

Sistema de gestão de uma loja online de livros – **KnOWLedge**



**Docente da disciplina: Miguel Boavida**

**Martinha Piteira**

**Relatório realizado por:**

**Ana Sequeira Nº120221055 email: ana\_acoss@hotmail.com**

**André Benevides Nº120221013 email: andre\_fcpstb@hotmail.com**

**Índice**

[**Objetivo** 3](#_Toc378108206)

[**Análise de requisitos** 4](#_Toc378108207)

[**Detalhe das entidades** 5](#_Toc378108208)

[**20 questões e respostas para o sistema** 7](#_Toc378108209)

[**Procedimentos relativos às questões:** 7](#_Toc378108210)

[**Modelo Entidade-Relação** 9](#_Toc378108211)

[**Descrição do Modelo Entidade-Relação** 10](#_Toc378108212)

[**Pacotes, Procedimentos e Funções** 12](#_Toc378108213)

[**Pacotes relativos às tabelas:** 12](#_Toc378108214)

[**Procedimentos relativos à listagem de tabelas:** 12](#_Toc378108215)

[**Procedimentos relativos à inserção de dados em tabelas:** 12](#_Toc378108216)

[**Procedimentos relativos à atualização de dados em tabelas:** 12](#_Toc378108217)

[**Procedimentos relativos à remoção de dados em tabelas:** 12](#_Toc378108218)

[**Views** 13](#_Toc378108219)

[**Anexos** 14](#_Toc378108220)

# **Objetivo**

O sistema tem como objetivo gerir a informação associada as vendas, aos clientes e aos livros de uma loja online de livros.

# **Análise de requisitos**

Cada livro pode ter um ou mais géneros, um ou mais autores, uma editora, um idioma e um preço.

Os livros disponíveis na loja deverão ser visíveis para qualquer visitante.

Os clientes necessitam de uma conta para poderem fazer log-in no website da loja.

Para efetuar compras é necessário que o cliente tenha o log-in efetuado.

O inventário disponível deve estar catalogado por género (desporto, beleza, cientifico, etc…).

# **Detalhe das entidades**

**Sobre a Tipo de Utilizador:**

Tipo de utilizador (Administrador ou utente).

**Sobre os livros:**

Título;

Género;

Editora;

Obra Literária;

Preço.

**Sobre os utilizadores:**

Tipo de utilizador;

Username;

Password;

Email;

Nome Completo;

Morada;

NIB;

Data de nascimento;

BI.

**Sobre a entrega:**

Data de compra;

Data de entrega;

Estado da encomenda;

Utilizador associado;

Livro associado

**Sobre a Obra Literária:**

Nome da obra;

Data de criação da obra;

Nome do seu autor.

**Sobre a Género:**

Tipo de género (drama, comédia, fantasia, etc…).

**Sobre a Autor:**

Nome do autor;

Data de nascimento do autor.

**Sobre a editora:**

Nome da editora

# **20 questões e respostas para o sistema**

# **Procedimentos relativos às questões:**

1. *Listar os utilizadores e o seu respetivo tipo de conta (conta de administrador ou utente)?*

select uti\_nome, tpu\_tipo from utilizador

full join tipoUtilizador

on tpu\_id\_f = tpu\_id;

1. *Listar o estado dos pedidos em relação a cada obra literaria?*

*select liv\_nome "nome do livro", NVL(ent\_estado, 'Por entregar ou Sem pedidos')*

*"pedidos de obras literarias" from livros full join entregas*

*on liv\_id=ent\_id;*

1. *Quais os livros que já foram encomendados e quantas vezes?*

SELECT L.liv\_nome, COUNT( E.ent\_id ) FROM entregas E, livros L where L.liv\_id=E.liv\_id\_f group by liv\_nome;

1. *Visualizar o nome dos livros, o respetivo autor, obra literária, editora, preço e idioma?*

*select liv\_nome, aut\_nome, obr\_nome, edi\_nome, liv\_preco, liv\_idioma from*

