb_avaliacao (id_avaliacao, id_problema, id_parceiro,
dt_avaliacao, img_avaliacao, st_avaliacao, ds_solucao, ds_porcentagem)
VALUES (seq_avaliacao.NEXTVAL, seq_problema.CURRVAL, seq_parceiro.CURRVAL,
TO_DATE('2023-09-11', 'YYYY-MM-DD'), NULL, 'Reprovado', 'Solução 5', '88%');
```

7 – Procedures

```
-- Procedimento 1: Atualizar o status de avaliação
DROP PROCEDURE atualizar_status_avaliacao;

CREATE OR REPLACE PROCEDURE atualizar_status_avaliacao(
p_id_avaliacao IN NUMBER,
p_novo_status IN VARCHAR2
) AS
v_cd_erro NUMBER := SQLCODE;
v_ds_mensagem VARCHAR2(255) := SQLERRM;
BEGIN
BEGIN
UPDATE dsl_tb_avaliacao
SET st_avaliacao = p_novo_status
WHERE id_avaliacao = p_id_avaliacao;

COMMIT;
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

WHEN OTHERS THEN

```
INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia,
usuario_logado)
VALUES (v_cd_erro, v_ds_mensagem, TO_CHAR(SYSDATE, 'DD-MON-YYYY HH24:MI:SS'),
USER);
DBMS_OUTPUT.PUT_LINE('Erro ao atualizar o status da avaliação.');
```

END;

END;

/

-- EXECUTAR PROCEDURE

```
BEGIN
atualizar_status_avaliacao(3, 'Aprovada');
END;
```

SELECT * FROM dsl_tb_avaliacao;

--

-- PROCEDURE 2

DROP PROCEDURE calcular_idade_cliente;

```

CREATE OR REPLACE PROCEDURE calcular_idade_cliente(
p_id_cliente IN NUMBER,
p_idade OUT NUMBER
) AS
v_data_nascimento DATE;
v_cd_erro NUMBER := SQLCODE;
v_ds_mensagem VARCHAR2(255) := SQLERRM;
BEGIN
BEGIN
SELECT dt_nascimento INTO v_data_nascimento
FROM dsl_tb_cliente
WHERE id_cliente = p_id_cliente;
-- Calcular a idade do cliente
SELECT EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM v_data_nascimento) INTO
p_idade FROM DUAL;
-- Retornar a idade
DBMS_OUTPUT.PUT_LINE('A idade do cliente é ' || p_idade);
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

```

WHEN OTHERS THEN
INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrendia,
usuario_logado)
VALUES (v_cd_erro, v_ds_mensagem, TO_CHAR(SYSDATE, 'DD-MON-YYYY HH24:MI:SS'),
USER);
DBMS_OUTPUT.PUT_LINE('Erro ao calcular a idade do cliente.');
```

```

END;
END;
/

---
-- EXEMPLO EXECUCAO DE PROCEDURE
---
```

```

SET SERVEROUTPUT ON

DECLARE
v_idade NUMBER;
BEGIN
calcular_idade_cliente(4, v_idade);
DBMS_OUTPUT.PUT_LINE('A idade do cliente é ' || v_idade);
END;
/
```

```
SELECT * FROM dsl_tb_cliente;
```

8 – Funções e Trigger

```
--
-- Funcao 1
--

SET SERVEROUTPUT ON
CREATE OR REPLACE FUNCTION calcular_total_avaliacoes_por_parceiro(
p_id_parceiro IN NUMBER
) RETURN NUMBER IS
v_total_avaliacoes NUMBER;
BEGIN
BEGIN
SELECT COUNT(*) INTO v_total_avaliacoes
FROM dsl_tb_avaliacao
WHERE id_parceiro = p_id_parceiro;
EXCEPTION
WHEN NO_DATA_FOUND THEN
DBMS_OUTPUT.PUT_LINE('Nenhuma avaliação encontrada para o parceiro.');
```

WHEN OTHERS THEN
DBMS_OUTPUT.PUT_LINE('Erro ao calcular o total de avaliações.');

END;

RETURN v_total_avaliacoes;
END;

/

SET SERVEROUTPUT ON
DECLARE
v_id_parceiro NUMBER := 1000;
v_total_avaliacoes NUMBER;
BEGIN
v_total_avaliacoes := calcular_total_avaliacoes_por_parceiro(v_id_parceiro);
DBMS_OUTPUT.PUT_LINE('Total de avaliações para o parceiro ' || v_id_parceiro ||
' : ' || v_total_avaliacoes || ' avaliacao ');
END;

/

--
-- Funcao 2
--

CREATE OR REPLACE FUNCTION obter_informacoes_cliente(
p_id_cliente IN NUMBER