*livros l, autor a, obraliteraria o, editora e*

*where e.edi\_id=l.edi\_id\_f*

*and o.obr\_id=l.obr\_id\_f*

*and a.aut\_id=o.aut\_id\_f;*

1. *Lista de livros ordenados do mais barato para o mais caro?*

Select \* from livros order by liv\_preco ASC;

1. *Qual o livro mais caro?*

SELECT liv\_nome From livros WHERE liv\_preco= (select MAX (liv\_preco) from livros);

1. *Visualizar o nome do utilizador que fez um pedido de entrega, o nome do livro que foi pedido por esse utilizador, a data de entrega, a data de compra e o estado atual da entrega?*

select uti\_nome, liv\_nome, ent\_dt\_entrega, ent\_dt\_compra, ent\_estado from livros l, entregas e, utilizador u

where e.uti\_id\_f=u.uti\_id and e.liv\_id\_f = l.liv\_id;

1. *Quais os diferentes idiomas existente entre a lista de livros?*

SELECT DISTINCT liv\_idioma FROM livros;

1. *Quais os diferentes livros que existem escritos no idioma “PT”?*

Select liv\_nome, liv\_idioma from livros where liv\_id = liv\_id and UPPER(liv\_idioma) like '%PT%' ;

1. *Como mostrar apenas os livros com preço inferior a 30€?*

SELECT liv\_nome, liv\_preco FROM livros where liv\_id=liv\_id group by liv\_nome, liv\_preco having liv\_preco<30;

1. *Quantos pedidos de entrega existem no total?*

*SELECT COUNT( ent\_id ) FROM entregas;*

1. *Qual é a média do preços dos livros existentes?*

SELECT AVG(liv\_preco) FROM livros;

1. *Qual é o preço a pagar no total por todos os livros existentes?*

SELECT SUM (liv\_preco) FROM livros;

1. *Qual é a inserção mais antiga na tabela de livros?*

SELECT liv\_nome FROM livros WHERE ROWNUM <=1 ORDER BY liv\_id ASC;

1. *Qual o utilizador com mais pedidos de entrega?*

select uti\_nome from utilizador, entregas

where uti\_id=ent\_id

group by uti\_nome having count(\*) = (select max(count(\*)) from entregas group by ent\_id);

1. *Quais os livros disponíveis em inglês e português?*

SELECT liv\_nome FROM livros WHERE liv\_idioma IN ('PT','EN');

1. *Quais os livros cujo preço se encontra entre 10 e 20 euros?*

SELECT liv\_nome "NOME ", liv\_preco "PRECOS" FROM livros

WHERE liv\_preco BETWEEN 10 AND 20;

1. *Qual foi o primeiro utilizador a registar-se?*

SELECT uti\_nome FROM utilizador WHERE ROWNUM <=1 ORDER BY uti\_id ASC;

1. *Listar o id de cada editora, o seu nome e respetiva quantidade de livros?*

*select edi\_id, edi\_nome, count(\*) numero\_livros from livros l*

*full join editora on*

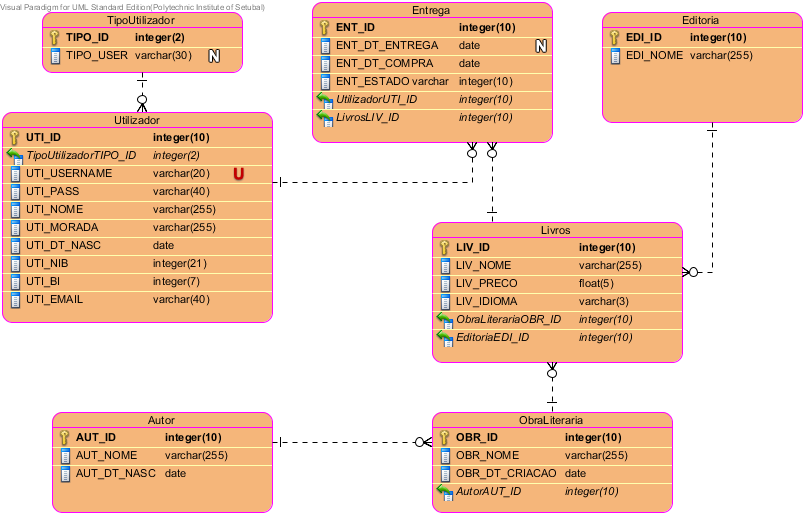
*edi\_id=edi\_id\_f group by edi\_id,edi\_nome;*

1. *Visualizar o id dos autores, o nome dos autores, a sua data de nascimento respetiva e quantas obras literárias tem cada um?*

SELECT aut\_id, aut\_nome, aut\_dt\_nasc, count(\*) num\_livros from autor a, obraliteraria o where a.aut\_id=o.aut\_id\_f group by aut\_nome, aut\_id, aut\_dt\_nasc;

# 

# **Modelo Entidade-Relação**



# **Descrição do Modelo Entidade-Relação**

**Tabela Livros:**

**Chave-primária:** liv\_id number(10) not null

**Chaves-estrangeiras:** edi\_id\_f number(10) not null

obr\_id\_f number(10) not null

**Atributos:** liv\_nome varchar2(255) not null,

liv\_preco number(5) not null,

liv\_idioma varchar2(3)

**Tabela TipoUtilizador:**

**Chave-primária:** tpu\_id number(10) not null

**Atributos:** tpu\_tipo varchar2(20) not null

**Tabela Utilizador:**

**Chave-primária:** uti\_id number(10) not null

**Chaves-estrangeiras:** tpu\_id\_f number(10) not null

**Atributos:** uti\_user varchar2(20) not null,

uti\_pass varchar2(20) not null,

uti\_nome varchar2(255) not null,

uti\_morada varchar2(255) not null,

uti\_dt\_nasc Date not null,

uti\_nib number(21) not null,

uti\_bi number(7) not null,

uti\_email varchar2(40) not null

**Tabela Entregas:**

**Chave-primária:** ent\_id number(10) not null

**Chaves-estrangeiras** uti\_id\_f number(10) not null,

liv\_id\_f number(10) not null

**Atributos:** ent\_dt\_entrega Date,

ent\_dt\_compra Date not null,

ent\_estado varchar(20)

**Tabela ObraLiteraria:**

**Chave-primária:** obr\_id number(10) not null

**Chaves-estrangeiras:** aut\_id\_f number(10) not null

**Atributos**: obr\_nome varchar(255) not null,

obr\_dt\_criacao Date not null

**Tabela Autor:**

**Chave-primária:** aut\_id number(10)

**Atributos:** aut\_nome varchar2(255),

aut\_dt\_nasc Date

**Tabela Genero:**

**Chave-primária:** gen\_id number(10) not null

**Atributos:** gen\_nome varchar(255) not null

**Tabela Editora:**

**Chave-primária:** edi\_id number(10) not null

**Atributos:** edi\_nome varchar(255) not null

**Tabela Visualizacoes:**

**Chave-primária:** vis\_id number(10) not null

**Atributos:** vis\_numero number(38) not null

**Tabela GeneroLivros:**

**Chave-primária:** liv\_id\_f number(10)

**Chave-estrangeira:** gen\_id\_f number(10)

# **Pacotes, Procedimentos e Funções**

# **Pacotes relativos às tabelas:**

# **Autores:**

create or replace

PACKAGE P\_AUTOR AS

/\* TODO enter package declarations (types, exceptions, methods etc) here \*/

PROCEDURE ins (p\_aut\_nome IN VARCHAR2, p\_aut\_dt\_nasc IN DATE, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

PROCEDURE sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

PROCEDURE del(p\_aut\_id IN NUMBER,p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

PROCEDURE upd (p\_aut\_id IN NUMBER, p\_aut\_nome IN VARCHAR2, p\_aut\_dt\_nasc IN DATE, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

PROCEDURE selByNome(p\_aut\_nome IN VARCHAR2, p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

END P\_AUTOR;

create or replace

PACKAGE BODY P\_AUTOR AS

PROCEDURE ins (p\_aut\_nome IN VARCHAR2, p\_aut\_dt\_nasc IN DATE, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

INSERT INTO autor(aut\_id,aut\_nome,aut\_dt\_nasc) VALUES(seq\_Autor.nextVal,p\_aut\_nome, p\_aut\_dt\_nasc);