```

) RETURN dsl_tb_cliente%ROWTYPE
IS
v_cliente_info dsl_tb_cliente%ROWTYPE;
v_data_nascimento DATE;
v_cd_erro NUMBER := SQLCODE;
v_ds_mensagem VARCHAR2(255) := SQLERRM;
BEGIN
BEGIN
SELECT *
INTO v_cliente_info
FROM dsl_tb_cliente
WHERE id_cliente = p_id_cliente;
EXCEPTION
WHEN NO_DATA_FOUND THEN
v_cd_erro := SQLCODE;
v_ds_mensagem := SQLERRM;
DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

codigo_erro	nome_erro	data_ocorrencia	usuario_logado
v_cd_erro	v_ds_mensagem	SYSDATE	USER

```

INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia,
usuario_logado)
VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER);
RETURN NULL;
WHEN OTHERS THEN
v_cd_erro := SQLCODE;
v_ds_mensagem := SQLERRM;
DBMS_OUTPUT.PUT_LINE('Erro ao encontrar Cliente.');
```

codigo_erro	nome_erro	data_ocorrencia	usuario_logado
v_cd_erro	v_ds_mensagem	SYSDATE	USER

```

INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia,
usuario_logado)
VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER);
RETURN NULL;
END;

RETURN v_cliente_info;
END;
/

--
-- TRIGGER
--

CREATE OR REPLACE TRIGGER trg_validacao_idade
BEFORE INSERT OR UPDATE ON dsl_tb_cliente
FOR EACH ROW
BEGIN
```

```
DECLARE
v_idade NUMBER;
BEGIN
v_idade := TRUNC(MONTHS_BETWEEN(SYSDATE, :new.dt_nascimento) / 12);

IF v_idade < 18 THEN
RAISE_APPLICATION_ERROR(-20001, 'A idade do cliente deve ser maior que 18
anos.');
```

```
END IF;
END;
END;
/

INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'joaozinho@email.com', 'joaozinho12345');
```