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

PROCEDURE sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

BEGIN

p\_error:=0;

OPEN p\_cursor FOR SELECT aut\_id, aut\_nome, aut\_dt\_nasc FROM autor;

EXCEPTION

WHEN OTHERS THEN

p\_error := 1;

p\_errormsg := 'Erro no select: '||sqlerrm;

END;

PROCEDURE del(p\_aut\_id IN NUMBER,p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

delete from generolivros where liv\_id\_f in (select liv\_id from livros where obr\_id\_f in (select obr\_id from obraliteraria where aut\_id\_f=p\_aut\_id));

delete from entregas where liv\_id\_f in (select liv\_id from livros where obr\_id\_f in (select obr\_id from obraliteraria where aut\_id\_f=p\_aut\_id));

delete from livros where obr\_id\_f in (select obr\_id from obraliteraria where aut\_id\_f=p\_aut\_id);

DELETE FROM generolivros WHERE EXISTS (select liv\_id from livros where exists (select obr\_id from obraliteraria where aut\_id\_f=p\_aut\_id));

DELETE FROM obraliteraria WHERE aut\_id\_f=p\_aut\_id;

DELETE FROM autor WHERE aut\_id = p\_aut\_id;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

PROCEDURE upd (p\_aut\_id IN NUMBER, p\_aut\_nome IN VARCHAR2, p\_aut\_dt\_nasc IN DATE, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

UPDATE autor SET aut\_nome = p\_aut\_nome, aut\_dt\_nasc = p\_aut\_dt\_nasc WHERE aut\_id=p\_aut\_id;

COMMIT;

EXCEPTION WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: ' || sqlerrm;

END;

PROCEDURE selByNome(p\_aut\_nome IN VARCHAR2, p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

BEGIN

p\_error:=0;

OPEN p\_cursor FOR SELECT aut\_nome, aut\_dt\_nasc FROM autor WHERE aut\_nome=p\_aut\_nome;

EXCEPTION

WHEN OTHERS THEN

p\_error := 1;

p\_errormsg := 'Erro no select: '||sqlerrm;

END;

END P\_AUTOR;

# **Editora:**

create or replace

PACKAGE P\_EDITORA AS

PROCEDURE ins (p\_edi\_nome IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

PROCEDURE del(p\_edi\_id IN NUMBER,p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

PROCEDURE upd (p\_edi\_id IN NUMBER, p\_edi\_nome IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

END P\_EDITORA;

create or replace

PACKAGE BODY P\_EDITORA AS

PROCEDURE ins (p\_edi\_nome IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

INSERT INTO editora (edi\_id,edi\_nome) VALUES(seq\_Editora.nextVal,p\_edi\_nome);

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

Begin

p\_error:=0;

OPEN p\_cursor FOR SELECT edi\_id, edi\_nome FROM Editora;

EXCEPTION

WHEN OTHERS THEN

p\_error := 1;

p\_errormsg := 'Erro no select: '||sqlerrm;

END;

PROCEDURE del(p\_edi\_id IN NUMBER,p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

DELETE FROM livros WHERE edi\_id\_f=p\_edi\_id;

DELETE FROM editora WHERE edi\_id = p\_edi\_id;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

PROCEDURE upd (p\_edi\_id IN NUMBER, p\_edi\_nome IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

UPDATE editora SET edi\_nome = p\_edi\_nome WHERE edi\_id = p\_edi\_id;

COMMIT;

EXCEPTION WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: ' || sqlerrm;

END;

END P\_EDITORA;

# **Entregas:**

create or replace

PACKAGE P\_ENTREGAS AS

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

procedure ins(p\_liv\_nome in varchar2,p\_uti\_id\_f in number,p\_error out number, p\_errormsg out varchar2) ;

END P\_ENTREGAS;

create or replace

PACKAGE BODY P\_ENTREGAS AS

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

Begin

p\_error:=0;

Open p\_cursor for select ent\_dt\_compra, ent\_dt\_entrega, ent\_estado, ent\_id, liv\_id\_f, uti\_id\_f from entregas;

null;

exception when others then p\_error:=1;

p\_errormsg:='Select in entregas:'||sqlerrm ;

end;

procedure ins(p\_liv\_nome in varchar2,p\_uti\_id\_f in number,p\_error out number, p\_errormsg out varchar2) IS

BEGIN

p\_error := 0;

INSERT INTO entregas (ent\_id, uti\_id\_f, liv\_id\_f,ent\_dt\_entrega, ent\_dt\_compra, ent\_estado) VALUES(seq\_Entregas.nextVal, p\_uti\_id\_f, (select liv\_id from livros where liv\_nome=p\_liv\_nome), SYSDATE+5, SYSDATE, 'expedida');

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_error := 1;

p\_errormsg := 'Erro: '||sqlerrm;

END;

END P\_ENTREGAS;

# **Género:**

create or replace

PACKAGE P\_GENERO AS procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

END P\_GENERO;

create or replace

PACKAGE BODY P\_GENERO AS

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

Begin

p\_error:=0;

Open p\_cursor for select gen\_id, gen\_nome from genero;

null;

exception when others then p\_error:=1;

p\_errormsg:='Select in entregas:'||sqlerrm ;

end;

END P\_GENERO;

# **Livro:**

create or replace

PACKAGE P\_LIVRO AS

/\* TODO enter package declarations (types, exceptions, methods etc) here \*/

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

PROCEDURE insLivro(p\_liv\_nome IN VARCHAR2,p\_liv\_preco IN NUMBER, p\_liv\_idioma IN VARCHAR2, p\_edi\_id\_f IN NUMBER, p\_obr\_id\_f IN NUMBER, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

PROCEDURE upd(p\_liv\_id IN NUMBER,p\_liv\_nome IN VARCHAR2, p\_liv\_preco IN NUMBER, p\_liv\_idioma IN VARCHAR2, p\_erro OUT NUMBER, p\_msg\_erro OUT VARCHAR2);

PROCEDURE del(p\_liv\_id IN NUMBER, p\_erro OUT NUMBER, p\_msg\_erro OUT VARCHAR2);

PROCEDURE selByNome(p\_liv\_nome IN VARCHAR2, p\_error OUT NUMBER, p\_error\_msg OUT VARCHAR2, p\_cursor OUT sys\_refcursor);

END P\_LIVRO;

create or replace

PACKAGE BODY P\_LIVRO

AS

PROCEDURE sel(

p\_error OUT NUMBER,

p\_errormsg OUT VARCHAR2,

p\_cursor OUT sys\_refcursor)

IS

BEGIN

p\_error:=0;

OPEN p\_cursor FOR SELECT liv\_id, edi\_id\_f, obr\_id\_f, liv\_nome, TO\_CHAR(liv\_preco, '99999999') liv\_preco, liv\_idioma FROM livros;

EXCEPTION

WHEN OTHERS THEN

p\_error := 1;

p\_errormsg := 'Erro no select: '||sqlerrm;

END;

PROCEDURE insLivro(

p\_liv\_nome IN VARCHAR2,

p\_liv\_preco IN NUMBER,

p\_liv\_idioma IN VARCHAR2,

p\_edi\_id\_f IN NUMBER,

p\_obr\_id\_f IN NUMBER,

p\_erro OUT NUMBER,

p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

INSERT

INTO livros

(

liv\_id,

edi\_id\_f,

obr\_id\_f,

liv\_nome,

liv\_preco,

liv\_idioma

)

VALUES

(

seq\_Livros.nextVal,

p\_edi\_id\_f,

p\_obr\_id\_f,

p\_liv\_nome,

p\_liv\_preco,

p\_liv\_idioma

);

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

--Actualizar Livro

PROCEDURE upd

(

p\_liv\_id IN NUMBER,

p\_liv\_nome IN VARCHAR2,

p\_liv\_preco IN NUMBER,

p\_liv\_idioma IN VARCHAR2,

p\_erro OUT NUMBER,

p\_msg\_erro OUT VARCHAR2

)

IS

BEGIN

p\_erro := 0;