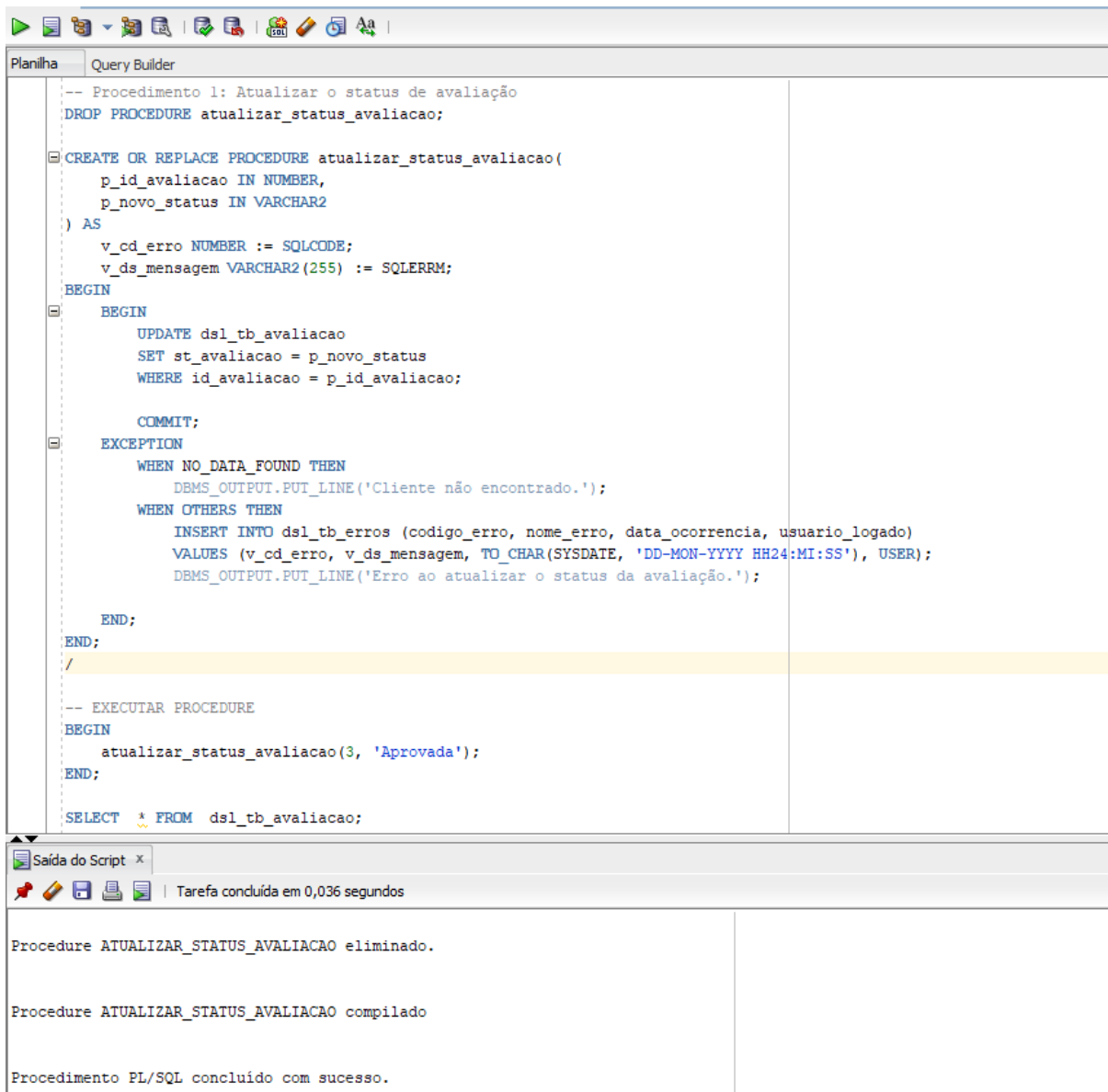
```
INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero,
ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Joãozinho', 'Masculino',
'joaozinho@email.com', 12345678902, NULL, TO_DATE('2004-01-01', 'YYYY-MM-DD'),
TO_DATE('2023-09-12', 'YYYY-MM-DD'), 'A');
```

```
UPDATE dsl_tb_cliente
SET dt_nascimento = TO_DATE('2003-01-01', 'YYYY-MM-DD')
WHERE id_cliente = 10;

SELECT * FROM dsl_tb_cliente;

SELECT * FROM dsl_tb_login;
```

[illegible]



The screenshot displays the Oracle SQL Developer environment. The top toolbar includes icons for running, saving, and editing scripts. The main window is titled "Query Builder" and contains a PL/SQL script for updating evaluation status. The script includes a comment, a DROP statement, a CREATE OR REPLACE PROCEDURE block with an UPDATE, an EXCEPTION block for error handling, and a final SELECT statement. The bottom window, titled "Saída do Script", shows the execution results, indicating that the procedure was successfully compiled and executed.

```
-- Procedimento 1: Atualizar o status de avaliação
DROP PROCEDURE atualizar_status_avaliacao;

CREATE OR REPLACE PROCEDURE atualizar_status_avaliacao(
    p_id_avaliacao IN NUMBER,
    p_novo_status IN VARCHAR2
) AS
    v_cd_erro NUMBER := SQLCODE;
    v_ds_mensagem VARCHAR2(255) := SQLERRM;
BEGIN
    BEGIN
        UPDATE dsl_tb_avaliacao
        SET st_avaliacao = p_novo_status
        WHERE id_avaliacao = p_id_avaliacao;

        COMMIT;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