UPDATE livros

SET liv\_nome = p\_liv\_nome,

liv\_preco = p\_liv\_preco,

liv\_idioma = p\_liv\_idioma

WHERE liv\_id = p\_liv\_id;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: ' || sqlerrm;

END;

PROCEDURE del(

p\_liv\_id IN NUMBER,

p\_erro OUT NUMBER,

p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

DELETE FROM generolivros WHERE liv\_id\_f=p\_liv\_id;

DELETE FROM entregas WHERE liv\_id\_f=p\_liv\_id;

DELETE FROM livros WHERE liv\_id = p\_liv\_id;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

--Selecionar por nome

PROCEDURE selByNome(

p\_liv\_nome IN VARCHAR2,

p\_error OUT NUMBER,

p\_error\_msg OUT VARCHAR2,

p\_cursor OUT sys\_refcursor)

IS

BEGIN

p\_error := 0;

OPEN p\_cursor FOR SELECT liv\_nome,

TO\_CHAR(liv\_preco, '99999999') liv\_preco,

liv\_idioma FROM livros WHERE liv\_nome=p\_liv\_nome;

EXCEPTION

WHEN OTHERS THEN

p\_error := 1;

p\_error\_msg := 'Erro no select: '||sqlerrm;

END;

END P\_LIVRO;

# **ObraLiteraria:**

create or replace

PACKAGE P\_OBRALITERARIA AS

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

END P\_OBRALITERARIA;

create or replace

PACKAGE BODY P\_OBRALITERARIA AS

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

Begin

p\_error:=0;

Open p\_cursor for select aut\_id\_f, obr\_dt\_criacao, obr\_id, obr\_nome from obraliteraria;

null;

exception when others then p\_error:=1;

p\_errormsg:='Select in autores:'||sqlerrm ;

end;

END P\_OBRALITERARIA;

# **TipoUtilizador**

create or replace

PACKAGE P\_TIPOUTILIZADOR AS

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

END P\_TIPOUTILIZADOR;

create or replace

PACKAGE BODY P\_TIPOUTILIZADOR AS

procedure sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

Begin

p\_error:=0;

Open p\_cursor for select tpu\_id, tpu\_tipo from tipoutilizador;

null;

exception when others then p\_error:=1;

p\_errormsg:='Select in tipoutilizadpr'||sqlerrm ;

end;

END P\_TIPOUTILIZADOR;

# **Utilizador**

create or replace

PACKAGE P\_UTILIZADORES AS

/\* TODO enter package declarations (types, exceptions, methods etc) here \*/

PROCEDURE ins (p\_tpu\_id\_f IN NUMBER, p\_uti\_user IN VARCHAR2, p\_uti\_pass IN VARCHAR2, p\_uti\_nome IN VARCHAR2, p\_uti\_morada IN VARCHAR2, p\_uti\_dt\_nasc IN DATE, p\_uti\_nib IN NUMBER, p\_uti\_bi IN NUMBER, p\_uti\_email IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

PROCEDURE sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor);

PROCEDURE del(p\_uti\_id IN NUMBER,p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

PROCEDURE upd (p\_uti\_id IN NUMBER, p\_tpu\_id\_f IN NUMBER, p\_uti\_user IN VARCHAR2, p\_uti\_pass IN VARCHAR2, p\_uti\_nome IN VARCHAR2, p\_uti\_morada IN VARCHAR2, p\_uti\_dt\_nasc IN DATE, p\_uti\_nib IN NUMBER, p\_uti\_bi IN NUMBER, p\_uti\_email IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2);

END P\_UTILIZADORES;

create or replace

PACKAGE BODY P\_UTILIZADORES AS

PROCEDURE ins (p\_tpu\_id\_f IN NUMBER, p\_uti\_user IN VARCHAR2,

p\_uti\_pass IN VARCHAR2, p\_uti\_nome IN VARCHAR2, p\_uti\_morada IN VARCHAR2, p\_uti\_dt\_nasc IN DATE,

p\_uti\_nib IN NUMBER, p\_uti\_bi IN NUMBER, p\_uti\_email IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

INSERT INTO utilizador(uti\_id, tpu\_id\_f, uti\_user, uti\_pass, uti\_nome, uti\_morada, uti\_dt\_nasc, uti\_nib, uti\_bi, uti\_email)

VALUES(seq\_Utilizador.nextVal, p\_tpu\_id\_f, p\_uti\_user, p\_uti\_pass , p\_uti\_nome , p\_uti\_morada, p\_uti\_dt\_nasc, p\_uti\_nib , p\_uti\_bi , p\_uti\_email);

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

PROCEDURE sel(p\_error out number, p\_errormsg out varchar2, p\_cursor out sys\_refcursor) IS

BEGIN

p\_error:=0;

OPEN p\_cursor FOR SELECT uti\_id, tpu\_id\_f, uti\_user, uti\_pass, uti\_nome, uti\_morada, uti\_dt\_nasc, uti\_nib, uti\_bi, uti\_email FROM utilizador;

EXCEPTION

WHEN OTHERS THEN

p\_error := 1;

p\_errormsg := 'Erro no select: '||sqlerrm;

END;

PROCEDURE del(p\_uti\_id IN NUMBER,p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

DELETE FROM entregas WHERE uti\_id\_f = p\_uti\_id;

DELETE FROM utilizador WHERE uti\_id = p\_uti\_id;

COMMIT;

EXCEPTION

WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: '||sqlerrm;

END;

PROCEDURE upd (p\_uti\_id IN NUMBER, p\_tpu\_id\_f IN NUMBER, p\_uti\_user IN VARCHAR2,

p\_uti\_pass IN VARCHAR2, p\_uti\_nome IN VARCHAR2, p\_uti\_morada IN VARCHAR2, p\_uti\_dt\_nasc IN DATE,

p\_uti\_nib IN NUMBER, p\_uti\_bi IN NUMBER, p\_uti\_email IN VARCHAR2, p\_erro OUT NUMBER,p\_msg\_erro OUT VARCHAR2)

IS

BEGIN

p\_erro := 0;

UPDATE utilizador SET tpu\_id\_f=p\_tpu\_id\_f, uti\_user=p\_uti\_user, uti\_pass=p\_uti\_pass,

uti\_nome=p\_uti\_nome, uti\_morada=p\_uti\_morada, uti\_dt\_nasc=p\_uti\_dt\_nasc, uti\_nib=p\_uti\_nib, uti\_bi=p\_uti\_bi, uti\_email=p\_uti\_email WHERE uti\_id= p\_uti\_id;

COMMIT;

EXCEPTION WHEN OTHERS THEN

p\_erro := 1;

p\_msg\_erro := 'Erro: ' || sqlerrm;

END;

END P\_UTILIZADORES;

# **Procedimentos relativos à listagem de tabelas:**

* Listar Livros
* Listar Autores
* Listar Visualizações
* Listar Utilizadores
* Listar Editoras
* Listar Géneros
* Listar Obra Literárias
* Listar Entregas
* Listar Tipos de Utilizador

# **Procedimentos relativos à inserção de dados em tabelas:**

* Inserir Livros
* Inserir Autores
* Inserir Entregas
* Inserir Obra Literária
* Inserir Editora
* Inserir Utilizadores

# **Procedimentos relativos à atualização de dados em tabelas:**

* Atualizar Livros
* Atualizar Entregas
* Atualizar Visualizações
* Atualizar Autores
* Atualizar Utilizadores

# **Procedimentos relativos à remoção de dados em tabelas:**

* Remover Livros
* Remover Utilizadores
* Remover Autores

# **Views**

**AUTOR\_VIEW**

* Esta lista mostra os atributos nome do autor e data de nascimento do mesmo.

**EDITORA\_VIEW**

* Esta lista mostra o nome da editoea.

**LIVRO\_VIEW**

* Esta lista mostra o nome, o preço e o idioma dos livros em questão.

**UTILIZADOR\_VIEW**

* Esta lista mostra o tipo de utilizador e o nome do utilizador.

**OBRALITERARIA\_VIEW**

* Esta lista mostra o nome da obra literária e a data em que esta foi criada.