```
        WHEN OTHERS THEN
            INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia, usuario_logado)
            VALUES (v_cd_erro, v_ds_mensagem, TO_CHAR(SYSDATE, 'DD-MON-YYYY HH24:MI:SS'), USER);
            DBMS_OUTPUT.PUT_LINE('Erro ao atualizar o status da avaliação.');
```

```
    END;
END;
/

-- EXECUTAR PROCEDURE
BEGIN
    atualizar_status_avaliacao(3, 'Aprovada');
END;

SELECT * FROM dsl_tb_avaliacao;
```

Saída do Script x

Tarefa concluída em 0,036 segundos

Procedure ATUALIZAR_STATUS_AVALIACAO eliminado.

Procedure ATUALIZAR_STATUS_AVALIACAO compilado

Procedimento PL/SQL concluído com sucesso.

Planilha Query Builder

```
-- Procedimento 1: Atualizar o status de avaliação
DROP PROCEDURE atualizar_status_avaliacao;

CREATE OR REPLACE PROCEDURE atualizar_status_avaliacao(
  p_id_avaliacao IN NUMBER,
  p_novo_status IN VARCHAR2
) AS
  v_cd_erro NUMBER := SQLCODE;
  v_ds_mensagem VARCHAR2(255) := SQLERRM;
BEGIN
  BEGIN
    UPDATE dsl_tb_avaliacao
    SET st_avaliacao = p_novo_status
    WHERE id_avaliacao = p_id_avaliacao;

    COMMIT;
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

```
    WHEN OTHERS THEN
      INSERT INTO dsl_tb_erro (codigo_erro, nome_erro, data_ocorrencia, usuario_logado)
      VALUES (v_cd_erro, v_ds_mensagem, TO_CHAR(SYSDATE, 'DD-MON-YYYY HH24:MI:SS'), USER);
      DBMS_OUTPUT.PUT_LINE('Erro ao atualizar o status da avaliação.');
```

```
  END;
END;
/

-- EXECUTAR PROCEDURE
BEGIN
  atualizar_status_avaliacao(3, 'Aprovada');
END;
```

```
SELECT * FROM dsl_tb_avaliacao;
```

Salida do Script x Resultado da Consulta x

Todas as Linhas Extraídas: 5 em 0,058 segundos

ID_AVALIACAO	ID_PROBLEMA	ID_PARCEIRO	DT_AVALIACAO	IMG_AVALIACAO	ST_AVALIACAO	DS_SOLUCAO	DS_PORCENTAGEM
1	1	1	107/09/23	(null)	Aprovado	Solução 1	95%
2	2	2	208/09/23	(null)	Reprovado	Solução 2	90%
3	3	3	309/09/23	(null)	Aprovada	Solução 3	85%
4	4	4	410/09/23	(null)	Aprovado	Solução 4	75%
5	5	5	511/09/23	(null)	Reprovado	Solução 5	88%

Planilha Query Builder

```
-- PROCEDURE 2

DROP PROCEDURE calcular_idade_cliente;

CREATE OR REPLACE PROCEDURE calcular_idade_cliente(
  p_id_cliente IN NUMBER,
  p_idade OUT NUMBER
) AS
  v_data_nascimento DATE;
  v_cd_erro NUMBER := SQLCODE;
  v_ds_mensagem VARCHAR2(255) := SQLERRM;
BEGIN
  BEGIN
    -- Obter a data de nascimento do cliente
    SELECT dt_nascimento INTO v_data_nascimento
    FROM dsl_tb_cliente
    WHERE id_cliente = p_id_cliente;
```

```
    -- Calcular a idade do cliente
    SELECT EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM v_data_nascimento) INTO p_idade FROM DUAL;
```

```
    -- Retornar a idade
    DBMS_OUTPUT.PUT_LINE('A idade do cliente é ' || p_idade);
```

```
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
      DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