# **Anexos**

Código SQL para criar a base de dados:

drop table TipoUtilizador cascade constraints;

drop table Livros cascade constraints;

drop table Utilizador cascade constraints;

drop table Entregas cascade constraints;

drop table ObraLiteraria cascade constraints;

drop table Autor cascade constraints;

drop table Editora cascade constraints;

drop table Genero cascade constraints;

drop table Visualizacoes cascade constraints;

drop table GeneroLivros cascade constraints;

create table TipoUtilizador(

tpu\_id number(10) not null,

tpu\_tipo varchar2(20) not null

);

create table Livros(

liv\_id number(10) not null,

edi\_id\_f number(10) not null,

obr\_id\_f number(10) not null,

liv\_nome varchar2(255) not null,

liv\_preco number(5) not null,

liv\_idioma varchar2(3)

);

create table Utilizador(

uti\_id number(10) not null,

tpu\_id\_f number(10) not null,

uti\_user varchar2(20) not null,

uti\_pass varchar2(20) not null,

uti\_nome varchar2(255) not null,

uti\_morada varchar2(255) not null,

uti\_dt\_nasc Date not null,

uti\_nib number(21) not null,

uti\_bi number(7) not null,

uti\_email varchar2(40) not null

);

create table Entregas(

ent\_id number(10) not null,

uti\_id\_f number(10) not null,

liv\_id\_f number(10) not null,

ent\_dt\_entrega Date,

ent\_dt\_compra Date not null,

ent\_estado varchar(20)

);

create table ObraLiteraria(

obr\_id number(10) not null,

aut\_id\_f number(10) not null,

obr\_nome varchar(255) not null,

obr\_dt\_criacao Date not null

);

create table Autor(

aut\_id number(10),

aut\_nome varchar2(255),

aut\_dt\_nasc Date

);

create table Genero(

gen\_id number(10) not null,

gen\_nome varchar(255) not null

);

create table Editora(

edi\_id number(10) not null,

edi\_nome varchar(255) not null

);

create table Visualizacoes(

vis\_id number(10) not null,

vis\_numero number(38) not null

);

create table GeneroLivros(

liv\_id\_f number(10),

gen\_id\_f number(10)

);

alter table Livros add constraint pk\_Livros primary key(liv\_id);

alter table TipoUtilizador add constraint pk\_TipoUtilizador primary key(tpu\_id);

alter table Utilizador add constraint pk\_Utilizador primary key(uti\_id);

alter table Entregas add constraint pk\_Entregas primary key(ent\_id);

alter table ObraLiteraria add constraint pk\_ObraLiteraria primary key(obr\_id);

alter table Autor add constraint pk\_Autor primary key(aut\_id);

alter table Visualizacoes add constraint pk\_Visualizacoes primary key(vis\_id);

alter table Genero add constraint pk\_Genero primary key(gen\_id);

alter table Editora add constraint pk\_Editora primary key(edi\_id);

alter table GeneroLivros add constraint pk\_GeneroLivros primary key(liv\_id\_f, gen\_id\_f);

alter table Livros add constraint Livros\_fk\_ObraLiteraria foreign key(obr\_id\_f) references ObraLiteraria(obr\_id);

alter table Livros add constraint Livros\_fk\_Editora foreign key(edi\_id\_f) references Editora(edi\_id);

alter table Utilizador add constraint Utilizador\_fk\_TipoUtilizador foreign key(tpu\_id\_f) references TipoUtilizador(tpu\_id);

alter table ObraLiteraria add constraint ObraLiteraria\_fk\_Autor foreign key(aut\_id\_f) references Autor(aut\_id);

alter table GeneroLivros add constraint GeneroLivros\_fk\_Livros foreign key(liv\_id\_f) references Livros(liv\_id);

alter table GeneroLivros add constraint GeneroLivros\_fk\_Genero foreign key(gen\_id\_f) references Genero(gen\_id);

alter table Entregas add constraint Entregas\_fk\_Utilizador foreign key(uti\_id\_f) references Utilizador(uti\_id);

alter table Entregas add constraint Entregas\_fk\_Livros foreign key(liv\_id\_f) references Livros(liv\_id);

Endereço website: http://tpsi.estsetubal.ips.pt:7777/bd2013/