```
    WHEN OTHERS THEN
      INSERT INTO dsl_tb_erro (codigo_erro, nome_erro, data_ocorrencia, usuario_logado)
      VALUES (v_cd_erro, v_ds_mensagem, TO_CHAR(SYSDATE, 'DD-MON-YYYY HH24:MI:SS'), USER);
      DBMS_OUTPUT.PUT_LINE('Erro ao calcular a idade do cliente.');
```

```
  END;
END;
```

Salida do Script x Resultado da Consulta x

Tarefa concluída em 0,063 segundos

Procedure CALCULAR_IDADE_CLIENTE eliminado.

Procedure CALCULAR_IDADE_CLIENTE compilado






```
--
-- EXEMPLO EXECUCAO DE PROCEDURE
--

SET SERVEROUTPUT ON

DECLARE
    v_idade NUMBER;
BEGIN
    calcular_idade_cliente(4, v_idade); -- Substitua 1 pelo ID do cliente desejado
    DBMS_OUTPUT.PUT_LINE('A idade do cliente é ' || v_idade);
END;

/

SELECT * FROM dsl.tb_cliente;
```






 | Tarefa concluída em 0,555 segundos

Procedure CALCULAR IDADE CLIENTE eliminado.

Procedure CALCULAR IDADE CLIENTE compilado

A idade do cliente é 24

A idade do cliente é 24

Procedimento PL/SQL concluído com sucesso.

```



Planilha Query Builder
-- Funcao 1
--
SET SERVEROUTPUT ON
CREATE OR REPLACE FUNCTION calcular_total_avaliacoes_por_parceiro(
    p_id_parceiro IN NUMBER
) RETURN NUMBER IS
    v_total_avaliacoes NUMBER;
BEGIN
    BEGIN
        SELECT COUNT(*) INTO v_total_avaliacoes
        FROM dsl_tb_avaliacao
        WHERE id_parceiro = p_id_parceiro;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Nenhuma avaliação encontrada para o parceiro. ');
        WHEN OTHERS THEN
            DBMS_OUTPUT.PUT_LINE('Erro ao calcular o total de avaliações. ');
    END;

    RETURN v_total_avaliacoes;
END;
/

SET SERVEROUTPUT ON
DECLARE
    v_id_parceiro NUMBER := 1;
    v_total_avaliacoes NUMBER;
BEGIN
    v_total_avaliacoes := calcular_total_avaliacoes_por_parceiro(v_id_parceiro);

    DBMS_OUTPUT.PUT_LINE('Total de avaliações para o parceiro ' || v_id_parceiro || ': ' || v_total_avaliacoes || ' avaliacao ');
END;
/

```

 Saída do Script x
  Resultado da Consulta x

Function CALCULAR TOTAL AVALIACOES POR PARCEIRO compilado

Total de avaliações para o parceiro 1: 1 avaliacao

Procedimento PL/SQL concluído com sucesso.

Planilha	
Query Builder	
<pre>-- Funcao 2 -- CREATE OR REPLACE FUNCTION obter_informacoes_cliente(p_id_cliente IN NUMBER) RETURN dsl_tb_cliente%ROWTYPE IS v_cliente_info dsl_tb_cliente%ROWTYPE; v_data_nascimento DATE; v_cd_erro NUMBER := SQLCODE; v_ds_mensagem VARCHAR2(255) := SQLERRM; BEGIN BEGIN SELECT * INTO v_cliente_info FROM dsl_tb_cliente WHERE id_cliente = p_id_cliente; EXCEPTION WHEN NO_DATA_FOUND THEN v_cd_erro := SQLCODE; v_ds_mensagem := SQLERRM; DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');</pre>	
<pre> INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrendia, usuario_logado) VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER); RETURN NULL; WHEN OTHERS THEN v_cd_erro := SQLCODE; v_ds_mensagem := SQLERRM; DBMS_OUTPUT.PUT_LINE('Erro ao encontrar Cliente.');</pre>	
<pre> INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrendia, usuario_logado) VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER); RETURN NULL; END; RETURN v_cliente_info; END; /</pre>	
<div> <div> Saída do Script x Resultado da Consulta x </div> <div> Tarefa concluída em 0,088 segundos </div> </div>	
Function OBTHER_INFORMACOES_CLIENTE compilado	

Planilha	Query Builder
<pre> WHEN NO_DATA_FOUND THEN v_cd_erro := SQLCODE; v_ds_mensagem := SQLERRM; DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');</pre>	
<pre> INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia, usuario_logado) VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER); RETURN NULL; WHEN OTHERS THEN v_cd_erro := SQLCODE; v_ds_mensagem := SQLERRM; DBMS_OUTPUT.PUT_LINE('Erro ao encontrar Cliente.');</pre>	
<pre> INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia, usuario_logado) VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER); RETURN NULL; END; RETURN v_cliente_info; END; / SET SERVEROUTPUT ON</pre>	
<pre> DECLARE v_cliente_info dsl_tb_cliente%ROWTYPE; BEGIN v_cliente_info := obter_informacoes_cliente(1); DBMS_OUTPUT.PUT_LINE('Nome do Cliente: ' v_cliente_info.nm_cliente); DBMS_OUTPUT.PUT_LINE('Email do Cliente: ' v_cliente_info.ds_email); END; /</pre>	
<div> <div> Saída do Script x Resultado da Consulta x </div> <div> Tarefa concluída em 0,078 segundos </div> </div>	
<pre> Function OBTER_INFORMACOES_CLIENTE compilado Nome do Cliente: Joao Felipe Email do Cliente: joao@email.com Procedimento PL/SQL concluído com sucesso.</pre>	


```
WHEN NO_DATA_FOUND THEN
    v_cd_erro := SQLCODE;
    v_ds_mensagem := SQLERRM;
    DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

```
INSERT INTO dsl_tb_erro (codigo_erro, nome_erro, data_ocorrencia, usuario_logado)
VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER);
RETURN NULL;

WHEN OTHERS THEN
    v_cd_erro := SQLCODE;
    v_ds_mensagem := SQLERRM;
    DBMS_OUTPUT.PUT_LINE('Erro ao encontrar Cliente.');
```

```
INSERT INTO dsl_tb_erro (codigo_erro, nome_erro, data_ocorrencia, usuario_logado)
VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER);
RETURN NULL;

END;

RETURN v_cliente_info;
END;
/

SET SERVEROUTPUT ON

DECLARE
    v_cliente_info dsl_tb_cliente%ROWTYPE;
BEGIN
    v_cliente_info := obter_informacoes_cliente(100);
    DBMS_OUTPUT.PUT_LINE('Nome do Cliente: ' || v_cliente_info.nm_cliente);
    DBMS_OUTPUT.PUT_LINE('Email do Cliente: ' || v_cliente_info.ds_email);
END;
/

SELECT * FROM DSL_TB_ERROS
```

Saída do Script x Resultado da Consulta x

Tarefa concluída em 0,083 segundos

Cliente não encontrado.
Nome do Cliente:
Email do Cliente:

Procedimento PL/SQL concluído com sucesso.

Planilha

Query Builder

```
v_cd_erro := SQLCODE;
v_ds_mensagem := SQLERRM;
DBMS_OUTPUT.PUT_LINE('Cliente não encontrado.');
```

INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia, usuario_logado)

VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER);

RETURN NULL;

WHEN OTHERS THEN

v_cd_erro := SQLCODE;

v_ds_mensagem := SQLERRM;

DBMS_OUTPUT.PUT_LINE('Erro ao encontrar Cliente.');

INSERT INTO dsl_tb_erros (codigo_erro, nome_erro, data_ocorrencia, usuario_logado)

VALUES (v_cd_erro, v_ds_mensagem, SYSDATE, USER);

RETURN NULL;

END;

RETURN v_cliente_info;

END;

/

SET SERVEROUTPUT ON

DECLARE

v_cliente_info dsl_tb_cliente%ROWTYPE;

BEGIN

v_cliente_info := obter_informacoes_cliente(100);

DBMS_OUTPUT.PUT_LINE('Nome do Cliente: ' || v_cliente_info.nm_cliente);

DBMS_OUTPUT.PUT_LINE('Email do Cliente: ' || v_cliente_info.ds_email);

END;

/

SELECT * FROM DSL_TB_ERROS

Saída do Script x

Resultado da Consulta x

SQL

Todas as Linhas Extraídas: 1 em 0,02 segundos

	CODIGO_ERRO	NOME_ERRO	DATA_OCORRENCIA	USUARIO_LOGADO
1	100	ORA-01403: dados não encontrados	10/09/23	RM95324

Planilha

Query Builder

```
CREATE OR REPLACE TRIGGER trg_validacao_idade
BEFORE INSERT OR UPDATE ON dsl_tb_cliente
FOR EACH ROW
BEGIN
    DECLARE
        v_idade NUMBER;
    BEGIN
        v_idade := TRUNC(MONTHS_BETWEEN(SYSDATE, :new.dt_nascimento) / 12);

        IF v_idade < 18 THEN
            RAISE_APPLICATION_ERROR(-20001, 'A idade do cliente deve ser maior que 18 anos.');
```

END IF;

END;

/

INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)

VALUES (seq_login.NEXTVAL, 'joaozinho@email.com', 'joaozinho12345');

INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero, ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)

VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Joãozinho', 'Masculino', 'joaozinho@email.com', 12345678902, NULL, TO_DATE('2004-01-01', 'YYYY-MM-DD'), TO_DATE('2023-09-12', 'YYYY-MM-DD'), 'A');

UPDATE dsl_tb_cliente

SET dt_nascimento = TO_DATE('2003-01-01', 'YYYY-MM-DD')

WHERE id_cliente = 10;

SELECT * FROM dsl_tb_cliente;

SELECT * FROM dsl_tb_login;

Saída do Script x

Tarefa concluída em 0,084 segundos

Trigger TRG_VALIDACAO_IDADE compilado

<pre> CREATE OR REPLACE TRIGGER trg_validacao_idade BEFORE INSERT OR UPDATE ON dsl_tb_cliente FOR EACH ROW BEGIN DECLARE v_idade NUMBER; BEGIN v_idade := TRUNC(MONTHS_BETWEEN(SYSDATE, :new.dt_nascimento) / 12); IF v_idade < 18 THEN RAISE_APPLICATION_ERROR(-20001, 'A idade do cliente deve ser maior que 18 anos.');</pre>	<pre> END IF; END; END; / INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha) VALUES (seq_login.NEXTVAL, 'joaozinho@email.com', 'joaozinho12345'); INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero, ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente) VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Joãozinho', 'Masculino', 'joaozinho@email.com', 12345678902, NULL, TO_DATE('2007-01-01', 'YYYY-MM-DD'), TO_DATE('2023-09-12', 'YYYY-MM-DD'), 'A');</pre>
<pre> UPDATE dsl_tb_cliente SET dt_nascimento = TO_DATE('2003-01-01', 'YYYY-MM-DD') WHERE id_cliente = 10; SELECT * FROM dsl_tb_cliente; SELECT * FROM dsl_tb_login;</pre>	<p> Saída do Script x Tarefa concluída em 0,098 segundos Erros a partir da linha : 20 no comando - INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero, ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente) VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Joãozinho', 'Masculino', 'joaozinho@email.com', 12345678902, NULL, TO_DATE('2007-01-01', 'YYYY-MM-DD'), TO_DATE('2023-09-12', 'YYYY-MM-DD'), 'A') Relatório de erros - ORA-20001: A idade do cliente deve ser maior que 18 anos. ORA-06512: em "RM95324.TRG_VALIDACAO_IDADE", line 8 ORA-04088: erro durante a execução do gatilho 'RM95324.TRG_VALIDACAO_IDADE' </p>

```
CREATE OR REPLACE TRIGGER trg_validacao_idade
BEFORE INSERT OR UPDATE ON dsl_tb_cliente
FOR EACH ROW
BEGIN
    DECLARE
        v_idade NUMBER;
    BEGIN
        v_idade := TRUNC(MONTHS_BETWEEN(SYSDATE, :new.dt_nascimento) / 12);

        IF v_idade < 18 THEN
            RAISE_APPLICATION_ERROR(-20001, 'A idade do cliente deve ser maior que 18 anos.');
```

```
END IF;
    END;
END;
```

```
INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'joaozinho@email.com', 'joaozinho12345');
```

```
INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero, ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Joãozinho', 'Masculino', 'joaozinho@email.com', 12345678902, NULL, TO_DATE('2003-01-01', 'YYYY-MM-DD'), TO_DATE('2023-09-12', 'YYYY-MM-DD'), 'A');
```

```
UPDATE dsl_tb_cliente
SET dt_nascimento = TO_DATE('2003-01-01', 'YYYY-MM-DD')
WHERE id_cliente = 10;
```

```
SELECT * FROM dsl_tb_cliente;
```

```
SELECT * FROM dsl_tb_login;
```

Saída do Script
Tarefa concluída em 0,039 segundos

1 linha inserido.

```
CREATE OR REPLACE TRIGGER trg_validacao_idade
BEFORE INSERT OR UPDATE ON dsl_tb_cliente
FOR EACH ROW
BEGIN
    DECLARE
        v_idade NUMBER;
    BEGIN
        v_idade := TRUNC(MONTHS_BETWEEN(SYSDATE, :new.dt_nascimento) / 12);

        IF v_idade < 18 THEN
            RAISE_APPLICATION_ERROR(-20001, 'A idade do cliente deve ser maior que 18 anos.');
```

```
END IF;
    END;
END;
```

```
INSERT INTO dsl_tb_login (id_login, ds_email, ds_senha)
VALUES (seq_login.NEXTVAL, 'joaozinho@email.com', 'joaozinho12345');
```

```
INSERT INTO dsl_tb_cliente (id_cliente, id_login, nm_cliente, ds_genero, ds_email, nr_cpf, nr_cnpj, dt_nascimento, dt_cadastro, st_cliente)
VALUES (seq_cliente.NEXTVAL, seq_login.CURRVAL, 'Joãozinho', 'Masculino', 'joaozinho@email.com', 12345678902, NULL, TO_DATE('2003-01-01', 'YYYY-MM-DD'), TO_DATE('2023-09-12', 'YYYY-MM-DD'), 'A');
```

```
UPDATE dsl_tb_cliente
SET dt_nascimento = TO_DATE('2003-01-01', 'YYYY-MM-DD')
WHERE id_cliente = 10;
```

```
SELECT * FROM dsl_tb_cliente;
```

```
SELECT * FROM dsl_tb_login;
```

Saída do Script
Resultado da Consulta
Todas as Linhas Extraídas: 6 em 0,018 segundos

ID_CLIENTE	ID_LOGIN	NM_CLIENTE	DS_GENERO	DS_EMAIL	NR_CPF	NR_CNPJ	DT_NASCIMENTO	DT_CADASTRO	ST_CLIENTE
1	1	Joao Felipe	Masculino	joao@email.com	12345678901	(null)	01/01/90	07/09/23	A
2	2	Maria Cardoso	Feminino	maria@email.com	(null)	98765432101	05/05/85	08/09/23	A
3	3	Wagner Brito	Masculino	wagner@email.com	33333333333	(null)	15/12/00	09/09/23	I
4	4	Laura Ribeiro	Feminino	laura@email.com	(null)	444444444444444	17/05/99	10/09/23	A
5	5	Antonio Silva Outro	(null)	antonio@email.com	55555555555	(null)	20/03/98	11/09/23	A
6	7	Joãozinho	Masculino	joaozinho@email.com	12345678902	(null)	01/01/03	12/09/23	A